RK611/RK06

USERDEFINED TEST

MD-11-DZR6R-B

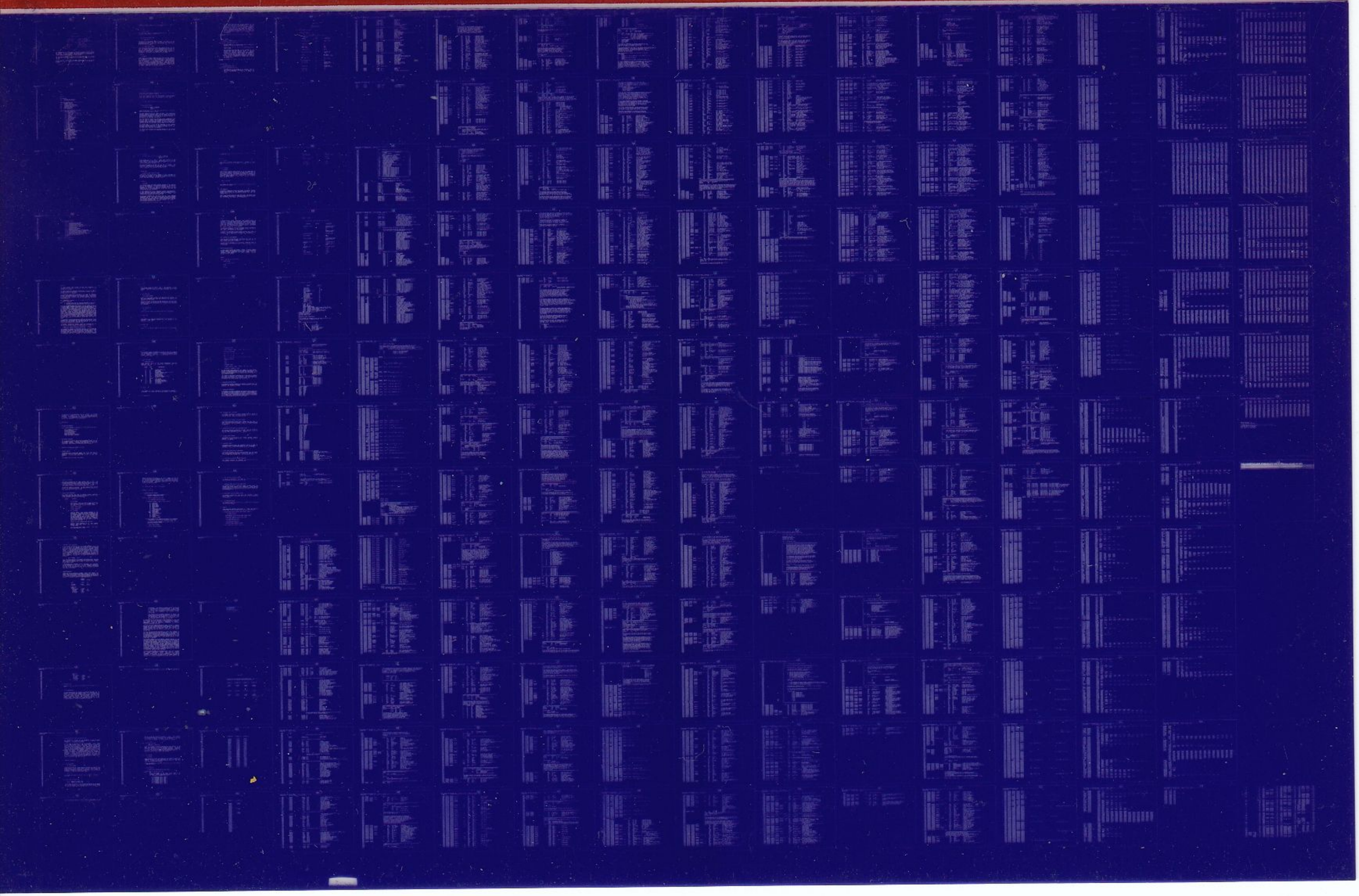
EP-DZR6R-B-DL-B

FICHE 1 OF 1

NOV 1976

digital

MADE IN USA



.REM 0

IDENTIFICATION

PRODUCT CODE:	MAINDEC-11-DZR6R-B-D
PRODUCT NAME:	RK611/RK06 USER DEFINED TEST
DATE:	AUGUST, 1976
MAINTAINER:	DIAGNOSTIC GROUP
AUTHOR:	MARV TEGROTENHUIS

TABLE OF CONTENTS

```
  102                                   4.4     TIME OUT
  103                                   4.5     TEST LOOPING AND LOOP COUNTERS
```

1.0 ABSTRACT

THE USER DEFINED TEST PROGRAM PROVIDES THE CAPABILITY OF ENTERING, EDITING, SAVING, RECALLING, AND EXECUTING TEST PROGRAMS DESIGNED BY THE USER.

THE USER DEFINED TEST OPERATES INTERACTIVELY TO ALLOW THE USER TO DEVELOP A SPECIFIC TEST MADE UP OF SUBSYSTEM COMMANDS, CHECKING, AND REPORTING IN ANY SEQUENCE.

AN INTERACTIVE COMMAND SET IS DEFINED TO BE USED IN ENTERING, STORING, RETRIEVING, EDITING, AND EXECUTING TESTS. THIS COMMAND SET INCLUDES OTHER COMMANDS THAT PERFORM COMPARE OPERATIONS, CHANGE TEST CONTROL VALUES, INITIALIZE THE BUFFER, AND INITIALIZE THE SUBSYSTEM.

THE INTERACTIVE COMMAND SET IS DIVIDED INTO TWO TYPES OF COMMANDS. THESE ARE:

* DEFERRED WHICH ARE THE COMMANDS THAT MAKE UP THE TESTS.
* IMMEDIATE WHICH ARE EXECUTED WHEN THEY ARE ENTERED.

WHEN THE DEFERRED COMMANDS ARE ENTERED THEY ARE STORED IN CORE IN A SOURCE AREA. EDITING CAN BE DONE ON THE STORED SOURCE TO ADD OR DELETE SPECIFIC LINES. AFTER THE SOURCE HAS BEEN ENTERED, IT IS "COMPILED" INTO OBJECT CODE, STORED IN CORE IN AN OBJECT AREA, AND EXECUTED. ONCE EXECUTION BEGINS, THE SOURCE CODE IS LOST. THE OBJECT CODE IS PRESERVED UNTIL ANOTHER COMPILE AND CAN BE REEXECUTED. EITHER THE SOURCE CODE BEFORE EXECUTION OR THE OBJECT CODE AFTER COMPILATION CAN BE PUNCHED OUT ON PAPER TAPE ALONG WITH ANY SPECIAL DATA PATTERNS ENTERED. CONVERSELY, EITHER TYPE OF CODE CAN BE READ FROM PAPER TAPE. SOURCE CODE IS PLACED IN THE SOURCE AREA AND OBJECT CODE IS PLACED IN THE OBJECT AREA AND EXECUTED. IF SPECIAL DATA PATTERNS WERE PUNCHED, THESE PATTERNS ARE PLACED IN THE APPROPRIATE BUFFER WHEN THE TAPE IS READ.

THE IMMEDIATE COMMANDS ARE EXECUTED WHEN THEY ARE ENTERED. THESE COMMANDS ARE NOT ENTERED INTO THE SOURCE AREA AND DO NOT BECOME PART OF THE COMPILED TEST. THEY CAN BE EXECUTED AT ANY TIME EXCEPT WHILE COMPILING OR EXECUTING A TEST.

THE GENERAL INTERACTIVE COMMAND (WITH THE EXCEPTION OF THE SPECIAL DATA PATTERN COMMAND, SEE PARAGRAPH 8.1.5) FORMAT IS:

    INTERACTIVE COMMAND,PARAMETER1,PARAMETER2,...(RETURN)

THE INTERACTIVE COMMAND IS TESTED FOR LEGALITY. IF IT IS NOT ONE THE DEFINED COMMANDS, THE COMMAND IS REJECTED AND AN ERROR MESSAGE IS PRINTED. THE COMMAND IN ERROR IS ECHOED BACK TO SHOW THE ERROR AND THE ENTIRE COMMAND MUST BE REENTERED. THE PARAMETERS ARE ACCEPTED WITHOUT CHECKING UNLESS SPECIFIC CHECKING IS DEFINED FOR THAT COMMAND IN THE FOLLOWING COMMAND

184 DESCRIPTIONS.

ANY ONE OR ALL OF THE PARAMETERS MAY BE OMITTED. AN OMITTED PARAMETER WILL DEFAULT TO THE VALUE LAST SPECIFIED FOR THAT PARAMETER. A CARRIAGE RETURN WILL CAUSE THE REMAINING PARAMETERS TO DEFAULT. IF A PARAMETER IS TO BE SPECIFIED AFTER ONE THAT IS OMITTED, THE SEPARATOR (,) MUST BE PROVIDED.

2.0 REQUIREMENTS

2.1 HARDWARE REQUIREMENTS

THE FOLLOWING HARDWARE IS REQUIRED TO RUN THE USER DEFINED TEST:

PDP-11 SYSTEM (16K MEMORY)
CONSOLE TERMINAL
RK06 UNIBUS CONTROLLER
1 TO 8 RK06 DRIVES
RK06 DISK CARTRIDGE.
PAPER TAPE READER (OPTIONAL)
PAPER TAPE PUNCH (OPTIONAL)

2.2 PRELIMINARY PROGRAMS

THE CONTROLLER DIAGNOSTIC AND/OR DRIVE DIAGNOSTIC SHOULD BE RUN TO DIAGNOSE FAULTS. HOWEVER, THIS PROGRAM DOES NOT RELY ON AN OPERATIONAL SUBSYSTEM. FEATURES ARE PROVIDED TO FACILITATE LOOPING ON A TEST OR ERROR FOR TROUBLESHOOTING PURPOSES.

3.0 OPERATING PROCEDURE AND CONTROL FUNCTIONS

3.1 PROGRAM LOADING

THE PROGRAM CAN BE LOADED FROM PAPER TAPE USING THE ABSOLUTE LOADER OR FROM ANY MEDIA SUPPORTED BY XXDP BUT IT IS NOT CHAINABLE. IT CAN ALSO BE LOADED BY ACT OR APT IN DUMP MODE ONLY.

THE PROGRAM DOES NOT DESTROY THE LOADER.

3.2 STARTING LOCATIONS

THE STARTING ADDRESS FOR THE USER DEFINED TEST IS 200. THE PROGRAM IDENTIFIES ITSELF, COMPUTES AND TYPES THE MAXIMUM NUMBER OF WORDS FOR DATA TRANSFER COMMANDS (BASED ON MEMORY SIZE), AND TYPES THE MESSAGE "TYPE HP TO PRINT HELP FILE". A STAR (*) IS THEN PRINTED TO SIGNIFY THE PROGRAM IS READY FOR A COMMAND.

THE RESTART ADDRESS IS ALSO 200. THE SAME MESSAGES ARE PRINTED EXCEPT FOR THE "HELP" MESSAGE. THE HELP FILE IS AVAILABLE ONLY WHEN THE PROGRAM IS INITIALLY LOADED.

3.3 CONSOLE SWITCH REGISTER

THE CONSOLE SWITCHES ARE USED TO PROVIDE CONTROL FUNCTIONS. THESE FUNCTIONS AND SWITCH ASSIGNMENTS GENERALLY CONFORM TO THE SYSMAC STANDARD WITH THE EXCEPTIONS NOTED:

SWITCH	FUNCTION
15	HALT ON ERROR
14	LOOP ON TEST. WHEN THE SWITCH IS RESET AFTER THE TEST HAS BEEN LOOPING THE PROGRAM WILL TYPE NNN=NUMBER OF LOOPS. NNN IS A DECIMAL NUMBER.
13	INHIBIT ERROR TYPEOUT
11	INHIBIT INERATION
10	BELL ON ERROR
9	LOOP ON ERROR. THIS SWITCH CONFORMS TO THE STANDARD IN THAT WHEN THE ERROR IS DETECTE , THE TEST PRESENTLY UNDER EXECUTION IS RESTARTED WITH THE FIRST COMMAND. IF THE TEST EVER COMPLETES WITHOUT AN ERROR THE TEST IS AGAIN STARTED FROM THE FIRST COMMAND AS LONG AS SWITCH 9 REMAINS SET. WHEN SWITCH 9 IS RESET THE LOOP WILL TERMINATE AFTER 1 MORE PASS AND TYPE THE NUMBER OF LOOPS AS WHEN SWITCH 9 WAS RESET.
2	INHIBIT ALL DATA COMPARE ERROR REPORTING.
1	WHEN SET FORCE REPORTING OF ALL DATA COMPARE ERRORS. WHEN RESET REPORT ONLY THE FIRST 10 COMPARE ERRORS.
0	WHEN SET FORCE SHORT ERROR REPORT. WHEN RESET FULL ERROR REPORT IS GIVEN.

3.4 '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 RK611 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.

3.5 UNIBUS ADDRESSES

UNIBUS AND VECTOR ADDRESS OF THE RK06, PAPER TAPE READER, AND PAPER TAPE PUNCH CAN BE CHANGED FROM THE DEFAULT AS PART OF THE PROGRAM STARTUP PROCEDURE. THE METHOD OF CHANGING THE PARAMETERS IS TO ALTER THE MEMORY LOCATIONS SPECIFIED BEFORE THE PROGRAM IS STARTED. THE DEFAULT VALUES OF THESE PARAMETERS ARE:

	UNIBUS ADDRESS	VECTOR
RK06	277400	210
TAG NAME	RKBAS	RKVEC
LOCATION	23342	23344
PAPER TAPE READER		
DATA BUFFER	177552	
TAG NAME	PTRDB	NOT
LOCATION	1730	USED
STATUS REG	177550	
TAG NAME	PTRSR	

350 LOCATION 1726

```
                                PAPER TAPE
                                PUNCH
                                  DATA BUFFER     177556
                                  TAG NAME        PTPDB       NOT
                                  LOCATION        1734        USED
                                  STATUS REG      177554
                                  TAG NAME        PTPSR
                                  LOCATION        1732
```

3.6 EXECUTION TIME

EXECUTION TIME WILL DEPEND ON THE SPECIFIC TEST DEFINED BY THE USER.

3.7 TEST PROGRAM SIZE

THE TEST PROGRAM SIZE IS LIMITED BY THE STORAGE PROVIDED IN THE PROGRAM FOR SOURCE AND OBJECT CODE. IT IS NOT POSSIBLE TO SPECIFY THE EXACT MAXIMUM NUMBER OF SOURCE LINES POSSIBLE SINCE THE SOURCE LINES AND THE RESULTANT OBJECT CODE VARIES FROM COMMAND TO COMMAND. HOWEVER, THE APPROXIMATE MAXIMUM IS 200 SOURCE LINES. WHEN THE TEST IS ENTERED AND WHEN THE TEST IS COMPILED CHECKS ARE PERFORMED TO INSURE THAT NEITHER THE SOURCE OR OBJECT CODE WILL EXCEED ITS RESPECTIVE STORAGE.

4.0 USER DEFINED TEST FUNCTIONAL DESCRIPTION

THE INTERACTIVE COMMAND SET IS DESCRIBED IN THE FOLLOWING PARAGRAPHS. THE IMMEDIATE COMMAND SET IS DESCRIBED FIRST, FOLLOWED BY THE DEFERRED COMMAND SET.

THE FOLLOWING IS AN EXAMPLE OF WHAT IS CONSIDERED TYPICAL USAGE OF THE USER DEFINED TEST. THE USER WILL USE THE IMMEDIATE COMMANDS TO SET UP THE ITERATION COUNT AND TIMEOUT, SELECT THE DRIVE, INPUT DATA PATTERNS, ETC. THEN, CHOSING FROM THE DEFERRED COMMANDS, THE TEST IS ENTERED. AT ANY TIME WHILE ENTERING THE TEST THE PRINT AND EDIT COMMANDS MAY BE USED TO DISPLAY AND/OR CHANGE THE ENTERED DEFERRED COMMANDS. AFTER THE TEST HAS BEEN ENTERED AND EDITED, THE COMPILE COMMAND IS EXECUTED. IF THE COMPILE IS SUCCESSFUL (NO ERRORS REPORTED) THE .OT,S COMMAND IS EXECUTED TO STORE THE SOURCE AND ALL ENTERED DATA PATTERNS ON PAPER TAPE. THE RUN COMMAND IS THEN USED TO HAVE THE TEST EXECUTED. IF A CHANGE TO THE PROGRAM IS DESIRED, THE SOURCE TYPE IS READ IN. THE EDITING PERFORMED, COMPILED AGAIN, PUNCHED AGAIN, ETC. AT ANY TIME THE OBJECT CODE MAY BE SAVED BY DOING AN OT,O COMMAND TO PUNCH A TAPE WITH THAT CODE.

4.1 IMMEDIATE COMMAND SET

4.1.1 DRIVE SELECTION

DN, DRIVE NUMBER

WHERE DRIVE NUMBER IS A SINGLE DIGIT 0 THROUGH 7 TO SPECIFY THE DRIVE TESTED. THIS DRIVE NUMBER WILL BE USED UNTIL EITHER ALTERED BY ANOTHER DN COMMAND OR A DIFFERENT DRIVE NUMBER IS SPECIFIED AS PART OF A SUBSYSTEM COMMAND (SF). WHEN THE PROGRAM IS INITIALLY LOADED THE DRIVE NUMBER IS SET TO 1.

DN,?

WILL CAUSE THE NUMBER OF THE DRIVE THAT IS PRESENTLY SELECTED TO BE TYPED.

4.1.2 OUTPUT TEST TO PAPER TAPE

OT,O WHERE O IS OBJECT CODE
OT,S WHERE S IS SOURCE CODE

THIS COMMAND IS PROVIDED TO ALLOW EITHER THE SOURCE OR THE OBJECT CODE TO BE PUNCHED. ONLY THE SOURCE CODE CAN BE PUNCHED BEFORE THE "COMPILE COMMAND HAS BEEN ISSUED, BOTH CAN BE PUNCHED AFTER

```
438                                THE  "COMPILE" BUT BEFORE THE "RUN", AND ONLY THE OBJECT CODE CAN
439                                BE PUNCHED AFTER THE "RUN".  ALL TEST SPECIFIC PARAMETERS  (DRIVE
```

NUMBER, CYLINDER, TRACK, ETC.), THE USER DEFINED DATA PATTERNS, AND THE RANDOM DATA PATTERN ARE ALSO PUNCHED.

4.1.3 COPY TAPE

CT (COPY TAPE)

THE PAPER TAPE LOADED IN THE READER IS REPRODUCED ON THE PUNCH. IT MAY BE EITHER SOURCE OR OBJECT CODE.

4.1.4 INPUT TEST FROM PAPER TAPE.

IT (INPUT TEST)

THE NEXT TEST ON THE PAPER TAPE IS READ. THE TEST WILL BE RECOGNIZED AS SOURCE OR OBJECT CODE AND THE CODE PLACED IN THE APPROPRIATE BUFFER. CONTROL WILL BE RETURNED TO THE CONSOLE AFTER THE TEST IS LOADED.

IS (INPUT TEST STRING)

THIS COMMAND DIFFERS FROM THE INPUT TEST COMMAND IN THAT IF THE TEST IS OBJECT CODE, THAT TEST IS IMMEDIATELY EXECUTED AND THE NEXT TEST IS READ FROM TAPE. THIS CONTINUES UNTIL A TEST OF SOURCE CODE IS READ OR ALL TESTS HAVE BEEN EXECUTED (TAPE SUPPLY EXHAUSTED). WHEN A TEST OF SOURCE CODE IS READ CONTROL IS RETURNED TO THE CONSOLE AS FOR THE INPUT TEST COMMAND.

4.1.5 TIME OUT

TO,NNNNN

WHERE NNNNN IS A DECIMAL NUMBER THAT WILL VARY THE TIME DURATION OF A SUBSYSTEM TIMEOUT (SEE PARAGRAPH 4.4). THE VALUE HAS NO RELATIONSHIP TO TIME, IT IS SIMPLY THE NUMBER OF TIMES A SOFTWARE LOOP IS EXECUTED. IT IS PRESET TO 2000. EXPERMENTING WITH VALUES IS SUGGESTED AS THE BEST PROCEDURE TO FIND THE DESIRED VALUE FOR A GIVEN PROCESSOR AND MEMORY CONFIGURATION. THE TIMEOUT FALUE IS OUTPUTED WHEN A TEST IS PUNCHED, EITHER SOURCE OR OBJECT. WHEN THAT TEST IS READ, THE ASSOCIATED TIMEOUT VALUE IS STORED. NNNNN MUST BE 32767(10) OR LESS.

TO,?

WILL CAUSE THE TIME OUT VALUE TO BE PRINTED.

4.1.6 ITERATION COUNT

IC,NNNNN

WHERE NNNNN IS THE DECIMAL NUMBER OF TIMES THE NEXT TEST EXECUTED WILL BE ITERATED (32767(10) OR LESS).

IF THE TEST IS WRITTEN ONTO PAPER TAPE EITHER AS SOURCE OR OBJECT CODE, THE ITERATION COUNT IS ALSO WRITTEN. WHEN THE TEST IS LOADED FROM PAPER TAPE, THE WRITTEN ITERATION COUNT IS USED.

IC,?

WILL TYPE THE CURRENT ITERATION COUNT ON THE CONSOLE.

4.1.7 SPECIAL DATA PATTERNS

DP,BUFFER NAME,DDDD----D(RETURN)
DDDD----D(RETURN)
(RETURN)

WHERE PATTERN NAME IS X, Y, OR Z AND WHERE DDDD----D IS THE OCTAL DATA TO BE USED AS A DATA PATTERN.

THE TOTAL LENGTH OF D IS 32 WORDS OR LESS. TWO CONSECUTIVE CARRIAGE RETURNS TERMINATE DATA PATTERN ENTRY AND A SINGLE CARRIAGE RETURN RETURNS THE CARRIAGE BUT IS IGNORED AS FAR AS THE DATA PATTERN IS CONCERNED. IF LESS THAN 32 WORDS ARE ENTERED THE REMAINDER OF THE WORDS ARE ZERO FILLED.

EACH LINE MUST BE LIMITED TO 8 WORDS (48 ASCII CHARACTERS) PER LINE OR LESS AT WHICH TIME A CARRIAGE RETURN MUST BE TYPED. ALTHOUGH 6 ASCII CHARACTERS ARE REQUIRED TO FILL A SINGLE WORD, A CARRIAGE RETURN AT SOMETHING OTHER THAN MODULE 6 CAUSES LEFT JUSTIFYING OF THE PARTIAL WORD GIVEN AND USING IT AS A FULL WORD.

THE PATTERN NAME (X, Y, OR Z) MAY BE SPECIFIED IN ANY COMMAND INVOLVING PATTERN SELECTION TO SELECT THE SPECIAL PATTERN. IF THE NAMED SPECIAL PATTERN HAS NOT BEEN DEFINED PRIOR TO ITS USE, A PATTERN OF ALL ZEROS WILL BE SUPPLIED.

THE SPECIAL DATA PATTERNS THAT HAVE BEEN DEFINED WILL BE PUNCHED WITH THE TEST. THESE DATA PATTERNS ARE RETRIEVED WHEN THE TEST IS LOADED.

4.1.8 EDIT BUFFER

EB,BUFFER NAME,WORD POSITION,DDDDD----D(RETURN)
DDDDD----D(RETURN)
(RETURN)

WHERE BUFFER NAME IS X, Y, OR Z; WHERE WORD POSITION IS THE FIRST WORD THAT IS TO BE EDITED; AND WHERE DDDD---D IS THE DATA TO BE ENTERED. WORD POSITION IS AN OCTAL NUMBER AND THE FIRST WORD IN THE BUFFER IS SPECIFIED AS 0.

THE ENTRY PROCEDURE IS THE SAME AS FOR THE DP COMMAND. THE SIGNIFICANT DIFFERENCE IS THAT THE BUFFER IS NOT CLEARED AND ANY WORDS NOT CHANGED BY THE EB COMMAND ARE UNCHANGED. AS BEFORE, PARTIAL WORDS ARE LEFT JUSTIFIED AND ZERO FILLED.

4.1.9 BUFFER DUMP

BD,BUFFER NAME, NUMBER OF WORDS

WHERE BUFFER NAME CAN BE SPECIAL BUFFER X, Y, OR Z, THE READ (R) OR WRITE (W) BUFFER, OR THE HEADER (H) BUFFER (SEE SUBSYSTEM COMMAND READ ALL HEADERS). THE PRINTOUT STARTS WITH WORD 0 AND PRINTS THE NUMBER OF WORDS SPECIFIED (OCTAL).

4.1.10 COMPILE

CO,NC,BII

THIS COMMAND CAUSES THE STORED DEFERRED COMMANDS TO BE COMPILED INTO A TEST SEQUENCE. THE OPTIONAL PARAMETER (NC) SPECIFIES IF THE TEST IS TO BE RUN IN A CHECK OR NO-CHECK MODE. IF THE PARAMETER IS OMITTED THE TEST WILL BE COMPILED TO BE RUN WITH NORMAL CHECKING. IF THE PARAMETER IS GIVEN THE TEST WILL BE COMPILED FOR NO-CHECK EXECUTION.

NO-CHECK PERTAINS ONLY TO ALL SUBSYSTEM COMMANDS (SEE DESCRIPTION OF THE SUBSYSTEM FUNCTION COMMAND AND THE COMMANDS LISTED IN APPENDIX B.2) WITH THE EXCEPTION OF THE READ ALL HEADER. THIS COMMAND CANNOT BE EXECUTED IN NO-CHECK MODE.

THE OPTIONAL PARAMETER (BII) SPECIFIES THAT ALL DATA TRANSFER OPERATIONS IN THE TEST ARE TO BE EXECUTED WITH "BUS ADDRESS INCREMENT INHIBIT". SPECIFYING BII CAUSES THE PROGRAM TO SUSPEND THE INTERNAL PROGRAM CHECK THAT LOOKS FOR WORD COUNTS THAT ARE GREATER THAN THE BUFFER SIZE, I.E., THE PROGRAM WILL ACCEPT A DATA TRANSFER WORD COUNT OF ANY SIZE.

IT SHOULD BE NOTED THAT AFTER THE COMPILE THE SOURCE CODE IS

```
597                                 STILL VALID AND CAN BE EDITED OR SAVED.
```

4.1.11 RUN

RU

THIS COMMAND CAUSES THE OBJECT CODE TO BE EXECUTED. A RUN COMMAND GIVEN BEFORE A TEST IS COMPILED IS REJECTED WITH AN ERROR MESSAGE. EXECUTING A RUN COMMAND CAUSES THE SOURCE CODE TO BE LOST.

4.1.12 EDIT ADD LINE

EA,LN,NEW COMMAND

WHERE LN IS THE NUMBER (DECIMAL) THAT SPECIFIES THE POSITION OF THE LINE INSERTION AND WHERE NEW COMMAND IS THE COMMAND TO BE INSERTED INTO THE SOURCE.

AFTER THE COMMAND IS EXECUTED THE NEW COMMAND WILL HAVE THE LINE NUMBER LN AND THE LINE NUMBERS FROM THE ENTRY POINT (INCLUDING THE LINE THAT WAS AT THE ENTRY POINT) TO THE END OF THE TEST WILL BE INCREMENTED.

4.1.13 EDIT DELETE LINE

ED,LN

WHERE LN IS THE NUMBER IN DECIMAL OF THE LINE TO BE DELETED.

4.1.14 PRINT TEST

PT

THIS COMMAND WILL CAUSE THE STORED SOURCE TO BE PRINTED ON THE TELETYPE. LINE NUMBERS (DECIMAL) WILL BE PRINTED AT THE BEGINNING OF EACH LINE.

4.1.15 PRINT LINE

PL,LN

WHERE LN IS THE DECIMAL NUMBER OF THE LINE THAT IS TO BE PRINTED.

4.1.16 NEW TEST

NT

THIS COMMAND CAUSES THE PROGRAM TO INITIALIZE ITSELF BY CLEARING THE SOURCE COMMANDS, CLEARING THE COMPILED TEST, AND TERMINATING ANY TEST PRESENTLY EXECUTING. ALL STORED PARAMETERS (DRIVE NUMBER, ITERATION COUNT, STALL, CYLINDER NUMBER, TRACK, SECTOR, ETC.) ARE NOT CHANGED.

4.1.17 PRINT REGISTER

PR,REGISTER SELECT

WHERE REGISTER SELECT IS THE UNIBUS ADDRESSABLE REGISTER, SPECIFIED AS AN OCTAL NUMBER OR A NMEMONIC NAME. THE POSSIBILITIES ARE:

NUMBER	NAME	DESCRIPTION
00	CS1	COMMAND STATUS REG 1
01	WC	WORD COUNT
02	BA	BUS ADDRESS
03	DA	DESIRED ADDRESS - TRACK & SECTOR
04	CS2	COMMAND STATUS REG 2
05	DS	DRIVE STATUS
06	ER	ERROR REGISTER
07	ASOF	ATTENTION SUMMARY & OFFSET
10	DC	DESIRED CYLINDER
11	UNUSED	
12	DB	DATA BUFFER
13	MR1	MAINTENACE REGISTER 1
14	MR2	MAINTENANCE REGISTER 2
15	MR3	MAINTENANCE REGISTER 3
16	POS	ECC/POSITION
17	PAT	ECC/PATTERN

4.1.18 HELP

HP

THIS COMMAND WILL CAUSE A SUMMARY OF THE INTERACTIVE COMMANDS TO BE PRINTED. IT IS VALID ONLY AFTER THE PROGRAM IS FIRST LOADED.

4.1.19 FORMAT SELECT

```
706                                     FT,NN
```

WHERE NN SPECIFIES THE FORMAT AS AN OCTAL NUMBER (24 FOR 20 SECTOR/TRACK AND 26 FOR 22 SECTOR/TRACK). THE FORMAT DEFAULTS TO 26. THE SELECTED FORMAT APPLIES TO ALL COMMANDS AND WILL REMAIN AS SET UNTIL CHANGED. THIS VALUE IS ALSO OUTPUTTED TO PAPER TAPE AS PART OF THE TEST. CONSEQUENTLY, INPUTTING A TEST CAN ALSO CHANGE THE FORMAT SELECTED.

FT,?

WILL CAUSE THE FORMAT VALUE TO BE PRINTED.

4.2 DEFERRED COMMAND SET

4.2.1 SUBSYSTEM FUNCTION COMMAND

SF,SUBSYSTEM COMMAND,DRIVE NUMBER,CCC,T,SS,
NUMBER OF WORDS,DATA PATTERN

WHERE ALL NUMERIC VALUES ARE OCTAL AND:

* SF IS SUBSYSTEM FUNCTION COMMAND.
* SUBSYSTEM COMMAND IS ONE OF THE FOLLOWING:

RD	READ DATA
WD	WRITE DATA
WC	WRITE CHECK
WH	WRITE HEADER
RH	READ HEADER
SK	SEEK
CC	CONTROLLER CLEAR
CS	CLEAR SUBSYSTEM
DC	DRIVE CLEAR
RC	RECALIBRATE
DS	DRIVE SELECT
PA	PACK ACKNOWLEDGE
UL	UNLOAD
SS	START SPINDLE
OF	OFFSET
AH	READ ALL HEADER

* DRIVE NUMBER IS THE NUMBER OF THE DRIVE TO BE ADDRESSED. IF UNSPECIFIED THE LAST DRIVE ADDRESSED WILL BE USED.
* CCC IS THE CYLINDER ADDRESS OR THE OFFSET VALUE IF THE SUBSYSTEM COMMAND IS OFFSET.
* T IS THE TRACK ADDRESS
* SS IS THE SECTOR ADDRESS

```
   763
   764                                 *  NUMBER  OF  WORDS  IS  THE  NUMBER  OF   WORDS   TO   BE
```

TRANSFERRED. THE NUMBER IS CHECKED AGAINST THE MAXIMUM ALLOWED CONSISTANT WITH THE BUFFER SIZE. IF THE NUMBER IS GREATER THAN THE MAXIMUM AN ERROR IS REPORTED WHEN THE TEST IS COMPILED UNLESS THE COMPILE COMMAND IS GIVEN WITH THE BII PARAMETER (SEE COMPILE COMMAND DESCRIPTION).

* DATA PATTERN IS AN ALPHABETIC CHARACTER TO SELECT THE DESIRED DATA PATTERN FROM THE VARIOUS PATTERNS AVAILABLE (SEE APPENDIX A). THE CHARACTERS X, Y, OR Z ARE VALID TO USE A USER DEFINED PATTERN. IF THIS PATTERN HAS NOT BEEN DEFINED, DATA OF ALL ZEROS WILL BE USED.

ANY PARAMETER MAY BE OMITTED AND ALLOWED TO DEFAULT TO THE LAST VALUE GIVEN FOR THAT PARAMETER. A CARRIAGE RETURN ANYWHERE IN THE COMMAND ALLOWS THE REMAINING PARAMETERS TO DEFAULT. SEPARATORS (,) MUST BE SUPPLIED IF A PARAMETER IS OMITTED AND FOLLOWING PARAMETERS ARE GIVEN.

OMITTING THE DRIVE NUMBER PARAMETER IS USEFUL TO HAVE A GENERAL PURPOSE TEST THAT CAN BE RUN ON ANY DRIVE ADDRESS. THE IMMEDIATE COMMAND DN CAN BE USED TO SPECIFY THE DESIRED DRIVE BEFORE THE GENERAL PURPOSE TEST IS EXECUTED. NOTE THAT THIS IS NOT APPLICABLE IF MORE THAN ONE DRIVE IS TO BE USED IN THE GENERAL PURPOSE TEST.

OMITTING THE DATA PATTERN PARAMETER WILL CAUSE THIS COMMAND TO USE THE BUFFER AS IT WAS LAST INITIALIZED. IF THE PARAMETER IS GIVEN THE OUTPUT BUFFER IS INITIALIZED AS PART OF THE TEST. THIS IS ESPECIALLY IMPORTANT WHEN EXECUTION SPEED SHOULD BE FAST FOR SCOPING PURPOSES. THE BUFFER INITIALIZE COMMAND CAN BE ENTERED AND EXECUTED AS A SEPARATE TEST TO AVOID BUFFER LOADING IN A TEST WHERE SPEED IS REQUIRED.

THE NO-CHECK CAPABILITY OF THE COMPILE COMMAND APPLIES TO THE SUBSYSTEM COMMANDS THAT ARE LISTED ABOVE (WITH THE EXCEPTION OF THE READ SPECIFIC HEADER). WHEN THE NO-CHECK MODE OF OPERATION IS INVOKED THE CHECKING FUNCTIONS THAT DETECT THE OCCURRANCE OF OPERATION ERRORS OR FAILURES IN THE CONTROLLER OR DRIVE ARE INHIBITED. THE SUBSYSTEM COMMANDS ARE EXECUTED REGARDLESS OF ERROR CONDITIONS AT THE START OF THE COMMAND OR ERROR OCCURRANCE DURING THE COMMAND. THE ONLY REQUIREMENT FOR THE TEST TO PROCEED TO THE NEXT COMMAND IS THE CONTROLLER MUST INDICATE COMMAND COMPLETION BY SETTING "READY". THE IMPLICATION IS THAT WHEN THE NO-CHECK MODE IS USED THE TEST PROGRAM IS RESPONSIBLE FOR TESTING FOR ERRORS AND CLEARING ERROR CONDITIONS.

THE WRITE HEADER COMMAND IS IMPLEMENTED SUCH THAT THE CYLINDER AND TRACK PARAMETERS SPECIFY THE PHYSICAL LOCATION (CYLINDER & TRACK) THAT IS TO BE FORMATTED. THE SECTOR PARAMETER, IF SPECIFIED AS ZERO, CAUSED THE CORRECT HEADER FOR THAT PHYSICAL LOCATION TO BE GENERATED IN THE OUTPUT BUFFER AND WRITTEN. IF THE SECTOR PARAMETER IS NON-ZERO THE CONTENTS OF THE

```
821                                 SPECIAL DATA PATTERN BUFFER X, Y, AND Z ARE USED AND  WRITTEN  AS
822                                 THE HEADERS ON THAT CYLINDER AND TRACK.  THE WRITE HEADER COMMAND
```

DOES NOT ALTER THE CONTENTS OF BUFFER X, Y, OR Z. THE CONTENTS OF THESE BUFFERS MUST BE SPECIFIED USING THE SPECIAL DATA PATTERN (DP) COMMAND TO LOAD ALL OF BUFFER X, ALL OF BUFFER Y, AND THE FIRST 2 WORDS OF BUFFER Z (66 WORDS REQUIRED FOR HEADERS).

4.2.2 BUFFER INITIALIZE

BI,ALPHABETIC CHARACTER

WHERE THE ALPHABETIC CHARACTER SPECIFIES THE DATA PATTERN TO BE USED (SEE APPENDIX A FOR THE AVAILABLE PATTERNS). THE OUTPUT BUFFER WILL BE INITIALIZED TO THE PATTERN SELECTED. CHARACTERS X, Y, OR Z ARE VALID TO SELECT THE USER DEFINED DATA PATTERN. IF THE SPECIAL DATA PATTERN HAS NOT BEEN USER DEFINED BEFORE IT IS SELECTED A PATTERN OF ALL ZEROS WILL BE USED.

4.2.3 DATA COMPARE

DC,NNNNNN

WHERE NNNNNN IS A OCTAL VALUE SPECIFING THE NUMBER OF WORDS TO BE COMPARED STARTING AT THE BEGINNING OF THE OUTPUT AND INPUT BUFFERS. IF NNNNNN IS OMITTED THE NUMBER OF WORDS IN THE LAST INPUT DATA TRANSFER ARE COMPARED.

A DATA MISCOMPARE WILL CAUSE THE GOOD AND BAD DATA TO BE REPORTED IN THE ERROR REPORT.

4.2.4 STATUS COMPARE

SC,STATUS WORD NUMBER,EXPECTED VALUE,MASK

WHERE ENTERED VALUES ARE OCTAL AND:

* STATUS WORD NUMBER IS THE DRIVE STATUS WORD TO BE COMPARED. STATUS WORDS ARE DESIGNATED 0 THROUGH 7 AND ARE ARBITRARILY ASSIGNED AS FOLLOWS:

  00 IS MESSAGE LINE A WORD 0
  01 IS MESSAGE LINE B WORD 0
  02 IS MESSAGE LINE A WORD 1
  03 IS MESSAGE LINE B WORD 1
  04 IS MESSAGE LINE A WORD 2
  05 IS MESSAGE LINE B WORD 2
  06 IS MESSAGE LINE A WORD 3
  07 IS MESSAGE LINE B WORD 3

```
                                        *  EXPECTED VALUE IS THE VALUE THE STATUS SPECIFIED  SHOULD
                                           BE.
```

* MASK SPECIFIES WHICH BITS ARE TO BE COMPARED IN THE STATUS WORD READ AND THE EXPECTED VALUE. A ONE IN A SPECIFIC BIT POSITION ALLOWS THAT BIT POSITION COMPARISON TO OCCUR AND A ZERO INHIBITS COMPARISON. MASK IS INITIALLY SET TO 077777 WHEN THE PROGRAM IS LOADED. (BIT IS OF DRIVE STATUS WORDS IS ALWAYS ZERO.) ONCE MASK HAS BEEN SPECIFIED, THAT VALUE IS STORED AND USED FOR SUBSEQUENT SC COMMANDS UNTILL IT IS CHANGED BY A SUBSEQUENT SC COMMAND THAT INCLUDES MASK SPECIFICATION.

A STATUS MISCOMPARE WILL CAUSE THE GOOD AND BAD STATUS WORDS TO BE REPORTED.

4.2.5 REGISTER COMPARE

RC.REGISTER NUMBER,EXPECTED VALUE,MASK

WHERE ENTERED VALUES ARE OCTAL AND:

* REGISTER NUMBER IS THE NUMBER OF THE UNIBUS ADDRESSABLE REGISTER OF THE RK611 TO BE COMPARED.

* EXPECTED VALUE IS THE VALUE THE SPECIFIED REGISTER SHOULD CONTAIN.

* MASK SPECIFIES WHICH BITS ARE TO BE COMPARED IN THE PEGISTER READ AND THE EXPECTED VALUE. A ONE IN A SPECIFIC BIT POSITION ALLOWS THAT BIT POSITION COMPARISON TO OCCUR AND A ZERO INHIBITS THAT COMPARISON. MASK IS INITIALLY SET TO 177777 WHEN THE PROGRAM IS LOADED. ONCE THE MASK HAS BEEN SPECIFIED, THAT VALUE IS STORED AND USED IN SUBSEQUENT RC COMMANDS UNTIL THE MASK IS CHANGED BY A SUBSEQUENT RC COMMAND THAT INCLUDES A MASK SPECIFICATION.

REGISTER MISCOMPARE WILL CAUSE GOOD AND BAD VALUES TO BE REPORTED.

4.2.6 REGISTER WRITE

RW,REGISTER SELECT,VALUE

WHERE ENTERED VALUES ARE OCTAL AND:

* REGISTER SELECT IS THE NUMBER OR THE MNEMONIC NAME OF THE UNIBUS ADDRESSABLE REGISTER OF THE RK611 TO BE WRITTEN. (SEE PARAGRAPH 4.1.14 FOR LIST OF NUMBERS AND NAMES.)

```
937
938                                     *  VALUE IS THE VALUE TO BE LOADED.
```

ANY REGISTER AND ANY VALUE MAY BE SPECIFIED. NO CHECK IS MADE FOR READ ONLY BITS.

4.2.7 STALL

ST,NNNNN

WHERE NNNNN IS A DECIMAL NUMBER SPECIFING A CONSTANT TO DELAY BETWEEN SUBSYSTEM FUNCTION COMMANDS. NNNNN MUST BE 32767(10) OR LESS.

4.2.8 PRINT MESSAGE

PM,MESSAGE

WHERE MESSAGE CAN BE ANY ASCII STRING UP TO 70 CHARACTERS IN LENGTH. THAT MESSAGE IS PRINTED ON THE CONSOLE TERMINAL WHEN THE PM COMMAND IS EXECUTED. THE MESSAGE IS PRINTED ONLY DURING THE FIRST EXECUTION OF THE TEST AFTER A RUN COMMAND IS EXECUTED AND SUPPRESSED DURING SUBSEQUENT TEST ITERATIONS IF THE ITERATION COUNT IS GREATER THAN 1, IF LOOP ON TEST (SW14 SET) OR LOOP ON ERROR (SW9 SET AND ERROR).

4.2.9 UNIBUS INITIALIZE

UI

THIS COMMAND WILL CAUSE A RESET TO BE EXECUTED TO CLEAR ALL UNITS CONNECTED TO THE UNIBUS.

4.3 LINE NUMBERING

AS DEFERRED COMMANDS ARE ENTERED AND STORED, THE PROGRAM ASSIGNS DECIMAL LINE NUMBERS TO THE COMMANDS SEQUENTIALLY. THE LINE NUMBERS ARE USED IN THE EDIT COMMANDS (EA AND ED) AND FOR LINE PRINTING (PL).

WHEN A LINE IS ADDED OR DELETED FROM THE SOURCE, THE STORED LINES ARE RENUMBERED IMMEDIATELY. SUBSEQUENT LINE ORIENTED COMMANDS MUST TAKE THE NEW LINE NUMBERS INTO CONSIDERATION.

4.4 TIMEOUT

TO PREVENT "SILENT DEATH" SITUATIONS (THE PROGRAM STARTS AN OPERATION ON THE RK06 SUBSYSTEM AND THE SUBSYSTEM NEVER SIGNALS COMPLETION) A SOFTWARE TIMER IS EMPLOYED. EACH TIME THE PROGRAM STARTS AN OPERATION ON THE RK06 SUBSYSTEM, THE TIMER IS USED TO INSURE THAT THE REQUESTED ACTIVITY COMPLETES WITHIN A REASONABLE PERIOD OF TIME. IF THE ACTIVITY DOES NOT COMPLETE, THE PROGRAM WILL DISPLAY A SUBSYSTEM TIMEOUT MESSAGE.

THE REASONABLE PERIOD OF TIME JUST MENTIONED IS NOT CALIBRATED TO REAL TIME. CALIBRATION IS NOT POSSIBLE BECAUSE OF VARIOUS CONFIGURATIONS OF MEMORIES AND PROCESSORS.

TO ENHANCE THE USEFULNESS OF THIS TIMEOUT FEATURE, THE TIMER IS VARIABLE. (SEE TIMEOUT COMMAND DESCRIPTION). THE DEFAULT VALUE OF THE VARIABLE IS LARGE ENOUGH TO INSURE COMMAND COMPLETION.

4.5 TEST LOOPING AND LOOP COUNTERS

USING THE SWITCH OPTIONS PROVIDED, THE PROGRAM WILL LOOP ON THE TEST (SWITCH 14) OR LOOP ON ERROR (SWITCH 9). WHENEVER A TEST IS BEING LOOPED, EACH LOOP IS COUNTED.

THE LOOP COUNT IS REPORTED IN TWO INSTANCES. THESE ARE WHEN AN ERROR OCCURS AND IS REPORTED (SWITCH 13 RESET) AND WHEN LOOPING IS TERMINATED.

5.0 ERROR REPORTING FORMATS

TWO BASIC REPORT FORMATS ARE DEFINED. FORMAT 1 IS FOR ALL ERRORS (EITHER PROGRAM OR HARDWARE DETECTED) WHERE COMMAND PARAMETERS AND RK611 REGISTER CONTENTS ARE APPLICABLE. FORMAT 2 IS FOR COMPARISON ERROR REPORTING, I.E., STATUS COMPARE, REGISTER COMPARE, AND DATA COMPARE.

5.1 FORMAT 1

FORMAT 1 HAS THE FOLLOWING ENTRIES:

ERROR MESSAGE

XXX = CMND LINE NUM

DRIVE=

CMND=

```
1049
1050                                    CURRENT OPERATIONS:
```

PARAMETERS GIVEN:

CYLNDR SECTOR TRACK OFFSET BAH BAL WDC

APPLICABLE REGISTERS:

CS1 CS2 WC BA DA DC ASOF

ER DS AC BC

A1 B1 A2 B2 A3 B3 ECC/POS ECC/PAT

PREVIOUS OPERATION:

DRIVE=

CMND=

PARAMETERS GIVEN:

CYLNDER SECTOR TRACK OFFSET BAH BAL WDC

XXXXXXXXXX = NUMBER OF LOOPS

ALL THE ENTRIES LISTED ABOVE WILL NOT APPEAR IN EVERY REPORT. ENTRIES THAT ARE NOT PERTINENT TO THE OPERATION ARE OMITTED. FOR EXAMPLE, THE PARAMETERS GIVEN ENTRIES ARE NOT APPLICABLE TO A PACK ACKNOWLEDGE OPERATION SO ALL THESE ENTRIES ARE OMITTED IF PA IS THE FAILING COMMAND.

THE NUMBER OF LOOPS ENTRY IS PRINTED ONLY IF THE TEST IS RUNNING WITH LOOP ON ERROR OR LOOP ON TEST SET. WITH THE EXCEPTION OF THE ERROR MESSAGE ENTRY, THE ENTRIES LISTED ABOVE ARE SELF EXPLANATORY. ALL ERROR MESSAGES ARE LISTED BELOW.

5.1.1 SUBSYSTEM DETECTED ERROR

THIS MESSAGE IS PRINTED WHENEVER THE PROGRAM IS ALERTED THAT THE SUBSYSTEM HAS DETECTED AN ERROR. THIS INCLUDES ALL THE ERRORS DETECTED IN THE CONTROLLER OR DRIVE.

5.1.2 UNSOLICITED ATTENTION

THIS MESSAGE INDICATES AN INTERRUPT WAS RECEIVED FROM A DRIVE BUT NO OPERATION HAS BEEN STARTED ON THAT DRIVE. THIS MESSAGE WILL BE SEEN IF WRITE LOCK IS CHANGED OR A DRIVE IS STARTED MANUALLY. THE MESSAGE IS NOT PRINTED WHEN THE CHANGE OCCURS IF THE PROGRAM IS AT COMMAND LEVEL (INTERRUPTS ARE LOCKED OUT) BIT IS PRINTED AS SOON AS CARRIAGE RETURN IS TYPED ON THE CONSOLE.

5.1.3 UNEXPECTED DATA TYPE ERROR

THIS MESSAGE INDICATES AN INTERRUPT OCCURRED THAT WAS CAUSED BY DATA ERROR TYPE (DCK, OPI, HUNC, OR WCE) WHEN NO COMMAND OR A NON-DATA TRANSFER COMMAND WAS BEING EXECUTED.

5.1.4 ATTENTION DID NOT RESET WITH DRIVE CLEAR

THIS MESSAGE INDICATES A DRIVE CLEAR COMMAND WAS NOT ABLE TO RESET THE DRIVE ATTENTION SIGNAL. THIS IS A CATASTROPHIC ERROR FOR THE PROGRAM. THE HIGH ATTENTION SIGNAL WILL CAUSE CONTINUOUS INTERRUPTS.

5.1.5 ATTENTION DID NOT RESET WITH SUBSYSTEM CLEAR

THIS MESSAGE INDICATES AN ERROR OF THE SAME TYPE AS "ATTENTION DID NO RESET WITH DRIVE CLEAR". THE DIFFERENCE IS THAT THE SUBSYSTEM CLEAR GENERATES RESET TO ALL DRIVES.

5.1.6 ILLEGAL DRIVER COMMAND

THIS MESSAGE IS AN INDICATION OF AN INTERNAL PROGRAM INTERLAU PROBLEM. IT SHOULD NEVER APPEAR. IF IT DOES, PLEASE NOTIFY DIAGNOSTIC ENGINEERING.

5.1.7 SUBSYSTEM TIMEOUT

THIS MESSAGE INDICATES THAT THE SUBSYSTEM FAILED TO SEND AN INTERRUPT WITHIN A REASONABLE PERIOD OF TIME. "REASONABLE" IS SUFFICIENTLY LONG SO THAT THE INTERRUPT SHOULD HAVE OCCURRED.

5.1.8 CLEAR CONTROLLER DID NOT CLEAR ERROR

THIS MESSAGE INDICATES THAT THE CONTROLLER ERROR WAS NOT RESET WHEN A CONTROLLER CLEAR WAS DONE. THIS HAS THE SAME IMPLICATION AS "DRIVE HARD ERROR" MESSAGE BUT AT THE CONTROLLER LEVEL.

5.1.9 NO ATTENTION IN ATTENTION SUMMARY REGISTER

THIS MESSAGE INDICATES AN INTERRUPT WAS RECEIVED FROM THE CONTROLLER BUT NO DRIVE HAS RAISED ATTENTION.

5.1.10 DATA LATE WHEN UNLOADING HEADER

THIS MESSAGE IS PERTINENT TO THE READ ALL HEADER COMMAND AND INDICATES PROBLEM REACHING THE DATA BUFFER.

5.1.11 CONTROLLER ERROR WHILE DRIVER SERVICING

THIS MESSAGE INDICATES A CONTROLLER ERROR OCCURRED WHILE THE PROGRAM WAS DOING SERVICE TYPE OPERATIONS. THESE SERVICE OPERATIONS ARE OF THE "DRIVE SELECT" OR "DRIVE CLEAR" NATURE AND ARE PERFORMED WHEN THE PROGRAM IS GATHERING STATUS FOR REPORTING, CLEARING ERRORS, ETC.

5.1.12 DRIVE PARITY WHILE GATHERING STATUS

THIS MESSAGE INDICATES THAT THE DRIVE HAS DETECTED A SERCON PARITY ERROR WHILE THE PROGRAM WAS DOING THE SERVICE OPERATION DESCRIBED ABOVE.

5.1.13 MULTIPLE DRIVE SELECT

THIS MESSAGE IS SELF-EXPLANATORY AND WILL APPEAR WITH A SUBSYSTEM DETECTED ERROR MESSAGE.

5.2 FORMAT 2

THREE ERRORS ARE REPORTED USING FORMAT 2. THESE ARE REGISTER COMPARE ERROR, STATUS COMPARE ERROR, AND DATA COMPARE ERROR.

THE REGISTER AND STATUS COMPARE ERROR REPORT FORMAT IS:

XXX=CMND LINE NUM

ERROR MESSAGE (STATUS OR REGISTER COMPARE ERROR)

NN=REGISTER NUMBER OF STATUS WORD NUMBER

(VALUE EXPECTED)=GOOD DATA

(VALUE RECEIVED)=BAD DATA

(SPECIFIED MASK)=NUMBER OF LOOPS

THE DATA COMPARE ERROR REPORT FORMAT IS:

1219

XXX=CMND LINE NUMBER

```
1220
1221
1222
1223                                    DATA COMPARE ERR ON WORD NNNN
1224                                    (DATA)=GOOD DATA
1225                                    (DATA)=BAD DATA
1226                                    XXXX=TOTAL MISCOMPARES
1227
1228                                    XXXXXXXXXX=NUMBER OF LOOPS
```

1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258

APPENDIX A

THE FOLLOWING DATA PATTERNS HAVE BEEN DEFINED. ADDITIONAL PATTERNS WILL BE INCLUDED WHEN THEY BECOME KNOWN.

PATTERN "A"	PATTERN "B"	PATTERN "C"	PATTERN "D"
177777	000000	125252	052525
000000	177777	125252	052525
.	.	.	.
.	.	.	.
.	.	.	.
(32 WORDS)	(32 WORDS)	(32 WORDS)	(32 WORDS)
.	.	.	.
.	.	.	.
000000	177777	125252	052525
177777	000000	125252	052525

PATTERN "E"	PATTERN "F"	PATTERN "G"
000001	177777	177776
000003	177776	177775
000007	177774	177773
000017	177770	177767
000037	177760	177757
000077	177740	177737
000177	177700	177677
000377	177600	177577
000777	177400	177377
001777	177000	176777
003777	176000	175777
007777	174000	173777
017777	170000	167777
037777	160000	157777
077777	140000	137777
177777	100000	077777
077777	000000	137777
037777	100000	157777
017777	140000	167777
007777	160000	173777
003777	170000	175777
001777	174000	176777
000777	176000	177377
000377	177000	177577
000177	177400	177677
000077	177600	177737
000037	177700	177757
000017	177740	177767
000007	177760	177773
000003	177770	177775
000001	177774	177776
000000	177776	177777

```
1297
1298
1299
1300                                        PATTERN "H"     PATTERN "I"
1301                                        -----------     -----------
1302
1303                                          000001          155555
1304                                          000002          155555
1305                                          000004            .
1306                                          000010            .
1307                                          000020            .
1308                                          000040            .
1309                                          000100          (32 WORDS)
1310                                          000200            .
1311                                          000400            .
1312                                          001000            .
1313                                          002000          155555
1314                                          004000          155555
1315                                          010000
1316                                          020000
1317                                          040000
1318                                          100000
1319                                          100000
1320                                          040000
1321                                          020000
1322                                          010000
1323                                          004000
1324                                          002000
1325                                          001000
1326                                          000400
1327                                          000200
1328                                          000100
1329                                          000040
1330                                          000020
1331                                          000010
1332                                          000004
1333                                          000002
1334                                          000001
```

APPENDIX B

COMMAND SUMMARIES

B.1 USER DEFINED COMMAND SET

B.1.1 IMMEDIATE COMMANDS

ALL DECIMAL VALUES MUST BE LESS THAN 32767(10).

COMMANDS	MNEMONIC	PARAMETERS
DRIVE SELECTION	DN	,DRIVE NUMBER
	DN	,?
OUTPUT TEST	OT	,0
		,S
INPUT TEST	IT	NONE
INPUT STRING	IS	NONE
ITERATION COUNT	IC	,NNNNN
	IC	,?
SPECIAL DATA BUFFER	DP	,PATTERN NAME
		,DDDD....D
COMPILE	CO	,NO CHECK
		,INCREMENT INHIBIT
EDIT ADD LINE	EA	,LN
		,NEW COMMAND
EDIT DELETE LINE	ED	,LN
EDIT BUFFER	EB	,BUFFER NAME
		,WORD POSITION
		,DDDD--D
BUFFER DUMP	BD	,BUFFER NAME
		,NUMBER OF WORDS
PRINT TEST	PT	NONE
PRINT LINE	PL	,LN

1391 NEW TEST NT NONE

```
1392
1393
1394
1395                    RUN                     RU      NONE
1396
1397                    PRINT REGISTER          PR      ,REGISTER NUMBER
1398
1399                    HELP                    HP      NONE
1400
1401                    TIME OUT CHANGE         TO      ,CONSTANT
1402                                            TO      ,?
1403
1404                    COPY TAPE               CT      ,NONE
1405
1406                    FORMAT SELECT           FT      ,FORMAT
1407                                                    ,?
```

B.1.2 DEFERRED COMMANDS

ALL DECIMAL VALUES MUST BE LESS THAN 32767(10).

COMMANDS	MNEMONIC	PARAMETERS
SUBSYSTEM FUNCTION	SF	,SUBSYSTEM COMMAND ,DRIVE NUMBER ,CCC ,T ,SS ,NUMBER OF WORDS ,DATA PATTERN
BUFFER INITIALIZE	BI	,PATTERN SELECT
DATA COMPARE	DC DC	NONE ,NNNNN
STATUS COMPARE	SC	,STATUS WORD SELECT ,EXPECTED VALUE ,MASK
REGISTER COMPARE NUMBER	RC	,REGISTER NAME OR ,EXPECTED VALUE ,MASK
REGISTER WRITE NUMBER	RW	,REGISTER NAME OR ,VALUE
STALL	ST	,NNNNN
PRINT MESSAGE	PM	,MESSAGE
UNIBUS INITIALIZE	UI	NONE

```
1451
1452
1453
1454                                    B.2  SUBSYSTEM COMMANDS
1455
1456
1457                                              COMMAND                 MNEMONIC
1458                                              -------                 --------
1459
1460                                            READ DATA                   RD
1461                                            WRITE DATA                  WD
1462                                            WRITE CHECK                 WC
1463                                            WRITE HEADER & DATA         WH
1464                                            READ HEADER                 RH
1465                                            SEEK                        SK
1466                                            CLEAR SUBSYSTEM             CS
1467                                            CONTROLLER CLEAR            CC
1468                                            DRIVE CLEAR                 DC
1469                                            RECALIBRATE                 RC
1470                                            DRIVE SELECT                DS
1471                                            PACK ACKNOWLEDGE            PA
1472                                            UNLOAD                      UL
1473                                            START SPINDLE               SS
1474                                            OFFSET                      OF
1475                                            READ ALL HEADER             AH
1476                                        @
1477                                        .NLIST  MC,MD,CND
1478                                        .LIST   ME
1479                                        .ENABL  ABS,AMA
1480                                        ;DEFINE SYSMAC MACROS
1481                                        .MCALL  .HEADER,.SWRHI,.SWRLO,.EQUAT,.SETUP,.$CATCH
1482                                        .MCALL  .$CMTAG,.$TYPE,.$TYPOCT,.$POWER,.$READ,.$TRAP
1483                                        .MCALL  .$TYPDEC,.$SAVE,.$DB2D
1484                                        .MCALL  .$RAND,.$SIZE,SETTRAP,SETPRI,GETPRI
1485    167000                              $SW.=   167000          ;DEFINE SWITCHES
1486                                    .TITLE  RK611/RK06 USER DEFINED TEST
1487                                    ;*COPYRIGHT (C) 1976
1488                                    ;*DIGITAL EQUIPMENT CORP.
1489                                    ;*MAYNARD, MASS. 01754
1490                                    ;*
1491                                    ;*PROGRAM BY MARV TEGROTENHUIS
1492                                    ;*
1493                                    ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
1494                                    ;*PACKAGE (MAINDEC-11-DZQAC-CO),MAR 21, 1976.
1495                                    ;*
1496    000001                          $TN=1
1497                                    .SBTTL  OPERATIONAL SWITCH SETTINGS
1498                                    ;*
1499                                    ;*      SWITCH                  USE
1500                                    ;*      ------          --------------------
1501                                    ;*        15            HALT ON ERROR
1502                                    ;*        14            LOOP ON TEST
1503                                    ;*        13            INHIBIT ERROR TYPEOUTS
1504                                    ;*        11            INHIBIT ITERATIONS
1505                                    ;*        10            BELL ON ERROR
1506                                    ;*         9            LOOP ON ERROR
```

```
                                ;*          2           INHIBIT MISCOMPARE PRINTING
                                ;*          1           REPORT ALL DATA MISCOMPARES
                                ;*          0           SHORT REPORT FORMAT

                                .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=   200             ;;CODE FOR CARRIAGE RETURN-LINE FEED
177776                          PS=     177776          ;;PROCESSOR STATUS WORD
                                .EQUIV  PS,PSW
177774                          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
```

```
1563            000040                  SW05=   40
1564            000020                  SW04=   20
1565            000010                  SW03=   10
1566            000004                  SW02=   4
1567            000002                  SW01=   2
1568            000001                  SW00=   1
1569                                    .EQUIV  SW09,SW9
1570                                    .EQUIV  SW08,SW8
1571                                    .EQUIV  SW07,SW7
1572                                    .EQUIV  SW06,SW6
1573                                    .EQUIV  SW05,SW5
1574                                    .EQUIV  SW04,SW4
1575                                    .EQUIV  SW03,SW3
1576                                    .EQUIV  SW02,SW2
1577                                    .EQUIV  SW01,SW1
1578                                    .EQUIV  SW00,SW0
1579
1580                                    ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
1581            100000                  BIT15=  100000
1582            040000                  BIT14=  40000
1583            020000                  BIT13=  20000
1584            010000                  BIT12=  10000
1585            004000                  BIT11=  4000
1586            002000                  BIT10=  2000
1587            001000                  BIT09=  1000
1588            000400                  BIT08=  400
1589            000200                  BIT07=  200
1590            000100                  BIT06=  100
1591            000040                  BIT05=  40
1592            000020                  BIT04=  20
1593            000010                  BIT03=  10
1594            000004                  BIT02=  4
1595            000002                  BIT01=  2
1596            000001                  BIT00=  1
1597                                    .EQUIV  BIT09,BIT9
1598                                    .EQUIV  BIT08,BIT8
1599                                    .EQUIV  BIT07,BIT7
1600                                    .EQUIV  BIT06,BIT6
1601                                    .EQUIV  BIT05,BIT5
1602                                    .EQUIV  BIT04,BIT4
1603                                    .EQUIV  BIT03,BIT3
1604                                    .EQUIV  BIT02,BIT2
1605                                    .EQUIV  BIT01,BIT1
1606                                    .EQUIV  BIT00,BIT0
1607
1608                                    ;*BASIC "CPU" TRAP VECTOR ADDRESSES
1609            000004                  ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
1610            000010                  RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
1611            000014                  TBITVEC=14              ;;"T" BIT
1612            000014                  TRTVEC= 14              ;;TRACE TRAP
1613            000014                  BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
1614            000020                  IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
1615            000024                  PWRVEC= 24              ;;POWER FAIL
1616            000030                  EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
1617            000034                  TRAPVEC=34              ;;"TRAP" TRAP
1618            000060                  TKVEC=  60              ;;TTY KEYBOARD VECTOR
```

```
1619            000064                  TPVEC=  64              ;;TTY PRINTER VECTOR
1620            000240                  PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
1621                                    .SBTTL  TRAP CATCHER
1622
1623            000000                          .=0
1624                                    ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
1625                                    ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
1626                                    ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
1627            000174                          .=174
1628  000174   000000                  DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
1629  000176   000000                  SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER
1630            000200                          .=200
1631  000200   000137  004116                  JMP     @#UDTSRT
1632
1633
```

```
                                .SBTTL  COMMON TAGS

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

                        001100                  .=1100
001100                          $CMTAG:                         ;;START OF COMMON TAGS
001100  000000                  $PASS:  .WORD   0               ;;CONTAINS PASS COUNT
001102     000                  $TSTNM: .BYTE   0               ;;CONTAINS THE TEST NUMBER
001103     000                  $ERFLG: .BYTE   0               ;;CONTAINS ERROR FLAG
001104  000000                  $ICNT:  .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
001106  000000                  $LPADR: .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
001110  000000                  $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
001112  000000                  $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
001114     000                  $ITEMB: .BYTE   0               ;;CONTAINS ITEM CONTROL BYTE
001115     001                  $ERMAX: .BYTE   1               ;;CONTAINS MAX. ERRORS PER TEST
001116  000000                  $ERRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001120  000000                  $GDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
001122  000000                  $BDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'BAD' DATA
001124  000000                  $GDDAT: .WORD   0               ;;CONTAINS 'GOOD' DATA
001126  000000                  $BDDAT: .WORD   0               ;;CONTAINS 'BAD' DATA
001130  000000                          .WORD   0               ;;RESERVED--NOT TO BE USED
001132  000000                          .WORD   0
001134     000                  $AUTOB: .BYTE   0               ;;AUTOMATIC MODE INDICATOR
001135     000                  $INTAG: .BYTE   0               ;;INTERRUPT MODE INDICATOR
001136  000000                          .WORD   0
001140  177570                  SWR:    .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
001142  177570                  DISPLAY: .WORD  DDISP           ;;ADDRESS OF DISPLAY REGISTER
001144  177560                  $TKS:   177560                  ;;TTY KBD STATUS
001146  177562                  $TKB:   177562                  ;;TTY KBD BUFFER
001150  177564                  $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
001152  177566                  $TPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
001154     000                  $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
001155     002                  $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001156     012                  $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A "LINE FEED"
001157     000                  $TPFLG: .BYTE   0               ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
001160  000000                  $TIMES: 0                       ;;MAX. NUMBER OF ITERATIONS
001162  000000                  $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
001164  177607  000377          $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
001170     077                  $QUES:  .ASCII  /?/             ;;QUESTION MARK
001171     015                  $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
001172  000012                  $LF:    .ASCIZ  <12>            ;;LINE FEED
                                ;;**************************************************************
                                .SBTTL  STORED PARAMETERS
001174  000000                  PUCODE: .WORD   0       ;PUNCH CODE (OBJECT OR SOURCE)
001176  000001                  DRIVE:  .WORD   1               ;LAST ADDRESSED DRIVE NUMBER
001200  000000                  CYLNUM: .WORD                   ;LAST GIVEN CYLINDER ADDRESS
001202  000000                  TRKNUM: .WORD                   ;LAST GIVEN TRACK ADDRESS
001204  000000                  SECNUM: .WORD                   ;LAST GIVEN SECTOR ADDRESS
001206  000400                  WDCNT:  .WORD   400             ;LAST GIVEN WORD COUNT
001210  000000                          .WORD   0               ;FILLER
001212  000026                  FORMAT: .WORD   26              ;LAST FORMAT SELECTED
001214  000000                  SUBCMD: .WORD                   ;LAST ENTERED SUBSYSTEM COMMAND
001216  000000                  STATRD: .WORD   0               ;LAST SELECTED STATUS MESSAGE
001220  000000                  STVAL:  .WORD                   ;LAST VALUE READ FROM STATRD
```

```
 1690  001222  177777          SMASK:  .WORD   177777          ;LAST GIVEN STATUS MASK
 1691  001224  000000          REGNUM: .WORD   0               ;LAST ADDRESSED RK611 REG.
 1692  001226  000000          REGVAL: .WORD                   ;LAST VALUE ENTERED OR READ FROM REGNUM
 1693  001230  177777          RMASK:  .WORD   177777          ;LAST GIVEN REGISTER MASK
 1694  001232  000001          ITCNT:  .WORD   1               ;ITERATION COUNT
 1695  001234     001          SFEMP:  .BYTE   1       ;SOURCE FILE EMPTY(NO VALID SOURCE)
 1696  001235     000          VLDOBJ: .BYTE   0       ;VALID OBJECT CODE
 1697  001236     101          PATSEL: .BYTE   101     ;LAST PATTERN SELECTED
 1698  001237     000          PATXDF: .BYTE   0       ;USER HAS DEFINED PAT X
 1699  001240     000          PATYDF: .BYTE   0       ;USER HAS DEFINED PAT Y
 1700  001241     000          PATZDF: .BYTE   0       ;USER HAS DEFINED PAT Z
 1701  001242     000          PATRDF: .BYTE   0       ;USER HAS DEFINED A RANDOM PAT
 1702  001243     000          LNCNT:  .BYTE           ;NUMBER OF LINES IN BUFFER
 1703  001244     000          OPFLGS: .BYTE   0
 1704          000002                  NOCK=   BIT1    ;BIT 1 - NO CHECK MODE SWITCH
 1705          000004                  BAII=   BIT2    ;BIT 2 - BUS ADDRESS INCREMENT INHIBIT SWITCH
 1706          000010                  SECT20= BIT3    ;BIT 3 - 20 SECTOR FORMAT
 1707          001246                  .EVEN
 1708  001246  000000          SFPTR:  .WORD           ;SOURCE FILE POINTER
 1709  001250  000000          PUFLSZ: .WORD   0       ;PUNCH FILE SIZE. NUM OF BYTES IN
 1710                                                  ;SOURCE OR OBJECT FILE.
 1711  001252  000000          STALL:  .WORD           ;STALL DURATION
 1712  001254  000000          COMSZE: .WORD   0       ;DATA COMPARE SIZE PARAMETER
 1713  001256  000000          LOFFST: .WORD   0       ;LAST OFFSET
 1714
 1715  001260  003720          TOVAL:  .WORD   ^D2000  ;TIMEOUT VALUE
 1716  001262  000040          PATX:   .BLKW   40      ;USER DEFINED PATTERN X
 1717
 1718  001362  000040          PATY:   .BLKW   40      ;USER DEFINED PATTERN Y
 1719
 1720  001462  000040          PATZ:   .BLKW   40      ;USER DEFINED PATTERN Z
 1721
 1722  001562  000040          PATR:   .BLKW   40      ;RANDOM PATTERN STORAGE
 1723                          .SBTTL  CONTROL PARAMETERS
 1724
 1725                          .EQUIV  $ERRPC,LINNUM
 1726  001662     000          PRINH:  .BYTE   0       ;PRINT INHIBIT SWITCH
 1727  001663     000          CHNFLG: .BYTE   0       ;CHAINING FLAG
 1728  001664     000          C$ERR:  .BYTE   0       ;COMPILE ERROR FLAG
 1729  001665     000          OFFLAG: .BYTE   0       ;OFFSET FLAG
 1730  001666     000          SAMDR:  .BYTE   0       ;SAME DRIVE SWITCH
 1731  001667     000          DONE:   .BYTE   0       ;DONE FLAG
 1732  001670     000          PBSW:   .BYTE   0       ;PARAMETER BLOCK SELECT SWITCH
 1733  001671     000          RPSWIT: .BYTE   0       ;REPORT PASS SWITCH
 1734  001672     377          HPVLD:  .BYTE   377     ;HELP VALID SWITCH
 1735          001674                  .EVEN
 1736  001674  001750          OBJSZE: .WORD   ^D1000  ;OBJECT FILESIZE
 1737  001676  000000          PATPTR: .WORD           ;POINTER TO PATTERN BUFFER
 1738
 1739  001700  000400          MAXWDS: .WORD   400     ;MAXIMUM WORD COUNT (SET BY PROGRAM)
 1740  001702  000000          TEMP1:  .WORD   0       ;TEMPORARY STORAGE
 1741  001704  000000          TEMP2:  .WORD   0       ;TEMPORARY STORAGE
 1742  001706  000000          OBUFPT: .WORD   0       ;OUTPUT BUFFER POINTER
 1743  001710  000000          IBUFPT: .WORD   0       ;INPUT BUFFER POINTER
 1744  001712  000000          OFPTR:  .WORD   0       ;OBJECT FILE POINTER
 1745  001714  000000          CNTSTR: .WORD   0       ;STORAGE FOR ITERATION COUNT
```

```
                                ;RETURN CONSTANT
                                ;JUMP CONSTANT
                                ;LOW ORDER LOOP COUNTER
                                ;HI ORDER LOOP COUNTER
                                ;PAPER TAPE READER STATUS REGISTER
                                ;PAPER TAPE READER DATA REGISTER
                                ;PAPER TAPE PUNCH STATUS REGISTER
                                ;PAPER TAPE PUNCH DATA REGISTER
                                ;READ ALL HEADERS BUFFER


                                ;CONTROL AND STATUS REGISTER 1
                                ;WORD COUNT REGISTER
                                ;BUS ADDRESS REGISTER
                                ;DESIRED TRACK SECTOR REGISTER
                                ;CONTROL AND STATUS REGISTER 2
                                ;DRIVE STATUS REGISTER
                                ;ERROR REGISTER
                                ;ATTENTION SUMMARY AND OFFSET REGISTER
                                ;DESIRED CYLINDER REGISTER
                                ;DESIRED CYLINDER REGISTER
                                ;DATA BUFFER
                                ;MAINTENANCE REGISTER 1
                                ;MAINTENANCE REGISTER 2
                                ;MAINTENANCE REGISTER 3
                                ;ECC POSITION INFORMATION
                                ;ECC POSITION INFORMATION
                                ;ECC PATTERN INFORMATION
                                ;ECC PATTERN INFORMATION


                                ;SELECT DRIVE
                                ;PACK ACKNOWLEDGE
                                ;DRIVE CLEAR
                                ;UNLOAD
                                ;START SPINDLE
                                ;RECALIBRATE
                                ;OFFSET
                                ;SEEK
                                ;READ DATA
                                ;WRITE DATA
                                ;READ HEADER
                                ;WRITE HEADER AND DATA
                                ;WRITE CHECK


                                ;RELEASE DRIVE
                                ;GET ALL STATUS FROM DRIVE
                                ;READ ALL HEADERS
                                ;CONTROLLER CLEAR (BIT 15 OF CS1)
                                ;SUBSYSTEM CLEAR (BIT 5 OF CS2)
```

```
1746  001716  000207          RTSPC:  .WORD   000207
1747  001720  004437          JSRR4:  .WORD   004437
1748  001722  000000          LPCNT1: .WORD   0
1749  001724  000000          LPCNT2: .WORD   0
1750  001726  177550          PTRSR:  .WORD   177550
1751  001730  177552          PTRDB:  .WORD   177552
1752  001732  177554          PTPSR:  .WORD   177554
1753  001734  177556          PTPDB:  .WORD   177556
1754  001736  000102          HDBUFF: .BLKW   102
1755
1756
1757                          .SBTTL  RK06 CONTROLLER REGISTER DEFINITION
1758
1759          000000          RKCS1=  0
1760          000002          RKWC=   2
1761          000004          RKBA=   4
1762          000006          RKDA=   6
1763          000010          RKCS2=  10
1764          000012          RKDS=   12
1765          000014          RKER=   14
1766          000016          RKASOF= 16
1767          000020          RKDC=   20
1768          000020          RKDCYL= 20
1769          000024          RKDB=   24
1770          000026          RKMR1=  26
1771          000034          RKMR2=  34
1772          000036          RKMR3=  36
1773          000030          RKPOS=  30
1774          000030          RKECPS= 30
1775          000032          RKPAT=  32
1776          000032          RKECPT= 32
1777
1778                          .SBTTL  DRIVE COMMANDS
1779
1780          000101          SELDRV= 101
1781          000103          PACK=   103
1782          000105          CLEAR=  105
1783          000107          UNLOAD= 107
1784          000111          SRTSPL= 111
1785          000113          RECAL=  113
1786          000115          OFFSET= 115
1787          000117          SEEK=   117
1788          000121          RDDATA= 121
1789          000123          WRDATA= 123
1790          000125          RDHEAD= 125
1791          000127          WRHEAD= 127
1792          000131          WRTCHK= 131
1793
1794                          ;       THE FOLLOWING ARE NOT DRIVE COMMANDS BUT ARE USED BY THE DRIVER
1795                          ;       TO SIMULATE A SPECIFIC DESIRED OPERATION
1796
1797          000140          RELEAS= 140
1798          000141          RDSTAT= 141
1799          000164          RDALHD= 164
1800          000176          CONCLR= 176
1801          000177          SUBCLR= 177
```

```
1802            000300                  INTR=   300                     ;GENERATE INTERRUPT TO CPU
1803
1804                                    ;       DRIVER ISSUED SERVICE COMMANDS
1805
1806            000001                  DR.SEL= 001                     ;DRIVE SELECT
1807            000005                  DR.CLR= 005                     ;DRIVE CLEAR
1809
1809                                    .SBTTL  CONTROL AND STATUS REGISTER 1 BITS
1810
1811            000001                  GO=     BIT0                    ;GO BIT
1812            000100                  IE=     BIT6                    ;INTERRUPT ENABLE
1813            000200                  RDY=    BIT7                    ;CONTROLLER READY
1814            000400                  BA16=   BIT8                    ;BUS ADDRESS BIT 16
1815            001000                  BA17=   BIT9                    ;BUS ADDRESS BIT 17
1816            002000                  CDT=    BIT10                   ;CONTROLLER DRIVE TYPE (0=RK06)
1817            004000                  CTO=    BIT11                   ;CONTROLLER TIMED OUT WAITING FOR
1818                                                                    ;  DRIVE RESPONSE
1319            010000                  CFMT=   BIT12                   ;CONTROLLER DRIVE FORMAT (0=22 SECTOR, 1=20 SECTOR)
1820            020000                  SPAR=   BIT13                   ;DRIVE BUS PARITY ERROR DETECTED BY CONTROLLER
1821            040000                  DI=     BIT14                   ;DRIVE INTERRUPT
1822            100000                  CERR=   BIT15                   ;CONTROLLER ERROR
1823            100000                  CCLR=   BIT15                   ;CONTROLLER CLEAR
1824
1825                                    ;       THESE BIT DEFINITIONS ARE USED FOR ADDRESS
1826                                    ;       THE HIGH BYTE OF RKCS1
1827
1828            000001                  B.BA16= BIT0                    ;BUS ADDRESS BIT 16
1829            000002                  B.BA17= BIT1                    ;BUS ADDRESS BIT 17
1830            000004                  B.CDT=  BIT2                    ;CONTROLLER DRIVE TYPE (0=RK06)
1831            000020                  B.CFMT= BIT4                    ;CONTROLLER DRIVE FORMAT (0=22 SECTOR, 1=20 SECTOR)
1832
1833                                    .SBTTL  CONTROL AND STATUS REGISTER 2 BITS
1834
1835            000007                  DRVMSK= 7                       ;MASK FOR DRIVE SELECTION CODE
1836            000010                  DESL=   BIT3                    ;DESELECT OR RELEASE DRIVE IN BITS 0-2
1837            000010                  RLS=    BIT3                    ;DESELECT OR RELEASE DRIVE IN BITS 0-2
1838            000020                  BAI=    BIT4                    ;BUS ADDRESS INCREMENT INHIBIT
1839            000040                  CLR=    BIT5                    ;CLEAR CONTROLLER AND ALL DRIVES
1840            000040                  SCLR=   BIT5                    ;CLEAR CONTROLLER AND ALL DRIVES
1841            000100                  IR=     BIT6                    ;INPUT READY
1842            000200                  OR=     BIT7                    ;OUTPUT READY
1843            000400                  UFE=    BIT8                    ;UNIT FIELD ERROR
1844            001000                  MDS=    BIT9                    ;MULTIPLE DRIVE SELECT
1845            002000                  PGE=    BIT10                   ;PROGRAMMING ERROR
1846            004000                  NEM=    BIT11                   ;NON-EXISTENT MEMORY
1847            010000                  NED=    BIT12                   ;NON-EXISTENT DRIVE
1848            020000                  UPE=    BIT13                   ;UNIBUS PARITY ERROR
1849            040000                  WCE=    BIT14                   ;WRITE CHECK ERROR
1850            100000                  DLT=    BIT15                   ;DATA LATE ERROR
1851
1852                                    .SBTTL  ERROR REGISTER BIT DEFINITION
1853
1854            000001                  ILC=    BIT0                    ;ILLEGAL FUNCTION CODE
1855                                    ;*ILF=  BIT0                    ;ILLEGAL FUNCTION CODE
1856            000002                  SKI=    BIT1                    ;SEEK INCOMPLETE
1857            000004                  ILF=    BIT2                    ;ILLEGAL DRIVE FUNCTION
```

```
1858            000004                  NXF=    BIT2                    ;ILLEGAL DRIVE FUNCTION
1859            000010                  DRPAR=  BIT3                    ;DRIVE DETECTED DRIVE BUS PARITY ERROR
1860            000020                  FMTE=   BIT4                    ;FORMAT ERROR
1861            000040                  DTYE=   BIT5                    ;DRIVE TYPE ERROR
1862            000100                  ECH=    BIT6                    ;ECC HARD
1863            000200                  BSE=    BIT7                    ;BAD SECTOR ERROR
1864            000400                  HCRC=   BIT8                    ;HEADER CRC ERRROR
1865            000400                  HVRC=   BIT8                    ;HEADER VRC ERRROR
1866            001000                  COE=    BIT9                    ;CYLINDER ADDRESS OVERFLOW ERROR
1867            002000                  IDAE=   BIT10                   ;INVALID DISK ADDRESS ERROR
1868            004000                  WLE=    BIT11                   ;WRITE LOCK ERROR
1869            010000                  DTE=    BIT12                   ;DRIVE TIMING ERROR
1870            020000                  OPI=    BIT13                   ;OPERATION (SEARCH) INCOMPLETE
1871            040000                  UNS=    BIT14                   ;DRIVE UNSAFE
1872            100000                  DCK=    BIT15                   ;DATA CHECK
1873
1874                                    .SBTTL  STATUS REGISTER BIT DEFINITION
1875
1876            000001                  DRA=    BIT0                    ;DRIVE AVAILABLE (CONTROLLER IS SET IF
1877                                                                    ;  THIS BIT IS RESET)
1878            000004                  OFST=   BIT2                    ;DRIVE OFFSET
1879            000010                  ACLO=   BIT3                    ;AC LOW
1880            000020                  SPDLSS= BIT4                    ;SPEED LOSS
1881            000020                  DCLO=   BIT4                    ;DC LOW
1882            000040                  DROT=   BIT5                    ;DRIVE OFF TRACK
1883            000100                  VV=     BIT6                    ;VOLUME VALID
1884            000200                  DRY=    BIT7                    ;DRIVE READY
1885            000200                  DRDY=   BIT7                    ;DRIVE READY
1886            000400                  DDT=    BIT8                    ;DRIVE TYPE (0=RK06)
1887            004000                  WRL=    BIT11                   ;WRITE LOCK
1888            020000                  PIP=    BIT13                   ;POSITIONING IN PROGRESS
1889            040000                  DSC=    BIT14                   ;DRIVE STATUS CHANGE
1890            100000                  SVAL=   BIT15                   ;STATUS VALID
1891
1892                                    .SBTTL  MAINTENANCE REGISTER 1 BIT DEFINITION
1893
1894            000017                  MESMSK= 17                      ;MESSAGE MASK
1895
1896            000020                  PAT=    BIT4                    ;FORCE EVEN PARITY ON DRIVE BUS MESSAGE LINES
1897            000040                  DMD=    BIT5                    ;DIAGNOSTIC MODE
1898            000100                  MSP=    BIT6                    ;MAINTENANCE SECTOR PULSE
1899            000200                  MIND=   BIT7                    ;MAINTENANCE INDEX
1900            000400                  MCLK=   BIT8                    ;MAINTENANCE CLOCK
1901            001000                  MERD=   BIT9                    ;MAINTENANCE ENCODED READ DATA
1902            002000                  MEWD=   BIT10                   ;MAINTENACME ENCODED WRITE DATA
1903            004000                  PCA=    BIT11                   ;PRECOMPENSATION ADVANCE
1904            010000                  PCD=    BIT12                   ;PRECOMPENSATION DELAY
1905            020000                  ECCW=   BIT13                   ;ECC WORD IS BEING READ OR WRITTEN
1906            040000                  WRTGAT= BIT14                   ;WRITE GATE
1907            100000                  RDGATE= BIT15                   ;READ GATE
1908
1909                                    .SBTTL  DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE A
1910
1911            000040                  S.DRA=  BIT5                    ;DRIVE AVAILIABLE
1912            000100                  S.VV=   BIT6                    ;VOLUME VALID
1913            000200                  S.DRY=  BIT7                    ;DRIVE READY
```

```
                        S.TYPE= BIT8                    ;DRIVE TYPE
                        S.FORM= BIT9                    ;DRIVE FORMAT
                        S.OFF=  BIT10                   ;OFFSET
                        S.WRL=  BIT11                   ;WRITE LOCK
                        S.SPIN= BIT12                   ;SPINDLE ON
                        S.PIP=  BIT13                   ;POSITIONING IN PROGRESS
                        S.DSC=  BIT14                   ;DRIVE STATUS CHANGE

                        .SBTTL  DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE B

                        S.ICYL= BIT5                    ;ILLEGAL CYLINDER ADDRESS
                        S.ACLO= BIT6                    ;AC LOW
                        S.FLT=  BIT7                    ;DRIVE FAULT
                        S.ILF=  BIT8                    ;ILLEGAL FUNCTION
                        S.PAR=  BIT9                    ;DRIVE DETECTED DRIVE BUS PARITY ERROR
                        S.SKI=  BIT10                   ;SEEK INCOMPLETE
                        S.WLE=  BIT11                   ;WRITE LOCK ERROR
                        S.SPLS= BIT12                   ;SPEED LOSS
                        S.DCLO= BIT12                   ;DC LOW
                        S.DROT= BIT13                   ;DRIVE OFF TRACK
                        S.UNS=  BIT14                   ;DRIVE UNSAFE

                        .SBTTL  DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE A

                        S.XDOK= BIT4                    ;TRANSDUCER OK
                        S.HOHM= BIT5                    ;HEADS HOME
                        S.BRHM= BIT6                    ;BRUSHES HOME
                        S.DOOR= BIT7                    ;DOOR INTERLOCKED
                        S.CART= BIT8                    ;CARTRAGE INTERLOCK
                        S.SPOK= BIT9                    ;SPEED OK
                        S.FWD=  BIT10                   ;FORWARD
                        S.REV=  BIT11                   ;REVERSE
                        S.LOAD= BIT12                   ;HEADS LOADING
                        S.RTZ=  BIT13                   ;RETURN TO ZERO
                        S.UNLD= BIT14                   ;HEADS UNLOADING

                        .SBTTL  DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE B

                        S.SECT= BIT4                    ;SECTOR ERROR
                        S.WCLK= BIT5                    ;WRITE CLOCK AND NO WRITE GATE
                        S.WGAT= BIT6                    ;WRITE GATE AND NO TRANSISTIONS
                        S.HDFL= BIT7                    ;HEAD FAULT
                        S.MHD=  BIT8                    ;MULTIPLE HEAD SELECT
                        S.XERR= BIT9                    ;INDEX ERROR
                        S.DIB=  BIT10                   ;DIBIT ERROR
                        S.PLO=  BIT11                   ;PLO ERROR
                        S.NMOV= BIT12                   ;SEEK AND NO MOTION
                        S.LIMD= BIT13                   ;LIMIT DETECT ON SEEK
                        S.BRKE= BIT14                   ;SERVO-BRAKE

                        .SBTTL  COMMON MASKS

                        M.DRV=  7                       ;DRIVE CODE
                        M.PAR=  BIT15                   ;PARITY
                        M.ID=   3                       ;BYTE ID
                        M.CDIF= 17760                   ;CYLINDER DIFFERENCE/OFFSET
```

```
1970  017760  M.CADD= 17760  ;CYLINDER ADDRESS
1971  077770  M.SER=  77770  ;DRIVE SERIAL NUMBER
1972  000760  M.SECT= 760   ;SECTOR COUNT
1973  007000  M.HEAD= 7000  ;HEAD DECODE
```

```
                        .SBTTL  PARAMETER BLOCK ALLOCATION

                        ;*      ************************************************
                        ;*  1   ! COMMAND                ! DRIVE NO.           !   0
                        ;*  3   ! CYLINDER ADDRESS                             !   2
                        ;*  5   ! TRACK                  ! SECTOR              !   4
                        ;*  7   !BA16-17,FORMAT,DRV TYPE! OFFSET               !   6
                        ;* 11   ! BUS ADDRESS (LOW 16 BITS)                    !  10
                        ;* 13   ! WORD COUNT (2'S COMPLEMENT)                  !  12
                        ;* 15   ! PROGRAM DRIVE STATUS INFORMATION             !  14
                        ;* 17   ! COMMAND AND STATUS REGISTER 1                !  16
                        ;* 21   ! COMMAND AND STATUS REGISTER 2                !  20
                        ;* 23   ! WORD COUNT REGISTER                          !  22
                        ;* 25   ! BUS ADDRESS REGISTER                         !  24
                        ;* 27   ! DESIRED TRACK AND SECTOR                     !  26
                        ;* 31   ! DESIRED CYLINDER                             !  30
                        ;* 33   ! ATTENTION SUMMARY AND DRIVE OFFSET           !  32
                        ;* 35   ! ERROR REGISTER                               !  34
                        ;* 37   ! STATUS REGISTER                              !  36
                        ;* 41   ! MESSAGE LINE A STATUS BYTE 00                !  40
                        ;* 43   ! MESSAGE LINE B STATUS BYTE 00                !  42
                        ;* 45   ! MESSAGE LINE A STATUS BYTE 01                !  44
                        ;* 47   ! MESSAGE LINE B STATUS BYTE 01                !  46
                        ;* 51   ! MESSAGE LINE A STATUS BYTE 10                !  50
                        ;* 53   ! MESSAGE LINE B STATUS BYTE 10                !  52
                        ;* 55   ! MESSAGE LINE A STATUS BYTE 11                !  54
                        ;* 57   ! MESSAGE LINE B STATUS BYTE 11                !  56
                        ;* 61   ! ECC POSITION INFORMATION                     !  60
                        ;* 63   ! ECC PATTERN INFORMATION                      !  62
                        ;*      ************************************************

                        .SBTTL  PARAMETERS PASSED TO THE DRIVER

                        ;       THE FOLLOWING DEFINITIONS ARE USED TO PASS PARAMETERS
                        ;         TO THE RK06 DRIVER

000000                  P.DRVN= 0                       ;DRIVE NUMBER
000001                  P.CMND= 1                       ;COMMAND
000002                  P.CYLN= 2                       ;CYLINDER ADDRESS
000004                  P.SECT= 4                       ;SECTOR
000005                  P.TRCK= 5                       ;TRACK
000006                  P.OFST= 6                       ;OFFSET
000007                  P.CS1H= 7                       ;RK.CS1 BITS 8-15
000007                  P.BAHI= 7                       ;BUS ADDRESS (BITS 16 AND 17)
000010                  P.BALO= 10                      ;BUS ADDRESS (BITS 0-15)
000012                  P.WC=   12                      ;WORD COUNT (2'S COMPLEMENT)
000014                  P.PRST= 14                      ;PROGRAM DRIVE STATUS INFORMATION

                        .SBTTL  PROGRAM DEVICE STATUS REGISTER DEFINITION

000001                  DRVUSE= BIT0                    ;DRIVE IN USE
000002                  DRVPOS= BIT1                    ;DRIVE POSITIONING
000004                  DRVPDT= BIT2                    ;DRIVE POSITIONED FOR DATA TRANSFER
000010                  UEXATT= BIT3                    ;UNEXPECTED ATTENTION
000020                  DRVHRD= BIT4                    ;DRIVE HAS HARD ERROR
000040                  DRVDSC= BIT5                    ;DRIVE STATUS CHANGE DID NOT CLEAR
```

```
2030            000100                  CMDTO=  BIT6                    ;NO TERMINATION TO COMMAND FOR AT
2031                                                                    ;  LEAST 1 SECOND
2032            000200                  W.WCK=  BIT7                    ;WRITE FOR WRITE WRITE CHECK
2033            000400                  NOCHK=  BIT8                    ;NO CHECK, DO NOT SET INTERRUPT ENABLE
2034            001000                  PBSVAL= BIT9                    ;PARAMETER STATUS WORDS VALID
2035                                                                    ;  (SET WHEN ERROR TERMINATION OR
2036                                                                    ;  READ STATUS COMMAND)
2037            002000                  DRPDRV= BIT10                   ;DROP DRIVE FROM TEST SEQUENCE
2038            004000                  NODSC=  BIT11                   ;ATTENTION SET BUT DCS AND FAULT RESET
2039            010000                  DRVSZD= BIT12                   ;DRIVE SEIZED BY OTHER PORT
2040            020000                  E.UNLD= BIT13                   ;DRIVE UNLOADED DUE TO ERROR
2041            040000                  Q.INIT= BIT14                   ;PARAMETER BLOCK ENQUEUED IN INITIATION QUEUE
2042            100000                  DTBAII= BIT15                   ;INHIBIT BUS ADDRESS INCREMENT
2043
2044                                    .SBTTL  PARAMETERS PASSED FROM DRIVER TO PROGRAM
2045
2046                                    ;       THE FOLLOWING DEFINITIONS ARE USED FOR REGISTER RETURNS
2047                                    ;         FROM THE DRIVER TO THE CALLING PROGRAM
2048
2049            000016                  P.CS1=  16                      ;COMMAND AND STATUS REGISTER 1
2050            000020                  P.CS2=  20                      ;COMMAND AND STATUS REGISTER 2
2051            000022                  P.WCR=  22                      ;WORD COUNT REGISTER
2052            000024                  P.BAR=  24                      ;BUS ADDRESS REGISTER
2053            000026                  P.DTS=  26                      ;DESIRED TRACK SECTOR REGISTER
2054            000030                  P.DCYL= 30                      ;DESIRED CYLINDER REGISTER
2055            000032                  P.ASOF= 32                      ;ATTENTION SUMMARY/OFFSET REGISTER
2056            000034                  P.ER=   34                      ;ERROR REGISTER
2057            000036                  P.DS=   36                      ;STATUS REGISTER
2058            000040                  P.A00=  40                      ;MESSAGE A STATUS BYTE 00
2059            000042                  P.B00=  42                      ;MESSAGE B STATUS BYTE 00
2060            000044                  P.A01=  44                      ;MESSAGE A STATUS BYTE 01
2061            000046                  P.B01=  46                      ;MESSAGE B STATUS BYTE 01
2062            000050                  P.A10=  50                      ;MESSGGE A STATUS BYTE 10
2063            000052                  P.B10=  52                      ;MESSAGE B STATUS BYTE 10
2064            000054                  P.A11=  54                      ;MESSAGE A STATUS BYTE 11
2065            000056                  P.B11=  56                      ;MESSAGE B STATUS BYTE 11
2066            000060                  P.EPOS= 60                      ;ECC POSITION INFORMATION
2067            000062                  P.EPAT= 62                      ;ECC PATTERN INFORMATION
2068            000064                  PRTCON=64                       ;PRINT CONTROL WORD
2069
2070                                    .SBTTL  PARAMETER BLOCK 0 FOR DRIVE
2071
2072    002142      000                 PARM0:  .BYTE   0               ;DRIVE NUMBER
2073    002143      000                         .BYTE   0               ;COMMAND
2074    002144  000000                          .WORD   0               ;CYLINDER ADDRESS
2075    002146      000                         .BYTE   0               ;SECTOR ADDRESS
2076    002147      000                         .BYTE   0               ;TRACK ADDRESS
2077    002150      000                         .BYTE   0               ;OFFSET VALUE
2078    002151      000                         .BYTE   0               ;BUS ADDRESS (BITS 16 AND 17)
2079    002152  000000                          .WORD   0               ;BUS ADDRESS (BITS 0 - 15)
2080    002154  000000                          .WORD   0               ;WORD COUNT (2'S COMPLEMENT)
2081    002156  000000                          .WORD   0               ;PROGRAM DRIVE STATUS INFORMATION
2082    002160  000000                          .WORD   0               ;COMMAND AND STATUS REGISTER 1
2083    002162  000000                          .WORD   0               ;COMMAND AND STATUS REGISTER 2
2084    002164  000000                          .WORD   0               ;WORD COUNT REGISTER
2085    002166  000000                          .WORD   0               ;BUS ADDRESS REGISTER
```

```
2086  002170  000000                    .WORD   0       ;DESIRED TRACK AND SECTOR REGISTER
2087  002172  000000                    .WORD   0       ;DESIRED CYLINDER REGISTER
2088  002174  000000                    .WORD   0       ;ATTENTION SUMMARY/OFFSET REGISTER
2089  002176  000000                    .WORD   0       ;ERROR REGISTER
2090  002200  000000                    .WORD   0       ;STATUS REGISTER
2091  002202  000000                    .WORD   0       ;MESSAGE LINE A STATUS BYTE 00
2092  002204  000000                    .WORD   0       ;MESSAGE LINE B STATUS BYTE 00
2093  002206  000000                    .WORD   0       ;MESSAGE LINE A STATUS BYTE 01
2094  002210  000000                    .WORD   0       ;MESSAGE LINE B STATUS BYTE 01
2095  002212  000000                    .WORD   0       ;MESSAGE LINE A STATUS BYTE 10
2096  002214  000000                    .WORD   0       ;MESSAGE LINE B STATUS BYTE 10
2097  002216  000000                    .WORD   0       ;MESSAGE LINE A STATUS BYTE 11
2098  002220  000000                    .WORD   0       ;MESSAGE LINE B STATUS BYTE 11
2099  002222  000000                    .WORD   0       ;ECC POSITION INFORMATION
2100  002224  000000                    .WORD   0       ;ECC PATTERN INFORMATION
2101  002226  000000                    .WORD   0       ;PRINT CONTROL WORD
2102
2103                          .SBTTL  PARAMETER BLOCK 1 FOR DRIVE
2104
2105  002230     000          PARM1:  .BYTE   0       ;DRIVE NUMBER
2106  002231     000                  .BYTE   0       ;COMMAND
2107  002232  000000                  .WORD   0       ;CYLINDER ADDRESS
2108  002234     000                  .BYTE   0       ;SECTOR ADDRESS
2109  002235     000                  .BYTE   0       ;TRACK ADDRESS
2110  002236     000                  .BYTE   0       ;OFFSET VALUE
2111  002237     000                  .BYTE   0       ;BUS ADDRESS (BITS 16 AND 17)
2112  002240  000000                  .WORD   0       ;BUS ADDRESS (BITS 0 - 15)
2113  002242  000000                  .WORD   0       ;WORD COUNT (2'S COMPLEMENT)
2114  002244  000000                  .WORD   0       ;PROGRAM DRIVE STATUS INFORMATION
2115  002246  000000                  .WORD   0       ;COMMAND AND STATUS REGISTER 1
2116  002250  000000                  .WORD   0       ;COMMAND AND STATUS REGISTER 2
2117  002252  000000                  .WORD   0       ;WORD COUNT REGISTER
2118  002254  000000                  .WORD   0       ;BUS ADDRESS REGISTER
2119  002256  000000                  .WORD   0       ;DESIRED TRACK AND SECTOR REGISTER
2120  002260  000000                  .WORD   0       ;DESIRED CYLINDER REGISTER
2121  002262  000000                  .WORD   0       ;ATTENTION SUMMARY/OFFSET REGISTER
2122  002264  000000                  .WORD   0       ;ERROR REGISTER
2123  002266  000000                  .WORD   0       ;STATUS REGISTER
2124  002270  000000                  .WORD   0       ;MESSAGE LINE A STATUS BYTE 00
2125  002272  000000                  .WORD   0       ;MESSAGE LINE B STATUS BYTE 00
2126  002274  000000                  .WORD   0       ;MESSAGE LINE A STATUS BYTE 01
2127  002276  000000                  .WORD   0       ;MESSAGE LINE B STATUS BYTE 01
2128  002300  000000                  .WORD   0       ;MESSAGE LINE A STATUS BYTE 10
2129  002302  000000                  .WORD   0       ;MESSAGE LINE B STATUS BYTE 10
2130  002304  000000                  .WORD   0       ;MESSAGE LINE A STATUS BYTE 11
2131  002306  000000                  .WORD   0       ;MESSAGE LINE B STATUS BYTE 11
2132  002310  000000                  .WORD   0       ;ECC POSITION INFORMATION
2133  002312  000000                  .WORD   0       ;ECC PATTERN INFORMATION
2134  002314  000000                  .WORD   0       ;PRINT CONTROL WORD
```

```
                                        .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


002316                                  $ERRTB:
002316  025012  051040  033113          PROGID: .ASCII  <12>@* Rk611/RK06 USER DEFINED TEST *@<15><12>
002324  030461  051057  030113
002332  020066  051525  051105
002340  042040  043105  047111
002346  042105  052040  051505
002354  020124  006452     012
002361     115  044501  042116          .ASCIZ  /MAINDEC-11-DZR6R-B/<15><12>
002366  041505  030455  026461
002374  055104  033122  026522
002402  006502  000012
002406  054524  042520  044040  HELPQ:  .ASCIZ  /TYPE HP TO PRINT HELP    .E/<15><12>
002414  020120  047524  050040
002422  044522  052116  044040
002430  046105  020120  044506
002436  042514  005015     000
002443     052     000          STAR:   .ASCIZ  /*/
002445     124  051505  020124  SCTOBG: .ASCIZ  /TEST PROGRAM TO BIG/<15><12>
002452  051120  043517  040522
002460  020115  047524  041040
002466  043511  005015     000
002473     123  052517  041522  NOROOM: .ASCIZ  /SOURCE FILE FULL/<15><12>
002500  020105  044506  042514
002506  043040  046125  006514
002514  000012
002516  000075                       EQSGN:  .ASCIZ  /=/
002520  047111  046126  020104  BADDEC: .ASCIZ  /INVLD DECIMAL/<15><12>
002526  042504  044503  040515
002534  006514  000012
002540  047111  046126  020104  IVDEDT: .ASCIZ  /INVLD EDIT CMND/<15><12>
002546  042105  052111  041440
002554  047115  006504  000012
002562  047111  046126  020104  IVDADD: .ASCIZ  /INVLD NEW CMND/<15><12>
002570  042516  020127  046503
002576  042116  005015     000
002603     111  053116  042114  IVDLN:  .ASCIZ  /INVLD LINE NUM/<15><12>
002610  046040  047111  020105
002616  052516  006515  000012
002624  047111  046126  020104  IVDPAR: .ASCIZ  /INVLD PARAMETER/<15><12>
002632  040520  040522  042515
002640  042524  006522  000012
002646  047111  046126  020104  BADOCT: .ASCIZ  /INVLD OCTAL/<15><12>
```

```
  2191  002654  041517  040524  006514
  2192  002662  000012
  2193  002664  042510  050114  043040  HPFILE: .ASCIZ  /HELP FILE AVAILABLE ONLY AS FIRST COMMAND AFTER PROGRAM LOAD./<15><12>
  2194  002672  046111  020105  053101
  2195  002700  044501  040514  046102
  2196  002706  020105  047117  054514
  2197  002714  040440  020123  044506
  2198  002722  051522  020124  047503
  2199  002730  046515  047101  020104
  2200  002736  043101  042524  020122
  2201  002744  051120  043517  040522
  2202  002752  020115  047514  042101
  2203  002760  006456  000012
  2204  002764  047516  047440  045102  IVDRUN: .ASCIZ  /NO OBJECT/<15><12>
  2205  002772  041505  006524  000012
  2206  003000  047503  050115  046111  COMPOK: .ASCIZ  /COMPILE OK/<15><12>
  2207  003006  020105  045517  005015
  2208  003014     000
  2209  003015     116  020117  047523  NOSRC:  .ASCIZ  /NO SOURCE/<15><12>
  2210  003022  051125  042503  005015
  2211  003030     000
  2212  003031     111  052116  051105  INTERR: .ASCIZ  /INTERNAL COMPILER ERROR/<15><12>
  2213  003036  040516  020114  047503
  2214  003044  050115  046111  051105
  2215  003052  042440  051122  051117
  2216  003060  005015     000
  2217  003063     111  053116  042114  BADCOM: .ASCIZ  /INVLD OR UNDEF CMND/<15><12>
  2218  003070  047440  020122  047125
  2219  003076  042504  020106  046503
  2220  003104  042116  005015     000
  2221  003111     127  051117  020104  IVDWCT: .ASCIZ  /WORD COUNT TOO BIG/<15><12>
  2222  003116  047503  047125  020124
  2223  003124  047524  020117  044502
  2224  003132  006507  000012
  2225  003136  047111  046126  020104  IVDNC:  .ASCIZ  /INVLD CMND IN NC/<15><12>
  2226  003144  046503  042116  044440
  2227  003152  020116  041516  005015
  2228  003160     000
  2229  003161     111  053116  042114  BADDRV: .ASCIZ  /INVLD DRIVE NUM/<15><12>
  2230  003166  042040  044522  042526
  2231  003174  047040  046525  005015
  2232  003202     000
  2233  003203     120  047125  044103  PUERR:  .ASCIZ  /PUNCH ERR/<15><12>
  2234  003210  042440  051122  005015
  2235  003216     000
  2236  003217     122  040505  042504  PRERR:  .ASCIZ  /READER ERR/<15><12>
  2237  003224  020122  051105  006522
  2238  003232  000012
  2239  003234  042012  044522  042526  STNTVD: .ASCIZ  <12>/DRIVE STATUS NOT VALID/<15><12><12>
  2240  003242  051440  040524  052524
  2241  003250  020123  047516  020124
  2242  003256  040526  044514  006504
  2243  003264  005012     000
  2244  003267     075  047524  040524  TOTMSC: .ASCIZ  /=TOTAL MISCOMPARES/<15><12><12>
  2245  003274  020114  044515  041523
  2246  003302  046517  040520  042522
```

```
2247  003310  006523  005012     000
2248  003315     075  040[illegible]5  020130  ICBFSZ: .ASCIZ  /=MAX WORD COUNT FOR DATA TRANSFER/<15><12>
2249  003322  047527  04[illegible]  041440
2250  003330  052517  05[illegible]  043040
2251  003336  051117  042040  052101
2252  003344  020101  051124  047101
2253  003352  043123  051105  005015
2254  003360     000
2255  003361     111  046114  043505  BADSEL: .ASCIZ  /ILLEGAL REGISTER SELECTED/<15><12>
2256  003366  046101  051040  043505
2257  003374  051511  042524  020122
2258  003402  042523  042514  052103
2259  003410  042105  005015     000
2260  003415     075  052516  041115  LPLABL: .ASCIZ  /=NUMBER OF LOOPS/<15><12><12>
2261  003422  051105  047440  020106
2262  003430  047514  050117  006523
2263  003436  005012     000
2264  003441     052  046040  047517  LCNTOF: .ASCIZ  /* LOOP COUNTER OVERFLOW */<15><12>
2265  003446  020120  047503  047125
2266  003454  042524  020122  053117
2267  003462  051105  046106  053517
2268  003470  025040  005015     000
2269  003475     127  051117  020104  DMPHDR: .ASCIZ  /WORD #          CONTENTS/<15><12>
2270  003502  020043  020040  020040
2271  003510  020040  020040  020040
2272  003516  047503  052116  047105
2273  003524  051524  005015     000
2274  003531     040  020040  020040  SPACE6: .ASCIZ  /      /
2275  003536  000040
2276  003540  020040     000          SPACE2: .ASCIZ  /  /
2277
2278                                  ;;*****************************************************************
2279                                  .SBTTL  TABLE OF INTERACTIVE COMMANDS
2280                                  ;*THIS TABLE CONTAINS ALL THE INTERACTIVE COMMANDS.
2281                                  ;*THERE ARE 4 WORDS PER ENTRY:
2282                                  ;*WORD 1         COMMAND MNEUMONIC
2283                                  ;*WORD 2         IF IMMEDIATE, ADDRESS OF INTERACTIVE COMMAND PROCESSOR
2284                                  ;*               ROUTINE FOR THIS COMMAND. IF DEFERRED, THE
2285                                  ;*               ADDRESS OF THE COMPILATION ROUTINE FOR THIS COMMAND.
2286                                  ;*WORD 3         IF DEFERRED, NUMBER OF PARAMETERS ASSOCIATED WITH
2287                                  ;*               THIS COMMAND. IF IMMEDIATE, -1.
2288                                  ;*WORD 4                IF DEFERRED, ADDRESS OF EXECUTE ROUTINE
2289                                  ;*               FOR THIS COMMAND. IF IMMEDIATE, ALL 0'S
2290                                  ;*THIS TABLE IS USED IN COMMAND ENTRY AND TEST
2291                                  ;*COMPILATION
2292                                  ;;*****************************************************************
2293          003544                      .EVEN
2294  003544  047524                  ICTBL:  .ASCII  /TO/            ;TIMEOUT
2295  003546  006062  177777  000000          .WORD   TORTE,-1,0
2296  003554  047104                          .ASCII  /DN/           ;DRIVE SELECT
2297  003556  006144  177777  000000          .WORD   DNRTE,-1,0
2298  003564  052117                          .ASCII  /OT/           ;OUTPUT TEST
2299  003566  012420  177777  000000          .WORD   OTRTE,-1,0
2300  003574  052111                          .ASCII  /IT/           ;INPUT TEST
2301  003576  013020  177777  000000          .WORD   ITRTE,-1,0
2302  003604  052103                          .ASCII  /CT/           ;COPY TAPE
```

```
2303  003606  022114  177777  000000          .WORD   CTRTE,-1,0
2304  003614  051511                          .ASCII  /IS/            ;INPUT STRING
2305  003616  013012  177777  000000          .WORD   ISRTE,-1,0
2306  003624  041511                          .ASCII  /IC/            ;ITERATION COUNT
2307  003626  006254  177777  000000          .WORD   ICRTE,-1,0
2308  003634  052106                          .ASCII  /FT/            ;FORMAT SELECT
2309  003636  006000  177777  000000          .WORD   FTRTE,-1,0
2310  003644  041105                          .ASCII  /EB/            ;EDIT SPECIAL BUFFERS
2311  003646  006346  177777  000000          .WORD   EBRTE,-1,0
2312  003654  050104                          .ASCII  /DP/            ;SPECIAL DATA PATTERN
2313  003656  006356  177777  000000          .WORD   DPRTE,-1,0
2314  003664  047503                          .ASCII  /CO/            ;COMPILE
2315  003666  010302  177777  000000          .WORD   CORTE,-1,0
2316  003674  040505                          .ASCII  /EA/            ;EDIT ADD LINE
2317  003676  004764  177777  000000          .WORD   EARTE,-1,0
2318  003704  042105                          .ASCII  /ED/            ;EDIT DELETE LINE
2319  003706  005266  177777  000000          .WORD   EDRTE,-1,0
2320  003714  052120                          .ASCII  /PT/            ;PRINT TEST
2321  003716  005700  177777  000000          .WORD   PTRTE,-1,0
2322  003724  046120                          .ASCII  /PL/            ;PRINT LINE
2323  003726  005500  177777  000000          .WORD   PLRTE,-1,0
2324  003734  052116                          .ASCII  /NT/            ;NEW TEST
2325  003736  007410  177777  000000          .WORD   NTRTE,-1,0
2326  003744  052522                          .ASCII  /RU/            ;RUN
2327  003746  010030  177777  000000          .WORD   RURTE,-1,0
2328  003754  051120                          .ASCII  /PR/            ;PRINT REGISTER
2329  003756  006740  177777  000000          .WORD   PRRTE,-1,0
2330  003764  050110                          .ASCII  /HP/            ;HELP
2331  003766  007364  177777  000000          .WORD   HPRTE,-1,0
2332  003774  042102                          .ASCII  /BD/            ;BUFFER DUMP
2333  003776  007522  177777  000000          .WORD   BDRTE,-1,0
2334  004004  043123                          .ASCII  /SF/            ;SUBSYSTEM FUNCTION
2335  004006  011570  000010  000000          .WORD   C$SF,10,0
2336  004014  044502                          .ASCII  /BI/            ;BUFFER INITIALIZE
2337  004016  011154  000001  020536  C$CBIS:  .WORD   C$BI,1,E$BI     ;SPECIAL TAG FOR BUFFER INIT
2338  004024  041504                          .ASCII  /DC/            ;DATA COMPARE
2339  004026  011264  000001  020204          .WORD   C$DC,1,E$DATC
2340  004034  041523                          .ASCII  /SC/            ;STATUS COMPARE
2341  004036  010762  000003  017610          .WORD   C$SC,3,E$SC
2342  004044  041522                          .ASCII  /RC/            ;REGISTER COMPARE
2343  004046  010744  000003  020026          .WORD   C$RC,3,E$REGC
2344  004054  053522                          .ASCII  /RW/            ;REGISTER WRITE
2345  004056  010726  000002  020426          .WORD   C$RW,2,E$RW
2346  004064  052123                          .ASCII  /ST/            ;STALL
2347  004066  011342  000001  020446          .WORD   C$ST,1,E$ST
2348  004074  046520                          .ASCII  /PM/            ;PRINT MESSAGE
2349  004076  010666  000001  013516          .WORD   C$PM,1,E$PM
2350  004104  044525                          .ASCII  /UI/            ;UNIBUS INITIALIZE
2351  004106  011414  000000  021424          .WORD   C$UI,0,E$UI
2352  004114  025052                          .ASCII  /**/            ;END OF TABLE
2353                          ;;**********,**********
2354
2355  004116                  UDTSRT:
2356                          .SBTTL  INITIALIZE THE COMMON TAGS
2357                          ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
2358  004116  012706  001100          MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
```

```
2359  004122  005026                          CLR     (R6)+           ;;CLEAR MEMORY LOCATION
2360  004124  022706  001140                  CMP     #SWR,R6 ;:DONE?
2361  004130  001374                          BNE     .-6             ;;LOOP BACK IF NO
2362  004132  012706  001100                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
2363                          ;;INITIALIZE A FEW VECTORS
2364  004136  012737  034362  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
2365  004144  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
2366  004152  012737  034204  000024          MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
2367  004160  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;LEVEL 7
2368                          ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
2369                          ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
2370  004166  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
2371  004172  012737  004226  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
2372  004200  012737  177570  001140          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
2373  004206  012737  177570  001142          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
2374  004214  022777  177777  174716          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
2375  004222  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
2376                                                                  ;;AND  THE HARDWARE SWR IS NOT = -1
2377  004224  000403                          BR      65$             ;;BRANCH IF NO TIMEOUT
2378  004226  012716  004234          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
2379  004232  000002                          RTI
2380  004234  012737  000176  001140  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
2381  004242  012737  000174  001142          MOV     #DISPREG,DISPLAY
2382  004250  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
2383
2384  004254  013701  023344                  MOV     RKVEC,R1        ;GET VECTOR STORAGE ADDRESS
2385  004260  012721  023552                  MOV     #I.INTR,(R1)+   ;LOAD IT WITH INTERRUPT HNDLR ADDR
2386  004264  012711  000340                  MOV     #PR7,(R1)       ;SET PSW TO PRIORITY 7
2387  004270  004737  034060                  JSR     PC,$SIZE
2388  004274  013700  034202                  MOV     $LSTAD,R0       ;GET LAST ADDRESS VALUE
2389  004300  012737  041416  001706          MOV     #ENDLOC,OBUFPT  ;SET OUTPUT BUFFER POINTER
2390  004306  162700  006000                  SUB     #6000,R0        ;ALLOW FOR XXDP LOADER
2391  004312  162700  041416                  SUB     #ENDLOC,R0      ;SUBTRACT FOR PROGRAM CODE
2392  004316  000241                          CLC                     ;CLEAR CARRY
2393  004320  006000                          ROR     R0              ;DIVIDE BY TWO FOR TWO BUFFERS
2394  004322  042700  000001                  BIC     #BIT0,R0        ;MAKE SURE ITS EVEN
2395  004326  012737  041416  001710          MOV     #ENDLOC,IBUFPT  ;SET THE INPUT BUFFER POINTER
2396  004334  060037  001710                  ADD     R0,IBUFPT       ;AT THE MIDDLE OF BUFFER AREA
2397  004340  000241                          CLC                     ;CLEAR CARRY
2398  004342  006000                          ROR     R0              ;DIVIDE BY TWO FOR MAXIMUM WORDS
2399  004344  162700  000002                  SUB     #2,R0           ;MAKE IT TWO LESS
2400  004350  010037  001700                  MOV     R0,MAXWDS       ;SET MAX WORD VALUE
2401  004354  004737  032156                  JSR     PC,$TKINT       ;INITIALIZE KEYBOARD
2402  004360  104400  002316                  TYPE    ,PROGID         ;TYPE PROGRAM NAME
2403  004364  010046                          MOV     R0,-(SP)
2404  004366  104401                          TYPOC                   ;TYPE MAX WORDS
2405  004370  104400  003315                  TYPE    ,IOBFSZ         ;LABEL IT
2406  004374  005046                          CLR     -(SP)           ;;PUT NEW PS ON STACK
2407  004376  012746  004404                  MOV     #67$,-(SP)      ;;PUT NEW PC ON  STACK
2408  004402  000002                          RTI                     ;;POP NEW PC AND PS
2409  004404                          67$:
2410  004404  105737  001673                  TSTB    HPVLD           ;TEST IF HELP FILE VALID
2411  004410  001402                          BEQ     COMLEV          ;NO - DON'T PRINT HELP QUESTION
2412  004412  104400  002406                  TYPE    ,HELPQ          ;TYPE HELP QUESTION
2413          ;;*****************************************************************
2414          .SBTTL  COMMAND LEVEL ROUTINE
```

```
                                ;*THIS ROUTINE WAITS FOR AND ACCEPTS A COMMAND FROM THE COSOLE AND DECODES
                                ;*IT TO DETERMINE IF THE COMMAND IS IMMEDIATE, DEFERRED, OR AN UNDEFINED.
                                ;*IF THE COMMAND IS DEFERRED IT CALLS THE DEFERRED COMMAND PROCESSOR IT
                                ;*(DCMDPR).  IF THE COMMAND IS IMMEDIATE THE 2ND WORD OF THE
                                ;*INTERACTIVE COMMAND TABLE (ICTBL) IS THE ADDRESS OF A SPECIAL
                                ;*ROUTINE FOR THAT COMMAND.


                                ;*      RETURN TO COMLEV IS
                                ;*      NORMAL - TST    (R4)+
                                ;*               RTS    R4
                                ;*      ERROR - MOV     (R4),R4
                                ;*               RTS    R4
                                ;;*****************************************************************

004416  104400  002443  COMLEV: TYPE    ,STAR           ;TYPE COMMAND LEVEL DESIGNATOR
004422  104410                  RDLIN                   ;READ COMMAND LINE
004424  012601                  MOV     (SP)+,R1        ;MOVE ADDR OF COMMAND R1
004426  004437  004704          JSR     R4,ICDEC        ;CALL INTERACTIVE COMMAND DECODE
004432  004470                  2$                      ;ERROR RETURN, GO TO ERROR
004434  005722                  TST     (R2)+           ;BUMP TO PARAM WORD
004436  005712                  TST     (R2)            ;TEST PARAM WORD
004440  100005                  BPL     1$              ;IF DEFERRED, BRANCH. ELSE
004442  005742                  TST     -(R2)           ;DEC TO ADDRESS WORD
004444  004472  000000          JSR     R4,@(R2)                ;BR IMMEDIATE COMMAND
                                                        ;ROUTINE WHOSE ADDRESS WAS
                                                        ;PLACED IN R2 BY ICDEC.
004450  004476                  3$                      ;ERROR RETURN
004452  000403                  BR      4$              ;RETURN TO COMMAND LEVEL
004454  004437  004510  1$:     JSR     R4,DCMDPR       ;JUMP TO DEFERRED CMD PROCESSOR
004460  004476                  3$                      ;ERROR RETURN
004462  105037  001672  4$:     CLRB    HPVLD           ;CLEAR HELP VALID SWITCH
004466  000753                  BR      COMLEV          ;RETURN TO COMMAND LEVEL
004470  104400  003063  2$:     TYPE    ,BADCOM         ;TYPE BAD COMMAND MESSAGE
004474  000400                  BR      3$
004476  104400  033332  3$:     TYPE    ,$TTYIN         ;TYPE LINE IN ERROR
004502  104400  001170          TYPE    ,$QUES          ;FOLLOWED BY QUESTION MARK
004506  000743                  BR      COMLEV          ;RETURN TO COMMAND LEVEL

                                ;:*****************************************************************
                                .SBTTL  DEFERRED COMMANDS INPUT PROCESSER
                                ;*ENTRY:        JSR     PC,DCMDPR
                                ;*WITH R1 POINTING TO INPUT CMND.
                                ;*RETURN:       NORMAL  TST     (R4)+
                                ;*                      RTS     R4
                                ;*              ERROR   MOV     (R4),R4
                                ;*                      RTS     R4
                                ;*THIS ROUTINE WILL PLACE THE DEFERRED COMMAND
                                ;*INTO THE SOURCE FILE. THE SOURCE FILE EMPTY
                                ;*(SFEMP) FLAG IS CHECKED AND IF SET THE SOURCE
                                ;*FILE IS CLEARED AND COUNTERS & POINTER INITIALIZED.
                                ;*A LINE NUMBER IS PREFIXED TO THE DEFERRED
                                ;*COMMAND AND STORED WITH THE COMMAND. THE
                                ;*LINE TERMINATOR (NULL) IS KEPT WITH THE COMMAND.
```

```
                                        ;*1000 WORDS ARE ALLOCATED FOR SOURCE FILE AND
                                        ;*IS NOT ALLOWED TO OVERFLOW. WHEN THIS ROUTINE IS
                                        ;*CALLED R1 MUST BE POINTING TO THE INPUT DEFERRED
                                        ;*COMMAND.
                                        ;;*****************************************************************
004510  010346                  DCMDPR: MOV     R3,-(SP)        ;STORE R3
004512  010546                          MOV     R5,-(SP)        ;STORE R5
004514  105737  001234                  TSTB    SFEMP           ;TEST SOURCE FILE EMPTY
004520  001420                          BEQ     2$              ;BR IF NOT EMPTY, APPEND TO SOURCE
004522  105037  001243                  CLRB    LNCNT           ;CLEAR LINE COUNTER
004526  013703  001700                  MOV     MAXWDS,R3       ;SET BUFFERSIZE
004532  013737  001710  001246          MOV     IBUFPT,SFPTR    ;SETUP SOURCE FILE POINTER
004540  013705  001710                  MOV     IBUFPT,R5       ;SET UP TO CLEAR SOURCE FILE.
004544  162705  000002                  SUB     #2,R5           ;START CLEAR 1 WORD EARLY
004550  005025                  1$:     CLR     (R5)+           ;CLEAR TO ZEROS
004552  005303                          DEC     R3              ;AND
004554  001375                          BNE     1$              ;LOOP UNTIL INPUT BUFFER CLEARED
004556  105037  001234                  CLRB    SFEMP           ;CLEAR SOURCE FILE EMPTY

004562  013705  001246          2$:     MOV     SFPTR,R5        ;SET UP R5 AS SOURCE FILE PTR
004566  013737  001700  001702          MOV     MAXWDS,TEMP1    ;STORE SF SIZE
004574  162737  000016  001702          SUB     #16,TEMP1       ;ALLOW FOR THIS LINE
004602  063737  001710  001702          ADD     IBUFPT,TEMP1    ;COMPUTE LAST ADDR OF SF
004610  020537  001702                  CMP     R5,TEMP1        ;TEST FOR SUFFICIENT ROOM FOR
004614  101406                          BLOS    3$              ;THIS LINE. BR IF YES
004616  104400  002473                  TYPE    ,NOROOM         ;TYPE NO ROOM MESSAGE
004622  012605                          MOV     (SP)+,R5        ;RESTORE R5
004624  012603                          MOV     (SP)+,R3        ;RESTORE R3
004626  011404                          MOV     (R4),R4         ;SET UP ERROR RETURN
004630  000414                          BR      5$              ;GO TO EXIT
004632  105237  001243          3$:     INCB    LNCNT           ;BUMP LINE COUNT
004636  113725  001243                  MOVB    LNCNT,(R5)+     ;PUT LINE COUNT IN SOURCE
004642  112115                  4$:     MOVB    (R1)+,(R5)      ;MOVE INPUT CMP TO SOURCE
004644  105725                          TSTB    (R5)+           ;TEST IF LAST CHAR MOVED IS NULL
004646  001375                          BNE     4$              ;BR IF NOT YET NULL
004650  010537  001246                  MOV     R5,SFPTR        ;STORE OFF NEW SOURCE FILE PTR
004654  012605                          MOV     (SP)+,R5        ;RESTORE R5
004656  012603                          MOV     (SP)+,R3        ;RESTORE R3
004660  005724                          TST     (R4)+           ;SET UP NORMAL RETURN
004662  000204                  5$:     RTS     R4              ;RETURN TO CALLER

                                        ;;*****************************************************************
                                        .SBTTL  SEARCH BYTE STRING FOR COMMA OR NULL
                                        ;*ENTRY
                                        ;*      JSR     PC,SBSCN
                                        ;*      R1 POINTS TO FIRST CHAR OF STRING
                                        ;*
                                        ;*RETURN
                                        ;*      RTS     PC
                                        ;*      WITH R1 NOW POINTING TO FIRST CHARACTER
                                        ;*      AFTER A COMMA OR TO THE NULL
                                        ;*
                                        ;;*****************************************************************
004664      000                 NULL:   .BYTE   0
004665      054                 COMMA:  .ASCII  /,/
```

```
2527  004666                          SBSCN:
2528  004666  121137  004664          1$:     CMPB    (R1),NULL       ;TEST FOR NULL
2529  004672  001403                          BEQ     2$              ;BR IF YES
2530  004674  122137  004665                  CMPB    (R1)+,COMMA     ;TEST OF COMMA
2531  004700  001372                          BNE     1$              ;LOOP IF NOT
2532  004702  000207                  2$:     RTS     PC              ;RETURN
2533
2534                                  ;;*****************************************************************
2535                                  .SBTTL  INTERACTIVE COMMAND DECODE ROUTINE
2536                                  ;*ENTRY:
2537                                  ;*      JSR     R4,ICDEC
2538                                  ;*              WITH R1 CONTAINING THE ADDRESS OF THE COMMAND LINE
2539                                  ;*RETURN:       MOV     (R4),R4
2540                                  ;*              RTS     R4      ERROR RETURN
2541                                  ;*              TST     (R4)+
2542                                  ;*              RTS     R4      NORMAL RETURN
2543                                  ;*
2544                                  ;*WHEN RETURNED R2 WILL POINT TO THE SECOND WORD OF
2545                                  ;*THE MATCHING TABLE ENTRY.
2546                                  ;*
2547                                  ;*
2548                                  ;*THIS ROUTINE SEARCHES THE TABLE OF INTERACTIVE
2549                                  ;*COMMANDS, LOOKING FOR A MATCH FOR THE COMMAND
2550                                  ;*POINTED TO BY R1. IF NO MATCH THE ROUTINE RETURNS
2551                                  ;*BACK TO RETURN 1. IF MATCH OCCURS RETURN IS TO RETURN 2.
2552                                  ;*R2 POINTS TO SECOND WORD OF TABLE ENTRY WHICH IS
2553                                  ;*ADDRESS OF INTERACTIVE COMMAND PROCESSOR
2554                                  ;*SUBROUTINE. THE CALLING ROUTINE MUST
2555                                  ;*STORE R2 BEFORE CALLING ICDEC$
2556                                  ;;*****************************************************************
2557
2558  004704  010346                  ICDEC:  MOV     R3,-(SP)        ;STORE R3
2559  004706  112137  001702                  MOVB    (R1)+,TEMP1     ;MOVE COMMAND INTO R3
2560  004712  111137  001703                  MOVB    (R1),TEMP1+1    ;TO INSURE WORD
2561  004716  013703  001702                  MOV     TEMP1,R3        ;ALIGNMENT FOR EASE OF COMPARE
2562  004722  005301                          DEC     R1              ;RESTORE R1 TO BEGINNING OF CMD
2563  004724  012702  003544                  MOV     #ICTBL,R2       ;ADDRESS OF TABLE INTO R2
2564  004730  020312                  1$:     CMP     R3,(R2)         ;TEST TABLE ENTRY AGAINST
2565  004732  001410                          BEQ     3$              ;ENTERED COMMAND. BR IF HIT
2566  004734  121237  002443                  CMPB    (R2),STAR       ;TEST IF END OF TABLE.
2567  004740  001403                          BEQ     2$              ;IF YES DO ERROR RETURN
2568  004742  062702  000010                  ADD     #10,R2          ;BUMP R2 TO NEXT TABLE ENTRY
2569  004746  000770                          BR      1$              ;LOOP - TEST NEXT ENTRY
2570  004750  011404                  2$:     MOV     (R4),R4         ;SET UP ERROR RETURN
2571  004752  000402                          BR      4$              ;JUMP TO EXIT
2572  004754  005724                  3$:     TST     (R4)+           ;SET UP FOR NO ERROR RETURN
2573  004756  005722                          TST     (R2)+           ;BUMP R2 TO SECOND WORD
2574                                                                  ;OF TABLE
2575  004760  012603                  4$:     MOV     (SP)+,R3        ;RESTORE R3
2576  004762  000204                          RTS     R4              ;RETURN TO CALLER
2577
2578                                  ;;*****************************************************************
2579                                  .SBTTL  EDIT ADD LINE ROUTINE
2580                                  ;*ENTRY:
2581                                  ;*      JSR     R4,EARTE
2582                                  ;*      R1 POINTS TO COMMAND LINE R2 POINTS TO 2ND WORD
```

```
                                        ;*      IN ICTBL
                                        ;*RETURN
                                        ;*      RTS     R4      IF ERROR
                                        ;*      RTS     R4+2    IF NO ERROR
                                        ;*ROUTINES CALLED
                                        ;*      ICDEC
                                        ;*      DECBIN
                                        ;*
                                        ;*THIS ROUTINE IS USED TO INSERT A NEW LINE INTO THE
                                        ;*SOURCE FILE. THE LINES BELOW THE ENTERED COMMAND
                                        ;*ARE SHIFTED DOWN, THE NEW LINE IS ENTERED, AND
                                        ;*THE COMMAND LINE NUMBERS ARE RESEQUENCED.
                                        ;*THE LINE ADDED IS CHECKED TO INSURE IT IS A
                                        ;*DEFERRED COMMAND. IF IT IS NOT, THE COMMAND IS
                                        ;*NOT ADDED AND AN ERROR IS REPORTED. THE SOURCE
                                        ;*FILE LENGTH IS CHECKED FOR SUFFICIENT ROOM.

004764                                  EARTE:
004764  010046                                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
004766  010146                                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
004770  010246                                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
004772  010346                                  MOV     R3,-(SP)        ;;PUSH R3 ON STACK
004774  010546                                  MOV     R5,-(SP)        ;;PUSH R5 ON STACK
004776  013705  001246                         MOV     SFPTR,R5        ;LOAD SOURCE FILE PTR INTO R5
005002  013700  001710                         MOV     IBUFPT,R0       ;GET START OF INPUT BUFFER
005006  063700  001700                         ADD     MAXWDS,R0       ;ADD THE MAX WORDS
005012  162700  000020                         SUB     #20,R0          ;ALLOW FOR THIS COMMAND
005016  020500                                  CMP     R5,R0            ;CHECK IF ROOM FOR THIS CMD
005020  101403                                  BLOS    1$              ;BR IF YES
005022  104400  002473                         TYPE    ,NOROOM         ;TYPE NO ROOM MESSAGE
005026  000506                                  BR      25$             ;GO TO EXIT
005030  004737  004666          1$:             JSR     PC,SBSCN        ;BUMP R1 TO NEXT PARAM (LN)
005034  121137  004664                         CMPB    (R1),NULL       ;CHECK IF NULL
005040  001471                                  BEQ     21$             ;BR IF YES (ERROR)
005042  010146                                  MOV     R1,-(SP)        ;ADDRESS OF ASCII LINE NUMBER
005044  004737  031102                         JSR     PC,DECBIN       ;CONVERT IT TO BINARY
005050  005216                                  20$
005052  012603                                  MOV     (SP)+,R3        ;STORE DECODED LINE NUMBER
005054  113700  001243                         MOVB    LNCNT,R0        ;GET NUMBER OF LAST LINE
005060  042700  177400                         BIC     #177400,R0      ;CLEAR ANY PROPAGATED BITS IN R0
005064  020300                                  CMP     R3,R0           ;REQUESTED ADD IN PRESENT SOURCE
005066  101064                                  BHI     23$             ;BR IF NO
005070  004737  004666                         JSR     PC,SBSCN        ;BUMP R1 TO NEXT PARAM (NEW CMND)
005074  121137  004664                         CMPB    (R1),NULL       ;CHECK IF NULL
005100  001451                                  BEQ     21$             ;BRANCH IF YES (ERROR)
005102  004437  004704                         JSR     R4,ICDEC        ;DECODE & CHECK NEW COMMAND
005106  005232                                  22$                     ;ERROR RETURN
005110  005722                                  TST     (R2)+           ;BUMP R2 TO 3RD WORD OF ICTBL
005112  005712                                  TST     (R2)            ;TEST IF WORD PLUS
005114  100446                                  BMI     22$             ;BR IF MINUS, NOT DEFERRED CMND
005116  005742                                  TST     -(R2)           ;DEC BACK TO 2ND WORD
005120  010146                                  MOV     R1,-(SP)        ;STORE R1 FOR REFERENCE
005122  005200                                  INC     R0              ;ADD 1 TO OLD LINE TOTAL
005124  110037  001243                         MOVB    R0,LNCNT        ;STORE IT OFF
005130  010500                                  MOV     R5,R0           ;STORE R4 FOR REFERENCE
```

```
                                                ;ROUTINE TO DETERMINE HOW FAR TO MOVE SFPTR TO ACCOMMODATE
                                                ;NEW LINE & MOVE OLD SOURCE TO MAKE ROOM
                                
005132  005205                        10$:     INC     R5              ;SCAN INPUT COMMAND, LOOK FOR
005134  105721                                 TSTB    (R1)+           ;NULL. ADD 1 TO R5 FOR LINE
005136  001375                                 BNE     10$             ;NUMBER, EACH NON-NULL CHAR.
005140  005205                                 INC     R5              ;AND ONE FOR NULL.
005142  010537  001246                         MOV     R5,SFPTR        ;STORE NEW SF LINE POINTER
005146  012601                                 MOV     (SP)+,R1        ;RECOVER R1 (COMMAND LINE PTR)
005150  114045                        11$:     MOVB    -(R0),-(R5)     ;MOVE LAST CHAR OF OLD SF TO
                                                                        ;NEW LAST CHAR LOC.
005152  001376                                 BNE     11$             ;IF NOT NULL, LOOP
005154  126003  000001                         CMPB    1(R0),R3        ;TEST IF NEXT TO LAST CHAR MOVED IS
                                                                        ;LINE NUMBER TO BE REPLACED
005160  001373                                 BNE     11$             ;MOVE NOT DONE, LOOP
005162  062700  000002                         ADD     #2,R0           ;GET R0 OFF NULL PAST LINE NUM
005166  010005                                 MOV     R0,R5           ;STORE R0 FOR REFERENCE
005170  111120                        12$:     MOVB    (R1),(R0)+      ;MOVE NEW COMMAND INTO SF
005172  105721                                 TSTB    (R1)+           ;TEST IF CHAR MOVED IS NULL
005174  001375                                 BNE     12$             ;NOT DONE STORING CMND, LOOP
005176  005203                        13$:     INC     R3              ;ADD ONE TO LINE NUMBER
005200  105725                        14$:     TSTB    (R5)+           ;TEST IF NULL
005202  001376                                 BNE     14$             ;GO UNTIL NULL
005204  020537  001246                         CMP     R5,SFPTR        ;SF RESEQUENCED, EXIT TEST
005210  001417                                 BEQ     30$             ;BR IF YES, EXIT
005212  110325                                 MOVB    R3,(R5)+        ;MOVE IN NEW LINE NUMBER
005214  000770                                 BR      13$             ;BRANCH TO NEXT LINE
005216  104400  002520                20$:     TYPE    ,BADDEC         ;TYPE BAD NUMBER ENTERED
005222  000410                                 BR      25$
005224  104400  002540                21$:     TYPE    ,IVDEDT         ;TYPE INVALID EDIT
005230  000405                                 BR      25$
005232  104400  002562                22$:     TYPE    ,IVDADD         ;TYPE INVALID NEW COMMAND
005236  000402                                 BR      25$
005240  104400  002603                23$:     TYPE    ,IVDLN          ;TYPE INVALID LINE NUMBER
005244  011404                        25$:     MOV     (R4),R4         ;ERROR RETURN
005246  000401                                 BR      35$
005250  005724                        30$:     TST     (R4)+
005252                                35$:
005252  012605                                 MOV     (SP)+,R5        ;;POP STACK INTO R5
005254  012603                                 MOV     (SP)+,R3        ;;POP STACK INTO R3
005256  012602                                 MOV     (SP)+,R2        ;;POP STACK INTO R2
005260  012601                                 MOV     (SP)+,R1        ;;POP STACK INTO R1
005262  012600                                 MOV     (SP)+,R0        ;;POP STACK INTO R0
005264  000204                                 RTS     R4


;;*****************************************************************
.SBTTL  EDIT DELETE LINE ROUTINE
;*ENTRY:         JSR     R4,EDRTE
;*       WITH R1 POINTING  TO INPUT COMMAND LINE (THE EDIT
;*       DELETE COMMAND) AND R2 POINTING TO THE SECOND
;*       WORD OF THE TABLE (IC TBC)
;*       RETURN: RTS     R4      ERROR RETURN
;*               RTS     R4+2    NO ERROR RETURN
;*THIS ROUTINE WILL REMOVE THE COMMAND DESIGNATED BY THE
;*LINE NUMBER FROM THE SOURCE FILE.  THE REMAINING
```

```
                                        ;*COMMANDS ARE MOVED UP IN THE SF AND RENUMBERED.
                                        ;*THE FOLLOWING SEQUENCE OF OPERATIONS IS PERFORMED:
                                        ;*1.  RETRIEVE LINE NUMBER FROM INPUT COMMAND
                                        ;*2.  CONVERT LN FROM ASCII TO BINARY
                                        ;*3.  CHECK IF LN EXISTS IN SF
                                        ;*4.  DELETE COMMAND FROM SF BY MOVING REST OF
                                        ;*    SF.
                                        ;*5.  DECREMENT LINE NUMBERS WHILE MOVING COMMANDS.
                                        ;*6.  ADJUST AND STORE POINTERS AN NEW LINE
                                        ;*    COUNT.
                                        ;*
                                        ;*    ROUTINES CALLED
                                        ;*      DECBIN
                                        ;*      SBSCN
                                        ;;*****************************************************************
005266                          EDRTE:
005266  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
005270  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
005272  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
005274  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
005276  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
005300  004737  004666                  JSR     PC,SBSCN        ;BUMP R1 TO NEXT PARAM (LN)
005304  105711                          TSTB    (R1)            ;CHECK IF NULL
005306  001455                          BEQ     20$             ;BR IF YES (ERROR)
005310  010146                          MOV     R1,-(SP)        ;ADDRESS OF LN ON STACK
005312  004737  031102                  JSR     PC,DECBIN       ;DECODE LN TO BINARY
005316  005450                          21$
005320  012603                          MOV     (SP)+,R3        ;STORE DECODED LINE NUMBER
005322  113702  001243                  MOVB    LNCNT,R2        ;GET NUMBER OF LAST LINE
005326  042702  177400                  BIC     #177400,R2      ;CLEAR UPPER BITS
005332  120203                          CMPB    R2,R3           ;TEST IF VALID LINE NUMBER
005334  103450                          BLO     22$             ;NO - GO PRINT ERROR
005336  013700  001710                  MOV     IBUFPT,R0       ;GET ADDRESS OF START OF SF
005342  121003                  1$:     CMPB    (R0),R3         ;TEST IF THIS IS LINE TO BE
005344  001406                          BEQ     3$              ;DELETED.  BR IF YES
005346  020037  001246                  CMP     R0,SFPTR                ;CHECK IF STILL IN SF
005352  101041                          BHI     22$             ;BR IF NO (ERROR)
005354  105720                  2$:     TSTB    (R0)+           ;TEST FOR NULL
005356  001376                          BNE     2$              ;BR IF NOT NULL, CHECK NEXT
005360  000770                          BR      1$              ;FOUND NULL, BR TO TEST FOR LN
                                        ;LINE TO BE DELETED HAS
                                        ;BEEN FOUND.  NOW FIND THE
                                        ;START OF NEXT LINE.  DEC
                                        ;THAT LINE NUMBER AND
                                        ;MOVE IT TO OVERLAY OLD CMD.
005362  010005                  3$:     MOV     R0,R5           ;STORE R0 FOR REFERENCE
005364  105720                  4$:     TSTB    (R0)+           ;LOOK FOR NEXT NULL
005366  001376                          BNE     4$              ;BR IF NOT, CHECK NEXT CHAR
005370  013702  001246                  MOV     SFPTR,R2        ;GET SOURCE FILE PTR
005374  020002                  5$:     CMP     R0,R2           ;;TEST IF END OF SF
005376  001404                          BEQ     7$              ;BR IF YES, END OF MOVE
005400  105310                          DECB    (R0)            ;DEC THAT LN
005402  112025                  6$:     MOVB    (R0)+,(R5)+     ;MOVE BYTE TO OVERLAY LINE
005404  001773                          BEQ     5$              ;TEST IF BYTE MOVED WAS MOVE
```

```
2751
2752  005406  000775                          BR      6$              ;END OF THAT LINE, CHECK IF MORE
2753  005410  010537  001246          7$:     MOV     R5,SFPTR        ;MOVE NEXT CHAR OF THIS LINE
2754  005414  105337  001243                  DECB    LNCNT           ;STORE NEW SF POINTER
2755  005420  001003                          BNE     9$              ;DEC TOTAL LINE COUNT
2756  005422  152737  000001  001234          BISB    #1,SFEMP        ;SKIP IF NOT ZERO
2757  005430  105025                  8$:     CLRB    (R5)+           ;ELSE SET SOURCE EMPTY FLAG
2758  005432  020500                          CMP     R5,R0           ;CLEAR REST OF SOURCE FILE
2759  005434  001375                          BNE     8$              ;CHECK IF FINISHED
2760                                                                  ;LOOP IF NOT DONE
2761  005436  005724                          TST     (R4)+           ;SET UP NORMAL RETURN
2762  005440  000411                          BR      30$             ;GO TO NORMAL EXIT
2763  005442  104400  002540          20$:    TYPE    ,IVDEDT         ;TYPE INVALID EDIT MESSAGE
2764  005446  000405                          BR      25$
2765  005450  104400  002520          21$:    TYPE    ,BADDEC         ;TYPE BAD DECIMAL MESSAGE
2766  005454  000402                          BR      25$
2767  005456  104400  002603          22$:    TYPE    ,IVDLN          ;TYPE INVALID LINE NUMBER MESSAGE
2768  005462  011404          25$:    MOV     (R4),R4         ;SET UP ERROR RETURN
2769  005464                  30$:
2770  005464  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
2771  005466  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
2772  005470  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
2773  005472  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
2774  005474  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
2775  005476  000204                          RTS     R4              ;RETURN TO CALLER
2776
2777
2778
2779                                  ;;*****************************************************************
2780                                  .SBTTL  PRINT LINE ROUTINE
2781                                  ;*ENTRY:        JSR     R4,PTRTE ON PLRTE
2782                                  ;*WITH R1 POINTING TO COMMAND (PRINT LINE)
2783                                  ;*RETURN:       RTS     R4      NORMAL
2784                                  ;*              RTS     R4+2    ERROR RETURN
2785                                  ;*THIS ROUTINE WILL PRINT A SINGLE LINE FROM THE
2786                                  ;*SOURCE FILE.  IF A LINE NUMBER IS GIVEN WITH THE PRINT
2787                                  ;*LINE (PL) REQUEST THE ROUTINE PRINTS THE SPECIFIED
2788                                  ;*LN.  IF LN IS NOT SUPPLIED THE LAST COMMAND
2789                                  ;*ENTERED IS PRINTED.
2790                                  ;*
2791                                  ;*ROUTINES CALLED
2792                                  ;*      TYPDS
2793                                  ;*      TYPE
2794                                  ;*      DECBIN
2795                                  ;*
2796                                  ;;*****************************************************************
2797
2798  005500                  PLRTE:
2799  005500  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
2800  005502  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
2801  005504  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
2802  005506  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
2803  005510  105737  001234                  TSTB    SFEMP           ;TEST IF ANY SOURCE CODE
2804  005514  001061                          BNE     22$             ;BRANCH IF NO SOURCE
2805  005516  004737  004666                  JSR     PC,SBSCN        ;BUMP R1 TO NEXT PARAM
2806  005522  105711                          TSTB    (R1)            ;TEST IF NULL
```

```
2807  005524  001424                        BEQ     4$              ;NO LN PRINT LAST CMND
2808  005526  010146                        MOV     R1,-(SP)        ;CONVERT ASCII LINE
2809  005530  004737  031102                JSR     PC,DECBIN       ;NUMBER TO BINARY
2810  005534  005644                        20$                     ;ERROR RETURN
2811  005536  012603                        MOV     (SP)+,R3                ;STORE CONVERTED LN
2812  005540  113705  001243                MOVB    LNCNT,R5        ;GET STORED LINE COUNT
2813  005544  042705  177400                BIC     #177400,R5      ;CLEAR UPPER BITS
2814  005550  020305                        CMP     R3,R5           ;TEST IF VALID LINE NUMBER
2815  005552  101037                        BHI     21$             ;NO - GO PRINT ERROR
2816  005554  013700  001710                MOV     IBUFPT,R0       ;GET ADDRESS OF START OF SF
2817  005560  121003                1$:     CMPB    (R0),R3         ;TEST IF THIS IS LINE TO BE
2818  005562  001413                        BEQ     3$              ;PRINTED. BR IF YES FOUND LN
2819  005564  121005                        CMPB    (R0),R5         ;CHECK IF STILL IN SF
2820  005566  101031                        BHI     21$             ;BR IF NO (ERROR-INVALID LN)
2821  005570  105720                2$:     TSTB    (R0)+           ;TEST FOR NULL
2822  005572  001376                        BNE     2$              ;BR IF NOT NULL, TEST NEXT CHAR
2823  005574  000771                        BR      1$              ;FOUND NULL, CHECK IF THIS IS LN
2824  005576  013700  001246          4$:   MOV     SFPTR,R0        ;GET SF POINTER
2825  005602  124040                        CMPB    -(R0),-(R0)     ;DEC R0 PAST NULL
2826  005604  105740                5$:     TSTB    -(R0)           ;TEST FOR NULL
2827  005606  001376                        BNE     5$              ;BR IF NO, LOOP TO FIND NULL
2828  005610  105720                        TSTB    (R0)+           ;BUMP R0 PAST NULL
2829
2830  005612  005046                3$:     CLR     -(SP)           ;CLEAR NEXT STACK WORD
2831  005614  112016                        MOVB    (R0)+,(SP)      ;PUT LINE NUMBER ON STACK
2832  005616  104404                        TYPDS                   ;TYPE LINE NUMBER
2833  005620  104400  002516                TYPE    ,EQSGN          ;TYPE EQUAL SIGN
2834  005624  010037  005632                MOV     R0,6$           ;SET UP PRINT ADDRESS
2835  005630  104400                        TYPE                    ;TYPE LINE
2836  005632  000000                6$:     .WORD
2837  005634  104400  001171                TYPE    ,$CRLF          ;TYPE CARIAGE RETURN
2838  005640  005724                        TST     (R4)+           ;SET UP NO ERROR RETURN
2839  005642  000411                        BR      30$
2840  005644  104400  002520          20$:  TYPE    ,BADDEC         ;TYPE BAD DECIMAL MESSAGE
2841  005650  000405                        BR      25$
2842  005652  104400  002603          21$:  TYPE    ,IVDLN          ;TYPE INVALID LINE NUMBER
2843  005656  000402                        BR      25$
2844  005660  104400  003015          22$:  TYPE    ,NOSRC          ;TYPE SO SOURCE MESSAGE
2845  005664  011404                25$:    MOV     (R4),R4         ;SET UP ERROR RETURN
2846  005666                        30$:
2847  005666  012605                        MOV     (SP)+,R5        ;;POP STACK INTO R5
2848  005670  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
2849  005672  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
2850  005674  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
2851  005676  000204                        RTS     R4              ;RETURN
2852
2853
2854                                ;;*****************************************************************
2855                                .SBTTL  PRINT TEST ROUTINE
2856                                ;*ENTRY:        JSR     R4,PTRTE
2857                                ;*WITH R1 POINTING TO COMMAND (PRINT TEST)
2858                                ;*RETURN:       RTS     R4      ERROR RETURN
2859                                ;*              RTS     R4+2    NORMAL RETURN
2860                                ;*THIS ROUTINE WILL PRINT THE ENTIRE CONTENTS
2861                                ;*OF THE SOURCE FILE.
2862                                ;*ROUTINES CALLED
```

```
                                ;*      TYPE
                                ;*      TYPDS
                                ;;*****************************************************************
                                PTRTE:
                                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
                                        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
                                        TSTB    SFEMP           ;TEST IF ANY SOURCE
                                        BNE     20$             ;IF NONE, BRANCH EXIT
                                        MOV     IBUFPT,R0       ;GET START OF SF
                                        MOV     SFPTR,R3        ;GET SOURCE FILE PTR
                                3$:     CLR     -(SP)           ;CLEAR NEXT STACK WORD
                                        MOVB    (R0)+,(SP)      ;PUT LINE NUM ON STACK
                                        TYPDS                   ;PRINT IT
                                        TYPE    ,EQSGN          ;TYPE EQUAL SIGN
                                        MOV     R0,1$           ;SET UP TYPE ADDRESS
                                        TYPE                    ;TYPE COMMAND
                                1$:     .WORD
                                        TYPE    ,$CRLF          ;TYPE CARRIAGE RETURN
                                2$:     TSTB    (R0)+           ;TEST FOR NULL
                                        BNE     2$              ;IF NOT LOOP UNTIL NULL
                                        CMP     R0,R3           ;TEST IF LAST LINE PRINTED
                                        BNE     3$              ;NOT DONE, LOOP
                                        TST     (R4)+           ;SET NORMAL RETURN
                                        BR      25$             ;GO TO NORMAL EXIT
                                20$:    TYPE    ,NOSRC          ;TYPE NO SOURCE
                                        MOV     (R4),R4         ;SET ERROR RETURN
                                25$:
                                        MOV     (SP)+,R3        ;;POP STACK INTO R3
                                        MOV     (SP)+,R0        ;;POP STACK INTO R0
                                        RTS     R4              ;RETURN


                                ;;*****************************************************************
                                .SBTTL  FORMAT SELECT ROUTINE
                                ;*ENTRY:        JSR     R4,FTRTE
                                ;*                              WITH R1 POINTING TO COMMAND
                                ;*
                                ;*RETURN:       RTS     R4      ERROR RETURN
                                ;*              RTS     R4+2    NO ERROR RETURN
                                ;*
                                ;*THIS ROUTINE WILL SELECT THE FORMAT (24 OR 26,OCTAL) THAT IS TO
                                ;*BE USED FOR THE SUBSYSTEM COMMANDS. ALL COMMANDS WILL BE EXECUTED
                                ;*WITH THIS FORMAT UNTIL IT IS CHANGED.
                                ;;*****************************************************************
                                FTRTE:  JSR     PC,SBSCN        ;BUMP TO NEXT PARAMETER
                                        TSTB    (R1)            ;TEST IF NULL
                                        BEQ     2$              ;NO CHANGE - EXIT
                                        CMPB    (R1),#'?        ;TEST IF QUESTION
                                        BNE     1$              ;NO - SKIP PRINT
                                        MOV     FORMAT,-(SP)    ;ELSE GET FORMAT STORED
                                        TYPOC                   ;PRINT IT
                                        TYPE    ,$CRLF
                                        BR      2$              ;GO TO EXIT
                                1$:     MOV     R1,-(SP)        ;PUT VALUE FOR CONVERSION ON STACK
```

```
 2919  006034  004737  030746                 JSR     PC,OCTBIN
 2920  006040  006052                         20$                     ;ERROR RETURN
 2921  006042  012637  001212                 MOV     (SP)+,FORMAT    ;STORE NEW FORMAT
 2922  006046  005724                 2$:     TST     (R4)+           ;GOOD RETURN
 2923  006050  000403                         BR      30$
 2924  006052  011404                 20$:    MOV     (R4),R4         ;SET ERROR RETURN
 2925  006054  104400  002646                 TYPE    ,BADOCT         ;PRINT ERROR MESSAGE
 2926  006060  000204                 30$:    RTS     R4              ;RETURN
 2927
 2928                                 ;;*****************************************************************
 2929                                 .SBTTL  TIMEOUT CHANGE ROUTINE
 2930                                 ;*ENTRY:        JSR     R4,TORTE
 2931                                 ;*                      WITH R1 POINTING TO COMMAND
 2932                                 ;*
 2933                                 ;*RETURN        RTS     R4      ERROR RETURN
 2934                                 ;*              RTS     R4+2    NO ERROR RETURN
 2935                                 ;*
 2936                                 ;*THIS ROUTINE WILL CHANGE THE TIME OUT DELAY FOR SUBSYSTEM OPERATIONS.
 2937                                 ;;*****************************************************************
 2938  006062  004737  004666         TORTE:  JSR     PC,SBSCN                ;BUMP TO PARAMETER
 2939  006066  105711                         TSTB    (R1)                    ;TEST IF 0
 2940  006070  001417                         BEQ     1$                      ;EXIT - NO TIME OUT CHANGE
 2941  006072  121127  000077                 CMPB    (R1),#'?                ;TEST IF QUESTION
 2942  006076  001006                         BNE     2$                      ;NO - SKIP TYPE
 2943  006100  013746  001260                 MOV     TOVAL,-(SP)             ;GET TOVAL TO STACK
 2944  006104  104404                         TYPDS                           ;TYPE IT
 2945  006106  104400  001171                 TYPE    ,$CRLF
 2945  006112  000406                         BR      1$
 29    006114  010146                 2$:     MOV     R1,-(SP)                ;GET VALUE FOR CONVERSION
 2948  006116  004737  031102                 JSR     PC,DECBIN               ;CONVERT VALUE TO BINARY
 2949  006122  006134                         20$                             ;ERROR RETURN
 2950  006124  012637  001260                 MOV     (SP)+,TOVAL             ;STORE VALUE
 2951  006130  005724                 1$:     TST     (R4)+                   ;SET GOOD RETURN
 2952  006132  000403                         BR      30$
 2953  006134  011404                 20$:    MOV     (R4),R4                 ;SET ERROR RETURN
 2954  006136  104400  002520                 TYPE    ,BADDEC                 ;TYPE MESSAGE
 2955  006142  000204                 30$:    RTS     R4                      ;RETURN
 2956                                 ;;***************************************************** ******
 2957                                 .SBTTL  DRIVE NUMBER CHANGE ROUTINE
 2958                                 ;*ENTRY:        JSR     R4,DNRTE
 2959                                 ;*      WITH R1 POINTING TO THE COMMAND FIELD IF DN
 2960                                 ;*      COMMAND OR SUBSYSTEM COMMAND PARAMETER IF
 2961                                 ;*      SYBSYSTEM FUNCTION COMMAND
 2962                                 ;*ERROR RETURN:         MOV     (R4),R4
 2963                                 ;*                      RTS     R4
 2964                                 ;*NO ERROR RETURN:      TST     (R4)+
 2965                                 ;*                      RTS     R4
 2966                                 ;*
 2967                                 ;*THIS ROUTINE CHANGES THE "DRIVE" PARAMETER BY STORING
 2968                                 ;*IN IT THE VALUE SPECIFIED BY THE "DN" OR "SF"
 2969                                 ;*COMMANDS NOTE R1 MUST POINT TO THE COMMAND
 2970                                 ;*FIELD (DN) OR THE SUBSYSTEM COMMAND PARAMETER (SF).
 2971                                 ;*
 2972                                 ;*ROUTINES CALLED
 2973                                 ;*OCTBIN
 2974                                 ;*
```

```
                                        ;*SBSCN
                                        
006144                                  DNRTE:
006144  010146                                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
006146  004737  004666                          JSR     PC,SBSCN        ;BUMP R1 TO NEXT PARAM
006152  105711                                  TSTB    (R1)            ;TEST IF PARAM NULL
006154  001425                                  BEQ     1$              ;NO DRIVE CHANGE EXIT
006156  121127  000077                          CMPB    (R1),#'?        ;TEST IF QUESTION
006162  001006                                  BNE     4$              ;NO - SKIP TYPE
006164  013746  001176                          MOV     DRIVE,-(SP)     ;GET DRIVE NUM TO STACK
006170  104401                                  TYPOC                   ;TYPE IT
006172  104400  001171                          TYPE    ,$CRLF
006176  000414                                  BR      1$
006200  121127  000054                  4$:     CMPB    (R1),#',        ;TEST IF NULL ENTRY
006204  001411                                  BEQ     1$              ;IF YES NO DRIVE CHANGE EXIT
006206  010146                                  MOV     R1,-(SP)        ;ADDRESS OF ASCII CHAR
006210  004737  030746                          JSR     PC,OCTBIN       ;CONVERT TO BINARY
006214  006234                                  2$                      ;ERROR RETURN
006216  011637  001176                          MOV     (SP),DRIVE      ;STORE NEW DRIVE NUMBER
006222  022627  000007                          CMP     (SP)+,#7                ;TEST IF VALID DRIVE
006226  101005                                  BHI     3$              ;YES - TYPE BAD DRIVE NUM
006230  005724                          1$:     TST     (R4)+           ;SET NORMAL RETURN
006232  000406                                  BR      30$
006234  104400  002646                  2$:     TYPE    ,BADOCT         ;TYPE INVALID OCTAL NUMBER
006240  000402                                  BR      29$
006242  104400  003161                  3$:     TYPE    ,BADDRV         ;PRINT MESSAGE
006246  011404                          29$:    MOV     (R4),R4         ;SET ERROR RETURN
006250                                  30$:
006250  012601                                  MOV     (SP)+,R1        ;;POP STACK INTO R1
006252  000204                                  RTS     R4              ;RETURN

                                        ;;******************************************************************
                                        ;SBTTL  ITERATION COUNT CHANGE ROUTINE
                                        ;*ENTRY:        JSR     R4,ICRTE
                                        ;*                      WITH R1 POINTING TO INPUT COMMAND
                                        ;*RETURN:       RTS     R4      ERROR RETURN
                                        ;*              RTS     R4+2    NO ERROR RETURN
                                        ;*
                                        ;*      ROUTINES CALLED:
                                        ;*      DECBIN
                                        ;*      SBSCN
                                        ;;******************************************************************

006254  004737  004666          ICRTE:  JSR     PC,SBSCN        ;BUMP R1 TO NEXT PARAM (IC)
006260  105711                          TSTB    (R1)            ;TEST FOR NULL
006262  001417                          BEQ     1$              ;IF NULL, EXIT. NO CHANGE IC
006264  121127  000077                  CMPB    (R1),#'?        ;TEST IF QUESTION
006270  001006                          BNE     2$              ;NO - SKIP TYPE
006272  013746  001232                  MOV     ITCNT,-(SP)     ;GET ITERATION CNT TO STACK
006276  104404                          TYPDS                   ;TYPE IT
006300  104400  001171                  TYPE    ,$CRLF
006304  000406                          BR      1$
006306  010146                  2$:     MOV     R1,-(SP)        ;ADDRESS OF PARAM ON STACK
006310  004737  031102                  JSR     PC,DECBIN       ;CONVERT IT TO BINARY
006314  006326                          20$                     ;ERROR RETURN
006316  012637  001232                  MOV     (SP)+,ITCNT     ;STORE ITERATION COUNT
```

```
3031  006322  005724                    1$:     TST     (R4)+           ;SET UP NORMAL RETURN
3032  006324  000403                            BR      3$
3033  006326  011404                    2$:     MOV     (R4),R4         ;SET UP ERROR RETURN
3034  006330  104400  002520                    TYPE    ,BADDEC         ;TYPE BAD DECIMAL MESSAGE
3035  006334  000204                    3$:     RTS     R4              ;RETURN
3036
3037                                    ;;*****************************************************************
3038                                    .SBTTL  SPECIAL DATA PATTERN ROUTINE
3039                                    ;*ENTRY:        JSR     R4,DPRTE
3040                                    ;*              WITH R1 POINTING TO THE INPUT COMMAND
3041                                    ;*RETURN:       RTS     R4      ERROR RETURN
3042                                    ;*              RTS     R4+2    NO ERROR ROUTINE
3043                                    ;*
3044                                    ;*THIS ROUTINE WILL ACCEPT 32 WORDS OR LESS AND STORE
3045                                    ;*THEM WITH THE IDENTIFIER X,Y, OR Z AS CHOSEN BY
3046                                    ;*PARAMETER 1.  INPUT DATA MUST BE IN OCTAL AND IN
3047                                    ;*WORD FORMAT (6 OCTAL CHARACTERS WITH THE UPPER BIT A 0).
3048                                    ;*CARRIAGE RETURNS MAY BE USED TO TERMINATE A LINE
3049                                    ;*WITHOUT AFFECTING THE INPUT DATA BUT IT MUST OCCUR
3050                                    ;*ON A WORD BOUNDARY.  (IF NOT ON WORD BOUNDARY THE
3051                                    ;*REMAINDER OF THE WORD IS ZERO FILLED.) THE
3052                                    ;*INPUT DATA IS TERMINATED WITH A CARRIAGE RETURN
3053                                    ;*AT THE BEGINNING OF A LINE.  IF LESS THAN 32
3054                                    ;*WORDS ARE ENTERED THE REMAINDER WORDS ARE
3055                                    ;*ZERO FILLED.
3056                                    ;*      ROUTINES CALLED:
3057                                    ;*              OCTBIN
3058                                    ;*              SBSCN
3059                                    ;;*****************************************************************
3060  006336  000000  000000  000000    CVTBUF: .WORD   0,0,0,0
3061  006344  000000
3062  006346  052737  100000  001702    EBRTE:  BIS     #BIT15,TEMP1            ;SET FLAG FOR EDIT BUFFER
3063  006354  000402                            BR      DPRTE1          ;SKIP
3064  006356  005037  001702            DPRTE:  CLR     TEMP1           ;CLEAR EDIT BUFFER FLAG
3065  006362                            DPRTE1:
3066  006362  010046                            MOV     R0,-(SP)        ;;PUSH R0 ON STACK
3067  006364  010146                            MOV     R1,-(SP)        ;;PUSH R1 ON STACK
3068  006366  010246                            MOV     R2,-(SP)        ;;PUSH R2 ON STACK
3069  006370  010346                            MOV     R3,-(SP)        ;;PUSH R3 ON STACK
3070  006372  010546                            MOV     R5,-(SP)        ;;PUSH R5 ON STACK
3071  006374  010446                            MOV     R4,-(SP)        ;;PUSH R4 ON STACK
3072  006376  004737  004666                    JSR     PC,SBSCN        ;BUMP R1 TO NEXT PARAM (PAT NAME)
3073  006402  105711                            TSTB    (R1)            ;TEST PARAM NULL
3074  006404  001540                            BEQ     2$              ;CANNOT ACCEPT NULL, BRANCH TO ERROR
3075
3076  006406  121127  000130                    CMPB    (R1),#'X        ;TEST PATTERN NAME FOR
3077  006412  001006                            BNE     1$              ;X, Y, OR Z. SET UP R3
3078  006414  012703  001262                    MOV     #PATX,R3        ;WITH ADDRESS OF AREA
3079  006420  152737  000377  001237            BISB    #377,PATXDF     ;SET PAT X DEFINED SWITCH
3080  006426  000421                            BR      3$              ;TO STORE DATA PATTERN
3081  006430  121127  000131          1$:     CMPB    (R1),#'Y
3082  006434  001006                            BNE     2$
3083  006436  012703  001362                    MOV     #PATY,R3
3084  006442  152737  000377  001240            BISB    #377,PATYDF     ;SET PAT Y DEFINED SWITCH
3085  006450  000410                            BR      3$
3086  006452  121127  000132          2$:     CMPB    (R1),#'Z
```

```
3087  006456  001113                        BNE     20$             ;INVALID PATTERN NAME, ERROR
3088  006460  012703  001462                MOV     #PATZ,R3
3089  006464  152737  000377  001241        BISB    #377,PATZDF     ;SET PAT Z DEFINED SWITCH
3090  006472  010300                3$:     MOV     R3,R0           ;STORE R3 (BUFFER AREA)
3091  006474  005737  001702                TST     TEMP1           ;TEST EDIT FLAG
3092  006500  100016                        BPL     39$             ;IF SET - SKIP
3093  006502  004737  004666                JSR     PC,SBSCN        ;ELSE BUMP TO NEXT PARAM, WORD NUM
3094  006506  010146                        MOV     R1,-(SP)        ;GET ADDRESS OF PARAM
3095  006510  004737  030746                JSR     PC,OCTBIN       ;CONVERT IT TO OCTAL
3096  006514  006706                        20$                     ;ERROR RETURN
3097  006516  011602                        MOV     (SP),R2         ;GET WORD NUMBER FOR START OF EDIT
3098  006520  006302                        ASL     R2              ;NOW ITS WORD ADDRESS
3099  006522  060200                        ADD     R2,R0           ;SET TO INDEX INTO BUFFER
3100  006524  012702  000040                MOV     #↑D32,R2        ;SET MAX OF EDIT
3101  006530  162602                        SUB     (SP)+,R2        ;NOW SET TO REMAINDER OF BUFFER LENGTH
3102  006532  100465                        BMI     20$             :ERROR, EDIT IS OUT OF BOUNDS
3103  006534  000407                        BR      45$
3104  006536  012704  000040          39$:  MOV     #↑D32,R4        ;SET R4 FOR COUNT
3105  006542  005023                44$:    CLR     (R3)+           ;CLEAR PATTERN STORAGE
3106  006544  005304                        DEC     R4              ;AREA, 32 WORDS
3107  006546  001375                        BNE     44$             ;LOOP
3108
3109  006550  012702  000040                MOV     #↑D32,R2        ;SET TOTAL WORD COUNT
3110                                        ;THE REGISTERS USAGE FOR THE REMAINDER IS AS FOLLOWS:
3111                                        ; R0    POINTS TO THE PATTERN STORAGE AREA
3112                                        ; R1    POINTS TO THE ASCII INPUT DATA
3113                                        ; R3    USED AS A SWITCH, SET WHEN NULL (CARRIAGE
3114                                        ;       RETURN IS DETECTED.
3115                                        ; R4    COUNTER FOR TICKING OFF 6 ASCII INPUT
3116                                        ;       CHARACTERS (ONE WORD)
3117                                        ; R5    POINTS TO THE TEMPORARY CONVERSION BUFFER (CVTBUF)
3118                                        ; R2    COUNTER TO LIMIT INPUT TO 32 WORDS. ALSO
3119                                        ;       USED TO FORCE EXIT IF TWO CONSECUTIVE
3120                                        ;       CARRIAGE RETURNS ARE TYPED.
3121
3122
3123  006554  005003                45$:    CLR     R3              ;RESET CR SWITCH
3124  006556  004737  004666                JSR     PC,SBSCN        ;BUMP R1 TO NEXT PARAM (DATA)
3125  006562  012705  006336          4$:   MOV     #CVTBUF,R5      ;SET UP CVTBUF POINTER
3126  006566  012704  000006                MOV     #6,R4           ;SET UP CONVERSION COUNT
3127  006572  112125                5$:    MOVB    (R1)+,(R5)+     ;MOVE ASCII CHAR TO CVTBUF. IF
3128  006574  001404                        BEQ     6$              ;0 (NULL) EXIT LOOP CLEAR CR SWITCH IF
3129  006576  005003                        CLR     R3              ;SET (NON-NULL CHAR TYPED). DEC CONVERT
3130  006600  005304                        DEC     R4              ;COUNT AND
3131  006602  001373                        BNE     5$              ;LOOP. IF ONE WORD READY,
3132  006604  000415                        BR      10$             ;BRANCH TO CONVERSION
3133  006606  005703                6$:    TST     R3              ;TEST IF CR SWITCH SET. IF NOT
3134  006610  001402                        BEQ     7$              ;SET CR SWITCH. IF SET, CLEAR R2
3135  006612  005002                        CLR     R2              ;(TOTAL WORD COUNT) TO PREPARE FOR
3136  006614  000401                        BR      8$              ;EXIT
3137  006616  005103                7$:    COM     R3              ;SETTING CR SWITCH FOR ABOVE
3138  006620  020427  000006          8$:   CMP     R4,#6           ;TEST IF PARTIAL WORD TYPED. IF
3139  006624  001414                        BEQ     11$             ;YES, 0 FILL REST OF WORD. GO TO CONVERT
3140  006626  005305                        DEC     R5              ;AND PLACE IN PAT STORE. IF NOT, GO TO
3141  006630  112725  000060          9$:   MOVB    #'0,(R5)+       ;CHECK IF DONE.
3142  006634  005304                        DEC     R4
```

```
3143  006636  001374                        BNE     9$
3144  006640  012746  006336        10$:    MOV     #CVTBUF,-(SP)   ;START OF CONVERT. RESET CVTBUF PTR
3145  006644  004737  030746                JSR     PC,OCTBIN       ;CALL CONVERSION
3146  006650  006714                        21$                     ;CONVERSION ERROR ROUTINE
3147  006652  012620                        MOV     (SP)+,(R0)+     ;STORE CONVERTED VALUE IN PAT STORE
3148  006654  005302                        DEC     R2              ;DEC TOTAL WORD COUNTER
3149  006656  005702                11$:    TST     R2              ;TEST IF WORD CNTR 0.
3150  006660  001407                        BEQ     12$             ;EXIT IF YES
3151  006662  005703                        TST     R3              ;TEST CARRIAGE RETURN SWITCH
3152  006664  001736                        BEQ     4$              ;IF NOT SET, GET NEXT 6 CHAR. ELSE
3153  006666  104400  003531                TYPE    ,SPACE6         ;TYPE 6 SPACES TO ALIGN DATA INPUT
3154  006672  104410                        RDLIN                   ;READ NEXT INPUT LINE
3155  006674  012601                        MOV     (SP)+,R1        ;GET ADDRESS OF INPUT
3156  006676  000731                        BR      4$              ;LOOP TO PROCESS NEW LINE
3157  006700  012604                12$:    MOV     (SP)+,R4        ;RESTORE R4 FOR RETURN
3158  006702  005724                        TST     (R4)+           ;NO ERROR RETURN
3159  006704  000407                        BR      30$
3160  006706  104400  002624        20$:    TYPE    ,IVDPAR         ;TYPE INVALID PARAMETER
3161  006712  000402                        BR      25$
3162  006714  104400  002646        21$:    TYPE    ,BADOCT         ;TYPE BAD OCTAL CHARACTERS
3163  006720  012604                25$:    MOV     (SP)+,R4        ;RESTORE R4 FOR RETURN
3164  006722  011404                        MOV     (R4),R4         ;ERROR RETURN
3165  006724                        30$:
3166  006724  012605                        MOV     (SP)+,R5        ;;POP STACK INTO R5
3167  006726  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
3168  006730  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
3169  006732  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
3170  006734  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
3171  006736  000204                        RTS     R4              ;RETURN
3172
3173                          ;;*****************************************************************
3174                          .SBTTL  PRINT REGISTER ROUTINE
3175                          ;*      ENTRY:  JSR     R4,PRRTE
3176                          ;*                      WITH R1 POINTING TO INPUT COMMAND
3177                          ;*      RETURN: RTS     R4      ERROR RETURN
3178                          ;*              RTS     R4+2    NO ERROR RETURN
3179                          ;*
3180                          ;*THIS ROUTINE WILL READ THE RK611 UNIBUS VISIBLE
3181                          ;*REGISTER AND PRINT THE VALUE (OCTAL) ON THE
3182                          ;*TERMINAL. THE REGISTER MAY BE SPECIFIED IN OCTAL
3183                          ;*FROM 00 TO 17 (NUMBER 13 RESERVED FOR FUTURE USE)
3184                          ;*OR NMEMONICALLY AS:
3185                          ;*      NUMBER  NAME    DESCRIPTION
3186                          ;*      0       CS1     COMMAND STATUS REGISTER 1
3187                          ;*      01      WC      WORD COUNT
3188                          ;*      02      BA      BUFFER ADDRESS
3189                          ;*      03      DA      DESIRED ADDRESS - TRACK AND SECTOR
3190                          ;*      04      CS2     COMMAND STATUS REGISTER 2
3191                          ;*      05      DS      DRIVE STATUS
3192                          ;*      06      ER      ERROR REGISTER
3193                          ;*      07      ASOF    ATTENTION SUMMARY AND OFFSET REGISTER
3194                          ;*      10      DC      DESIRED CYLINDER
3195                          ;*      12      DB      DATA BUFFER
3196                          ;*      13      MR1     MAINTENANCE REGISTER 1
3197                          ;*      14      POS     ECC POSITION REGISTER
3198                          ;*      15      PAT     ECC PATTERN REGISTER
```

```
                        ;*      16      MR2     MAINTENANCE REGISTER 2
                        ;*      17      MR3     MAINTENANCE REGISTER 3
                        ;*ROUTINES CALLED
                        ;*      TYPOC
                        ;*      SBSCN
                        ;;*************************************************************
                        PRRTE:
                                MOV     R3,-(SP)        ;;PUSH R3 ON STACK
                                JSR     PC,SBSCN        ;BUMP R1 TO NEXT PARAM (RN)
                                TSTB    (R1)            ;TEST IF PARAM NULL
                                BEQ     1$              ;NO REG SPECIFIED, USED LAST REG SELECTED
                                CMPB    (R1),#67        ;TEST IF NUMERIC PARAMETER
                                BLOS    4$              ;YES - SKIP
                                JSR     PC,RGDCDE       ;GO DECODE REGISTER
                                21$                     ;ERROR RETURN
                                BR      3$
                        4$:     MOV     R1,-(SP)        ;GET ADDRESS OF STRING FOR CNVERSION
                                JSR     PC,OCTBIN       ;CONVERT IT
                                20$                     ;ERROR RETURN
                                MOV     (SP)+,R3        ;GET CONVERTED NUMBER
                        3$:     MOV     R3,REGNUM       ;STORE INTEGER
                        1$:     MOV     REGNUM,R3       ;
                        2$:
                                ASL     R3              ;SHIFT R3, MULTIPLY INDEX BY 2
                                ADD     RKBAS,R3        ;COMPUTE RK611 ADDRESS
                                MOV     (R3),-(SP)      ;PUT SELECTED REGISTER CONTENTS ON STACK
                                TYPOC                   ;TYPE REGISTER CONTENTS
                                TYPE    ,$CRLF          ;TYPE CARRIAGE RETURN & LINE FEED
                                TST     (R4)+           ;SET UP NO ERROR RETURN
                                BR      30$
                        20$:    TYPE    ,BADOCT         ;TYPE BAD OCTAL MESSAGE
                                BR      25$             ;GO TO EXIT
                        21$:    TYPE    ,BADSEL         ;TYPE BAD REGISTER SELECTION
                        25$:    MOV     (R4),R4         ;SET UP ERROR RETURN
                        30$:
                                MOV     (SP)+,R3        ;;POP STACK INTO R3
                                RTS     R4              ;RETURN

                        ;;*************************************************************
                        .SBTTL  CONVERT REGISTER NAME (ASCII) TO REGISTER NUMBER (OCTAL)
                        ;*ENTRY:        JSR     PC,RGDCDE
                        ;*                      WITH R1 POINTING TO THE REG NAME
                        ;*RETURN:       RTS     PC      NORMAL RETURN
                        ;*              RTS     PC+2    ERROR RETURN
                        ;*                      WITH R3 CONTAINING THE REG NUMBER
                        ;*
                        ;*THE ASCII NAME OF THE REGISTER IS DECODED INTO AN OCTAL VALUE
                        ;*REQUIRED TO SELECT THE REGISTER. THIS OCTAL VALUE IS PLACED
                        ;*IN R3.
                        ;;*************************************************************
                        RGDCDE: CMPB    (R1),#'A        ;TEST IF FIRST CHAR IS A
                                BNE     1$              ;NO - SKIP
                                MOV     #7,R3           ;SET FOR ASOF
                                BR      40$             ;GO TO EXIT
                        1$:     CMPB    (R1),#'B        ;TEST IF B
```

```
3255  007074  001003                        BNE     2$
3256  007076  012703  000002                MOV     #2,R3           ;SET FOR BA
3257  007102  000525                        BR      40$             ;GO TO EXIT
3258  007104  121127  000103        2$:     CMPB    (R1),#'C        ;TEST IF C
3259  007110  001012                        BNE     5$
3260  007112  062701  000002                ADD     #2,R1           ;BUMP R1 TO 3RD CHAR
3261  007116  121127  000061                CMPB    (R1),#'1                ;TEST IF THIRD CHAR IS 1
3262  007122  001002                        BNE     3$              ;NO - BRANCH
3263  007124  005003                        CLR     R3              ;SET FOR CS1
3264  007126  000513                        BR      40$
3265  007130  012703  000004        3$:     MOV     #4,R3           ;SET FOR CS2
3266  007134  000510                        BR      40$
3267  007136  121127  000104        5$:     CMPB    (R1),#'D        ;TEST IF D
3268  007142  001026                        BNE     9$              ;NO - SKIP
3269  007144  005201                        INC     R1              ;BUMP TO 2ND CHAR
3270  007146  121127  000101                CMPB    (R1),#'A        ;TEST 2ND CHAR A
3271  007152  001003                        BNE     6$
3272  007154  012703  000003                MOV     #3,R3           ;SET FOR DA
3273  007160  000476                        BR      40$             ;EXIT
3274  007162  121127  000123        6$:     CMPB    (R1),#'S        ;TEST IF 2ND CHAR S
3275  007166  001003                        BNE     7$
3276  007170  012703  000005                MOV     #5,R3           ;SET FOR DS
3277  007174  000470                        BR      40$
3278  007176  121127  000102        7$:     CMPB    (R1),#'B        ;TEST IF 2ND CHAR B
3279  007202  001003                        BNE     8$
3280  007204  012703  000012                MOV     #12,R3          ;SET FOR DB
3281  007210  000462                        BR      40$
3282  007212  012703  000010        8$:     MOV     #10,R3          ;SET FOR DC
3283  007216  000457                        BR      40$
3284  007220  121127  000105        9$:     CMPB    (R1),#'E        ;TEST IF E
3285  007224  001003                        BNE     10$
3286  007226  012703  000006                MOV     #6,R3           ;SET FOR ER
3287  007232  000451                        BR      40$
3288  007234  121127  000115        10$:    CMPB    (R1),#'M        ;TEST IF M
3289  007240  001021                        BNE     13$
3290  007242  062701  000002                ADD     #2,R1           ;BUMP R1 TO 3RD CHAR
3291  007246  121127  000061                CMPB    (R1),#'1                ;TEST IF 3RD CHAR 1
3292  007252  001003                        BNE     11$
3293  007254  012703  000013                MOV     #13,R3          ;SET FOR MR1
3294  007260  000436                        BR      40$
3295  007262  121127  000062        11$:    CMPB    (R1),#'2                ;TEST IF 3RD CHAR 2
3296  007266  001003                        BNE     12$
3297  007270  012703  000016                MOV     #16,R3          ;SET FOR MR2
3298  007274  000430                        BR      40$
3299  007276  012703  000017        12$:    MOV     #17,R3          ;SET FOR MR3
3300  007302  000425                        BR      40$
3301  007304  121127  000120        13$:    CMPB    (R1),#'P        ;TEST IF P
3302  007310  001012                        BNE     15$
3303  007312  005201                        INC     R1              ;BUMP R1 TO 2ND CHAR
3304  007314  121127  000117                CMPB    (R1),#'O        ;TEST 2ND CHAR O
3305  007320  001003                        BNE     14$
3306  007322  012703  000014                MOV     #14,R3          ;SET FOR POS
3307  007326  000413                        BR      40$
3308  007330  012703  000015        14$:    MOV     #15,R3          ;SET FOR PAT
3309  007334  000410                        BR      40$
3310  007336  121127  000127        15$:    CMPB    (R1),#'W                ;TEST IF W
```

```
3311  007342  001003                        BNE     20$             ;BR TO ERROR EXIT
3312  007344  012703  000001                MOV     #1,R3           ;SET FOR WC
3313  007350  000402                        BR      40$
3314  007352  013616                20$:    MOV     @(SP)+,(SP)     ;SET ERROR RETURN
3315  007354  000207                        RTS     PC
3316  007356  062716  000002        40$:    ADD     #2,(SP)         ;SET FOR GOOD RETURN
3317  007362  000207                        RTS     PC
3318                                ;;*****************************************************************
3319                                .SBTTL  HELP PRINTOUT ROUTINE
3320                                ;*      ENTRY:  JSR     R4,HPRTE
3321                                ;*      RETURN: RTS     R4
3322                                ;*
3323                                ;*THIS ROUTINE PRINTS A SUMMARY OF THE COMMANDS
3324                                ;*AND PARAMETERS AVAILABLE TO THE USER.
3325                                ;* ROUTINES CALLED:
3326                                ;*      TYPE
3327                                ;*
3328                                ;;*****************************************************************
3329
3330  007364  105737  001672        HPRTE:  TSTB    HPVLD           ;TEST IF HELP FILE VALID
3331  007370  001004                        BNE     1$
3332  007372  104400  002664                TYPE    ,HPFILE         ;
3333  007376  005724                2$:     TST     (R4)+           ;GOOD RETURN
3334  007400  000204                        RTS     R4              ;RETURN
3335  007402  104400  034436        1$:     TYPE    ,HPDATA         ;TYPE HELP FILE
3336  007406  000773                        BR      2$
3337
3338                                ;;*****************************************************************
3339                                .SBTTL  NEW TEST ROUTINE
3340                                ;*      ENTRY   JSR     R4,NTRTE
3341                                ;*      RETURN: JMP     COMLEV  (RETURN TO COMMAND LEVEL)
3342                                ;*
3343                                ;*THIS ROUTINE CLEARS THE SOURCE AND OBJECT FILES
3344                                ;*AND TERMINATES ANY TEST PRESENTLY EXECUTING. ALL
3345                                ;*STORED TEST PARAMETERS AND THE OUTPUT BUFFER
3346                                ;*ARE LEFT UNCHANGED. SINCE THE INPUT BUFFER
3347                                ;*AND THE SOURCE FILE IS THE SAME MEMORY, THE
3348                                ;*INPUT BUFFER IS LOST AND MUST BE REINITIALIZED.
3349                                ;*
3350                                ;* ROUTINES CALLED
3351                                ;*      NONE
3352                                ;;*****************************************************************
3353
3354  007410                        NTRTE:
3355  007410  010346                        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
3356  007412  010546                        MOV     R5,-(SP)        ;;PUSH R5 ON STACK
3357  007414  152737  000377  001234        BISB    #377,SFEMP      ;SET SOURCE FILE EMPTY
3358  007422  105037  001235                CLRB    VLDOBJ          ;CLEAR OBJECT VALID
3359  007426  012737  034434  001712        MOV     #OFILE,OFPTR    ;RESET OBJECT FILE POINTER
3360  007434  013737  001710  001246        MOV     IBUFPT,SFPTR    ;RESET SOURCR FILE POINTER
3361  007442  105037  001116                CLRB    LINNUM          ;CLEAR LINE NUMBER
3362  007446  105037  001243                CLRB    LNCNT           ;CLEAR LINE COUNT
3363  007452  013703  001246                MOV     SFPTR,R3
3364
3365  007456  013705  001700                MOV     MAXWDS,R5
3366  007462  005023                1$:     CLR     (R3)+           ;SET UP REGISTERS
```

```
 3367  007464  005305                        DEC     R5              ;AND CLEAR SOURCE
 3368  007466  001375                        BNE     1$              ;FILE
 3369  007470  013705  001674                MOV     OBJSZE,R5
 3370  007474  013703  001712                MOV     OFPTR,R3
 3371  007500  005023              2$:       CLR     (R3)+           ;SET UP REGISTERS AND
 3372  007502  005305                        DEC     R5              ;CLEAR OBJECT FILE
 3373  007504  001375                        BNE     2$
 3374  007506  012605                        MOV     (SP)+,R5        ;;POP STACK INTO R5
 3375  007510  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
 3376  007512  012706  001100                MOV     #1100,SP        ;CLEAN OFF STACK
 3377  007516  000137  004416                JMP     COMLEV          ;GO TO COMMAND LEVEL
 3378                                  ;;*****************************************************************
 3379                                  ;SBTTL  BUFFER DUMP ROUTINE
 3380                                  ;*      ENTRY:  JSR     R4,BDRTE
 3381                                  ;*      RETURN: RTS     R4
 3382                                  ;*THIS ROUTINE WILL DUMP THE READ, WRITE, HEADER, OR SPECIAL BUFFER. THE
 3383                                  ;*NUMBER OF WORDS DUMPED IS GIVEN AS A PARAMETER. IF THE NUMBER OF
 3384                                  ;*WORDS IS NOT GIVEN THE LAST SPECIFIED WORD COUNT IS USED IN THE CASE
 3385                                  ;*OF THE READ OR WRITE BUFFER OR 32 IN THE CASE OF A SPECIAL DATA BUFFER.
 3386                                  ;;*****************************************************************
 3387  007522                        BDRTE:
 3388  007522  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 3389  007524  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 3390  007526  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
 3391  007530  010346                        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
 3392  007532  005002                        CLR     R2              ;CLEAR FOR POSSIBLE WORD COUNT
 3393  007534  004737  004666                JSR     PC,SBSCN                ;GET BUFFER PARAMETER
 3394  007540  105711                        TSTB    (R1)                    ;TEST IF IT IS NULL
 3395  007542  001522                        BEQ     21$                     ;YES - SKIP TO ERROR EXIT
 3396  007544  121127  000122                CMPB    (R1),#'R        ;TEST IF READ BUFFER
 3397  007550  001003                        BNE     1$              ;NO - SKIP
 3398  007552  013703  001710                MOV     IBUFPT,R3       ;ELSE GET ADDRESS OF READ BUFFER
 3399  007556  000441                        BR      5$              ;GO DO IT
 3400  007560  121127  000127        1$:     CMPB    (R1),#'W        ;TEST IF WRITE BUFFER
 3401  007564  001003                        BNE     2$              ;NO - SKIP
 3402  007566  013703  001706                MOV     OBUFPT,R3       ;ELSE GET ADDRESS OF WRITE BUFFER
 3403  007572  000433                        BR      5$              ;GO DO IT
 3404  007574  012702  000040        2$:     MOV     #40,R2          ;SET WORD COUNT FOR SPEC BUFF
 3405  007600  121127  000130                CMPB    (R1),#'X        ;TEST IF SPECIAL BUFFER X
 3406  007604  001003                        BNE     3$              ;NO - SKIP
 3407  007606  012703  001262                MOV     #PATX,R3        ;ELSE GET ADDRESS OF BUFFER X
 3408  007612  000423                        BR      5$              ;GO DO IT
 3409  007614  121127  000131        3$:     CMPB    (R1),#'Y        ;TEST IF BUFFER Y
 3410  007620  001003                        BNE     4$              ;NO - SKIP
 3411  007622  012703  001362                MOV     #PATY,R3        ;ELSE GET ADDRESS OF BUFFER Y
 3412  007626  000415                        BR      5$              ;GO DO IT
 3413  007630  121127  000132        4$:     CMPB    (R1),#'Z        ;TEST IF BUFFER Z
 3414  007634  001003                        BNE     44$             ;NO - SKIP
 3415  007636  012703  001462                MOV     #PATZ,R3        ;GET ADDRESS OF BUFFER Z
 3416  007642  000407                        BR      5$
 3417  007644  121127  000110        44$:    CMPB    (R1),#'H        ;TEST IF HEADER BUFF DUMP
 3418  007650  001057                        BNE     21$             ;NO - SKIP TO ERROR EXIT
 3419  007652  012703  001736                MOV     #HDBUFF,R3      ;SET ADDRESS FOR HEADER BUFF
 3420  007656  012702  000102                MOV     #102,R2         ;SET SPECIAL BUFF LENGTH IF WRD CNT NULL
 3421  007662  004737  004666        5$:     JSR     PC,SBSCN        ;GET NUMBER OF WORDS PARAM
 3422  007666  105711                        TSTB    (R1)            ;TEST IF NULL
```

```
3423  007670  001007                        BNE     7$              ;NO - SKIP TO USE GIVEN VALUE
3424  007672  005702                        TST     R2              ;ELSE USE DEFAULT WORD NUMBER
3425  007674  001402                        BEQ     6$              ;IF NO WRD CNT IN R2, GO USE WORD COUNT
3426  007676  010201                        MOV     R2,R1           ;ELSE SET TO R2 COUNT
3427  007700  000410                        BR      8$
3428  007702  013701  001206        6$:     MOV     WDCNT,R1        ;GET WORD COUNT
3429  007706  000405                        BR      8$
3430  007710  010146                7$:     MOV     R1,-(SP)        ;SET UP TO CONVERT PARAMETER
3431  007712  004737  030746                JSR     PC,OCTBIN       ;GO CONVERT
3432  007716  010002                        20$                     ;ERROR RETURN
3433  007720  012601                        MOV     (SP)+,R1        ;STORE WORD COUNT GIVEN
3434  007722  005002                8$:     CLR     R2              ;CLEAR COUNTERS
3435  007724  005000                        CLR     R0
3436  007726  104400  003475                TYPE    ,DMPHDR         ;TYPE DUMP HEADERS
3437  007732  012700  000004        9$:     MOV     #4,R0           ;SET NUMBER OF COLUMNS COUNTER
3438  007736  010246                        MOV     R2,-(SP)        ;SET TO PRINT WORD NUMBER
3439  007740  104401                        TYPOC                   ;TYPE IT
3440  007742  104400  003540        10$:    TYPE    ,SPACE2         ;TYPE FORMAT SPACES
3441  007746  012346                        MOV     (R3)+,-(SP)     ;GET WORD TO TYPE
3442  007750  104401                        TYPOC
3443  007752  005202                        INC     R2              ;BUMP WORD COUNTER
3444  007754  005301                        DEC     R1              ;DEC NUMBER OF WORDS TO TYPE COUNT
3445  007756  001405                        BEQ     11$             ;IF 0, EXIT
3446  007760  005300                        DEC     R0              ;DEC NUMBER OF COL COUNT
3447  007762  001367                        BNE     10$             ;IF NOT 0, GO TYPE FORMAT SPACES AND NEXT COL
3448  007764  104400  001171                TYPE    ,$CRLF          ;ELSE LF-CR AND START NEW LINE
3449  007770  000760                        BR      9$              ;LOOP
3450  007772  104400  001171        11$:    TYPE    ,$CRLF          ;RETURN CARRIAGE
3451  007776  005724                        TST     (R4)+           ;SET UP NO ERROR RETURN
3452  010000  000406                        BR      25$
3453  010002  104400  002646        20$:    TYPE    ,BADOCT         ;REPORT NON-OCTAL PARAMETER
3454  010006  000402                        BR      24$
3455  010010  104400  002624        21$:    TYPE    ,IVDPAR         ;REPORT INVALID PARAM
3456  010014  011404                24$:    MOV     (R4),R4         ;ERROR RETURN
3457  010016                        25$:
3458  010016  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
3459  010020  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
3460  010022  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
3461  010024  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
3462  010026  000204                        RTS     R4
3463
3464
3465                                ;;*****************************************************************
3466                                .SBTTL  RUN ROUTINE
3467                                ;*      ENTRY:  JSR     R4,RURTE
3468                                ;*      RETURN: RESET   STACK, JUMP TO COMMAND LEVEL
3469                                ;*
3470                                ;*THIS ROUTINE CHECKS TO BE SURE OBJECT CODE EXISTS.
3471                                ;*IT THEN CHECKS THE NOCK SWITCH TO SEE IF THE OBJECT
3472                                ;*WAS COMPILED WITH THE NOCK OPTION. IF IT WAS THE ROUTINE
3473                                ;*PROCEEDS TO CLEAN OFF THE STACK AND EXECUTE THE OBJECT CODE.
3474                                ;*
3475                                ;*IF THE NOCK SWITCH IS OFF THE ROUTINE CHECKS THE CONTROLLER
3476                                ;*ERROR BIT AND THE DRIVE INTERRUPT BIT (BIT 15 & 14 OF CS1).
3477                                ;*IF EITHER IS SET A SUBSYSTEM CLEAR IS EXECUTED BEFORE THE
3478                                ;*TEST IS STARTED. THIS IS NECESSARY BECAUSE IF EITHER OF THESE
```

```
3479                                    ;*BITS ARE SET WHEN THE TEST IS STARTED (WITHOUT NO CHECK)
3480                                    ;*CONTINUIOUS INTERRUPTS WILL BE GENERATED. THIS CASE CAN ONLY
3481                                    ;*OCCUR IF THE PREVIOUS TEST WAS A NO CHECK TEST AND AN ERROR
3482                                    ;*WAS LEFT UNCLEARED IN THE SUBSYSTEM.
3483                                    ;*IT THEN CLEANS OFF THE STACK AND DOES A JSR TO THE
3484                                    ;*START OF THE OBJECT FILE.
3485                                    ;*
3486                                    ;*THIS ROUTINE ALSO ACTS AS A MONITOR TO CONTROL THE
3487                                    ;*LOOPING ON THE OBJECT. IT HANDLES THE ITERATION
3488                                    ;*COUNTING AND RESTARTING THE OBJECT FILE, MAKING SURE
3489                                    ;*THE STACK IS CLEANED UP TO PREVENT ACCIDENTAL OVERFLOW.
3490                                    ;*
3491                                    ;*WHEN LOOPING IS DONE OR WHEN TEST IS ABORTED
3492                                    ;*(ABORTING IS PRESENTLY UNDEFINED) THE ROUTINE RESETS
3493                                    ;*THE STACK AND JUMPS TO COM LEV.
3494                                    ;*
3495
3496  010030  005037  001722            RURTE:  CLR     LPCNT1          ;CLEAR LOOP COUNTER
3497  010034  005037  001724                    CLR     LPCNT2
3498  010040  013737  001260  023366            MOV     TOVAL,W.SEC     ;SET TIMEOUT VALUE
3499  010046  105737  001235                    TSTB    VLDOBJ          ;TEST OBJECT VALID
3500  010052  001510                            BEQ     RUEXIT          ;NO OBJECT CODE
3501  010054  152737  000377  001234            BISB    #377,SFEMP      ;SET SOURCE FILE EMPTY
3502  010062  013702  023342                    MOV     RKBAS,R2        ;GET UNIBUS BASE ADDRESS
3503  010066  105037  001103                    CLRB    $ERFLG          ;CLEAR ERROR FLAG
3504  010072  032762  140000  000000            BIT     #CERR!DI,RKCS1(R2)  ;TEST FOR CONT ERR OR DEV INTERRUPT
3505  010100  001403                            BEQ     1$              ;BOTH OFF, SKIP CLEAR
3506  010102  052762  000040  000010            BIS     #SCLR,RKCS2(R2) ;SET CLEAR SUBSYSTEM, BIT 5 CS2
3507  010110  013737  001232  001714    1$:     MOV     ITCNT,CNTSTR    ;STORE OFF ITERATION COUNT
3508  010116  012737  010144  001110            MOV     #LOC2$,$LPERR   ;SET UP ERROR LOOP VALUE
3509  010124  012706  001100            LPRET:  MOV     #STACK,SP       ;CLEAN STACK
3510  010130  005037  001116                    CLR     LINNUM          ;INITIALIZE PSUEDO LINE COUNTER
3511  010134  004737  034434                    JSR     PC,OFILE        ;GO TO OBJECT CODE
3512  010140  000401                            BR      LOC2$           ;GOOD RETURN
3513  010142  000454                            BR      RUEXIT          ;ABORT RETURN
3514  010144  032777  040000  170766    LOC2$:  BIT     #SW14,@SWR      ;LOOP ON TEST?
3515  010152  001012                            BNE     1$
3516  010154  105737  001103                    TSTB    $ERFLG          ;TEST IF ERROR FLAG SET
3517  010160  001420                            BEQ     3$
3518  010162  032777  001000  170750            BIT     #SW9,@SWR       ;TEST IF LOOP ON ERROR SWITCH
3519  010170  001003                            BNE     1$
3520  010172  105037  001103                    CLRB    $ERFLG          ;NO LOOP ON ERROR, CLEAR FLAG
3521  010176  000411                            BR      3$
3522  010200  062737  000001  001722    1$:     ADD     #1,LPCNT1       ;ADD ONE TO COUNTER
3523  010206  005537  001724                    ADC     LPCNT2          ;PROPAGATE CARRY
3524  010212  103002                            BCC     2$              ;TEST IF COUNTER OVERFLOWED
3525  010214  104400  003441                    TYPE    ,LCNTOF         ;TYPE OVERFLOW WARNING
3526  010220  000741                    2$:     BR      LPRET           ;RETURN TO LOOP
3527  010222  032777  004000  170710    3$:     BIT     #SW11,@SWR      ;INHIBIT ITERATIONS?
3528  010230  001003                            BNE     LOC3$           ;BACK TO MONITOR
3529  010232  005337  001714                    DEC     CNTSTR          ;DEC IT COUNT
3530  010236  001332                            BNE     LPRET           ;BACK TO TEST
3531  010240  004737  016670            LOC3$:  JSR     PC,PRLPCT       ;GO PRINT LOOP COUNTER
3532  010244  012706  001100            5$:     MOV     #STACK,SP       ;CLEAN STACK
3533  010250  005037  001116                    CLR     LINNUM          ;SET LINE NUMBER TO ZERO
3534  010254  105737  001663                    TSTB    CHNFLG          ;TEST IF CHAINING
```

```
3535  010260  001403                        BEQ     RURETN          ;BRANCH IF NOT, ELSE
3536  010262  004437  013020                JSR     R4,ITRTE        ;JUMP TO INPUT TEST
3537  010266  010274                        RUEXIT
3538  010270  000137  004416      RURETN:   JMP     COMLEV          ;JUMP TO COMMAND LEVEL
3539  010274  104400  032764      RUEXIT:   TYPE    ,IVDRUN         ;TYPE NO OBJECT CODE
3540  010300  000757                        BR      LOC3$
3541
3542                              ;;*****************************************************************
3543                              .SBTTL  COMPILE ROUTINE
3544                              ;*      ENTRY   JSR     R4,CORTE
3545                              ;*      RETURN  RTS     R4+2    FOR ALL RETURNS.  ERROR AND ERROR
3546                              ;*                              MESSAGES ARE ALL HANDLED LOCALLY
3547                              ;*
3548                              ;*THIS ROUTINE ACTS AS A MONITOR FOR THE COMPILE PROCESS.
3549                              ;*THE DEFERRED COMMANDS ARE EXTRACTED FROM THE SOURCE
3550                              ;*FILE. THE INTERACTIVE COMMAND TABLE IS SCANNED TO
3551                              ;*LOCATE THE ENTRY FOR THIS COMMAND. R2 IS SET
3552                              ;*TO POINT TO THE 2ND WORD WHICH IS THE ADDRESS OF
3553                              ;*THE SPECIFIC COMMAND PROCESSOR ROUTINE. CONTROL
3554                              ;*IS THEN GIVEN TO THAT ROUTINE TO GENERATE THE OBJECT
3555                              ;*CODE. R5 POINTS TO WHERE THE OBJECT CODE IS TO BE
3556                              ;*PLACED.
3557                              ;*
3558                              ;*THIS ROUTINE CHECKS THE COMPILE COMMAND PARAMETERS. IF
3559                              ;*THE FIRST PARAMETER SPECIFIES NO CHECK AND THE SECOND SPECIFIES
3560                              ;*BUS ADDRESS INCREMENT INHIBIT FOR DATA TRANSFERS. IF THE FIRST IS
3561                              ;*NULL, THE NO CHECK (NOCK) SWITCH IS RESET. IF NOT NULL
3562                              ;*THE NO CHECK SWITCH IS SET. THIS SWITCH IS USED TO DETERMINE
3563                              ;*IF NO CHECKING IS TO BE DONE IN TEST EXECUTION. IF THE SECOND
3564                              ;*IS NULL THE BUS ADDRESS INCREMENT INHIBIT SWITCH IS RESET,
3565                              ;*ELSE IT IS SET.
3566
3567                              ;*
3568                              ;*A CHECK IS MADE TO INSURE THE COMMAND LINE NUMBERS ARE
3569                              ;*RETRIEVED SEQUENTIALLY. IF NOT, AN "INTERNAL ERROR" MESSAGE
3570                              ;*IS PRINTED OUT. THIS LINE COUNT IS TESTED AGAINST THE
3571                              ;*STORED LINE NUMBERS.  WHEN EQUAL, AN RTS PC IS INSERTED
3572                              ;*IN THE OBJECT FILE AND, IF NO ERROR HAS BEEN FOUND, THE
3573                              ;*VALID OBJECT CODE FLAG IS SET. A COMPILE OK MESSAGE IS
3574                              ;*THEN PRINTED.
3575                              ;*
3576                              ;*IF ANY OF THE SPECIFIC DEFERRED COMMAND PROCESSOR
3577                              ;*ROUTINES RETURNS AN ERROR, A MESSAGE AND THE
3578                              ;*BAD LINE IS PRINTED. THE COMPILE ERROR FLAG IS SET
3579                              ;*AND THE NEXT LINE IS PROCESSED.
3580                              ;*
3581                              ;*WHEN COMPILATION IS DONE, THE COMPILE ERROR FLAG IS
3582                              ;*CHECKED. IF SET, THE VALID OBJECT CODE FLAG IS NOT
3583                              ;*SET AND CONTROL IS RETURNED TO COMMAND LEVEL.
3584                              ;*
3585                              ;* ROUTINES CALLED
3586                              ;*      SBSCN
3587                              ;*      ICDEC
3588                              ;*      REQUIRED DEFERRED COMMAND PROCESSOR (C$XX)
3589                              ;*      TYPDS
3590                              ;*      TYPE
```

```
 3591                                   ;;*****************************************************************
 3592
 3593  010302                           CORTE:
 3594  010302  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 3595  010304  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 3596  010306  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
 3597  010310  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
 3598  010312  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
 3599  010314  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
 3600  010316  105737  001234                  TSTB    $FEMP           ;TEST IF SOURCE FILE EMPTY
 3601  010322  001132                          BNE     20$             ;NO SOURCE, EXIT
 3602  010324  105037  001664                  CLRB    C$ERR           ;CLEAR COMPILE ERROR
 3603  010330  004737  004666                  JSR     PC,SBSCN        ;SEARCH FOR PARAMETER
 3604  010334  142737  000006  001244                  BICB    #NOCK!BAII,OPFLGS ;CLEAR SWITCHES
 3605  010342  105711                          TSTB    (R1)            ;TEST IF EITHER PARAM GIVEN
 3606  010344  001415                          BEQ     6$              ;NEITHER GIVEN BRANCH
 3607  010346  121127  000054                  CMPB    (R1),#',        ;TEST IF NO CHECK GIVEN
 3608  010352  001403                          BEQ     5$              ;NO SKIP TO NEXT PARAN TEST
 3609  010354  152737  000002  001244          BISB    #NOCK,OPFLGS    ;ELSE SET NO CHECK SWITCH
 3610  010362  004737  004666          5$:     JSR     PC,SBSCN        ;BUMP TO NEXT PARAMETER
 3611  010366  105711                          TSTB    (R1)            ;TEST IF THAT NULL
 3612  010370  001403                          BEQ     6$              ;YES - SKIP
 3613  010372  152737  000004  001244          BISB    #BAII,OPFLGS    ;SET BAI INHIBIT
 3614  010400  012705  034434          6$:     MOV     #OFILE,R5       ;SET OBJECT POINTER
 3615  010404  013701  001710                  MOV     IBUFPT,R1       ;SET SOURCE FILE POINTER
 3616  010410  013703  001674                  MOV     OBJSZE,R3       ;SET OBJECT SIZE FOR CLEAR
 3617  010414  010504                          MOV     R5,R4
 3618  010416  005024                  1$:     CLR     (R4)+           ;CLEAR OBJECT FILE
 3619  010420  005303                          DEC     R3              ;DEC SIZE CNTR
 3620  010422  001375                          BNE     1$              ;LOOP UNTIL DONE
 3621  010424  105037  001235                  CLRB    VLDOBJ          ;CLEAR VALID OBJECT INDICATOR
 3622  010430  012704  000001                  MOV     #1,R4           ;SET LINE NUMBER
 3623  010434  013703  001720                  MOV     JSRR4,R3        ;PUT JSR R4 CONSTANT IN R3
 3624  010440  010100          2$:     MOV     R1,R0           ;STORE ADDRESS OF THIS LINE
 3625  010442  1[illegible]7421                  CMPB    R4,(R1)+        ;TEST CORRECT LINE NUMBER IN SOURCE
 3626  010444  001064                          BNE     21$
 3627  010446  004437  004704                  JSR     R4,ICDEC        ;DECODE INTERACTIVE COMMAND
 3628  010452  010624                          22$
 3629  010454  004472  000000                  JSR     R4,@(R2)                ;JUMP TO COMMAND COMPILE ROUTINE
 3630  010460  010624                          22$                     ;ERROR RETURN
 3631  010462  105721          3$:     TSTB    (R1)+           ;TEST FOR NULL
 3632  010464  001376                          BNE     3$              ;LOOP
 3633  010466  005204                          INC     R4              ;BUMP LINE NUMBER
 3634  010470  120437  001243                  CMPB    R4,LNCNT        ;TEST IF LAST LINE PROCESSED
 3635  010474  101017                          BHI     4$              ;DONE, EXIT
 3636  010476  013737  001674  001702          MOV     OBJSZE,TEMP1    ;GET OBJECT FILE SIZE(WORDS)
 3637  010504  006337  001702                  ASL     TEMP1           ;MULTIPLY BY TWO(BYTES)
 3638  010510  062737  034434  001702          ADD     #OFILE,TEMP1    ;GET LAST ADDRESS OF OFILE
 3639  010516  162737  000012  001702          SUB     #12,TEMP1       ;ALLOW FOR THE NEXT COMMAND
 3640  010524  020537  001702                  CMP     R5,TEMP1        ;CHECK IF ROOM
 3641  010530  101743                          BLOS    2$              ;YES - GO GET NEXT LINE
 3642  010532  000423                          BR      23$             ;NO - TYPE MESSAGE
 3643  010534  105737  001664          4$:     TSTB    C$ERR           ;TEST IF COMPILE ERROR
 3644  010540  001010                          BNE     35$             ;ERROR EXIT
 3645  010542  013725  001716                  MOV     RTSPC,(R5)+     ;MOVE RETURN TO OBJ FILE
 3646  010546  105137  001235                  COMB    VLDOBJ          ;SET OBJECT VALID
```

```
 3647  010552  010537  001712                    MOV     R5,OFPTR        ;SET OBJECT FILE POINTER
 3648  010556  104400  003000                    TYPE    ,COMPOK         ;TYPE COMPILE OK MESSAGE
 3649
 3650  010562                          35$:
 3651  010562  012605                            MOV     (SP)+,R5        ;;POP STACK INTO R5
 3652  010564  012604                            MOV     (SP)+,R4        ;;POP STACK INTO R4
 3653  010566  012603                            MOV     (SP)+,R3        ;;POP STACK INTO R3
 3654  010570  012602                            MOV     (SP)+,R2        ;;POP STACK INTO R2
 3655  010572  012601                            MOV     (SP)+,R1        ;;POP STACK INTO R1
 3656  010574  012600                            MOV     (SP)+,R0        ;;POP STACK INTO R0
 3657  010576  005724                            TST     (R4)+           ;SETUP RETURN
 3658  010600  000204                            RTS     R4              ;RETURN
 3659
 3660  010602  104400  002445          23$:      TYPE    ,SCTOBG         ;TYPE MESSAGE
 3661  010606  000765                            BR      35$
 3662  010610  104400  003015          20$:      TYPE    ,NOSRC          ;TYPE NO SOURCE MESSAGE
 3663  010614  000762                            BR      35$
 3664  010616  104400  003031          21$:      TYPE    ,INTERR         ;TYPE INTERNAL ERROR
 3665  010622  000757                            BR      35$
 3666  010624  104400  003063          22$:      TYPE    ,BADCOM         ;TYPE BAD COMMAND ERROR
 3667  010630  005046                            CLR     -(SP)           ;CLEAR NEXT STACK WORD
 3668  010632  112016                            MOVB    (R0)+,(SP)      ;MOVE LINE NUMBER TO STACK
 3669  010634  104404                            TYPDS                   ;TYPE IT
 3670  010636  104400  002516                    TYPE    ,EQSGN          ;TYPE EQUAL SIGN
 3671  010642  010037  010650                    MOV     R0,40$          ;ADDRESS OF REST OF BAD LINE
 3672  010646  104400                            TYPE                    ;TYPE BAD LINE
 3673  010650  000000                  40$:      .WORD
 3674  010652  104400  001171                    TYPE    ,$CRLF          ;TYPE CARRIAGE RETURN
 3675  010656  152737  000377  001664  26$:      BISB    #377,C$ERR      ;SET ERROR FLAG
 3676  010664  000676                            BR      3$              ;PROCESS NEXT COMMAND
 3677
 3678
 3679                                    ;;*****************************************************************
 3680                                    .SBTTL  DEFERRED COMMAND PROCESSOR ROUTINES
 3681                                    ;*ALL ROUTINES THAT PROCESS DEFERRED COMMANDS ARE
 3682                                    ;*CALLED AS FOLLOWS:
 3683                                    ;*      JSR     R4,C$XX WHERE XX IS THE COMMAND MNEMONIC
 3684                                    ;*THE RETURN IS:
 3685                                    ;*      RTS     R4      FOR ERROR RETURN
 3686                                    ;*      RTS     R4+2    FOR NORMAL RETURN
 3687                                    ;*
 3688                                    ;*WHEN THE ROUTINE IS CALLED R1 POINTS TO THE COMMAND FIELD,
 3689                                    ;*R3 CONTAINS THE JSR R4 CONSTANT,
 3690                                    ;*R5 POINTS TO THE OBJECT FILE WHERE
 3691                                    ;*THIS ROUTINE MUST INSERT THE OBJECT CODE, AND R2
 3692                                    ;*POINTS TO THE 2ND WORD OF THE TABLE.
 3693                                    ;*
 3694                                    ;*THE OBJECT CODE WILL CONSIST OF JUMPS TO SPECIFIC SUBROUTINES
 3695                                    ;*WHERE THE SPECIFIC COMMAND IS EXECUTED. THESE
 3696                                    ;*PROCESSOR ROUTINES WILL INSERT THE JSR R4 (004437 OCTAL)
 3697                                    ;*FOLLOWED BY THE ADDRESS OF THE COMMAND EXECUTION
 3698                                    ;*ROUTINE. THIS ADDRESS IS TAKEN FROM THE 4TH WORD
 3699                                    ;*OF THE INTERACTIVE COMMAND TABLE. THE EXCEPTION TO
 3700                                    ;*THIS IS THE SUBSYSTEM FUNCTION INTERACTIVE COMMAND
 3701                                    ;*WHERE THE ADDRESS OF THE SUBSYSTEM COMMAND
 3702                                    ;*EXECUTION ROUTINE IS FOUND IN THE SUBSYSTEM COMMAND
```

```
                                ;*TABLE (SCTBL).
                                ;*
                                ;*THE COMMAND PROCESSOR ROUTINES ALSO PLACE THE
                                ;*PARAMETERS REQUIRED BY THE EXECUTION ROUTINES IN
                                ;*THE OBJECT FILE. THE NUMBER OF PARAMETERS WILL
                                ;*VARY FROM ONE COMMAND TO ANOTHER BUT IS ALWAYS
                                ;*THE SAME FOR A SPECIFIC COMMAND. THUS THE
                                ;*PROCESSOR ROUTINES KNOW HOW MANY PARAMETERS TO
                                ;*PLACE IN THE OBJECT FILE AND THE EXECUTION
                                ;*ROUTINES KNOW HOW MANY TO RETRIEVE.
                                ;;*****************************************************************

                                ;;*****************************************************************
                                .SBTTL  PRINT MESSAGE PROCESSOR
                                ;*ENTRY          JSR     R4,C$PM
                                ;*RETURN         RTS     R4      ERROR RETURN
                                ;*              RTS     R4+2    NO ERROR RETURN
                                ;*
                                ;*OBJECT CODE:
                                ;*      JSR     R4,E$PM
                                ;*      <MESSAGE>       WHERE MESSAGE IS UP TO 20 WORDS
                                ;*
                                ;*THIS ROUTINE GENERATES THE CODE TO PRINT A MESSAGE OF UP
                                ;*TO 20 WORDS (40 CHARACTERS) ON THE TERMINAL
                                ;;*****************************************************************


010666  004737  004666          C$PM:   JSR     PC,SBSCN        ;BUMP R1 PAST COMMAND FIELD
010672  105711                          TSTB    (R1)            ;TEST IF MESSAGE NULL
010674  001411                          BEQ     2$              ;EXIT IF YES (NO MESSAGE)
010676  010325                          MOV     R3,(R5)+        ;INSERT JSR R4 CONSTANT
010700  022222                          CMP     (R2)+,(R2)+     ;BUMP R2 TO 4TH WORD (EXECUTE ADDR)
010702  011225                          MOV     (R2),(R5)+      ;INSERT EXECUTION ROUTINE ADDRESS
010704  112125                  1$:     MOVB    (R1)+,(R5)+     ;INSERT MESSAGE
010706  001376                          BNE     1$              ;LAST CHARACTER MOVED NULL?
                                                                ;LOOP IF NO
010710  032705  000001                  BIT     #1,R5           ;TEST IF R5 IS EVEN.
010714  001401                          BEQ     2$              ;IF NOT, CLEAR NEXT LOCATION
010716  105025                          CLRB    (R5)+           ;AND MAKE R5 EVEN.
010720  105741                  2$:     TSTB    -(R1)           ;DEC R1 TO POINT TO NULL IN SOURCE
010722  005724                          TST     (R4)+           ;SET GOOD RETURN
010724  000204                          RTS     R4              ;RETURN


                                ;;*****************************************************************
                                .SBTTL  REGISTER COMPARE, REGISTER WRITE, AND STATUS COMPARE PROCESSORS
                                ;*ENTRIES:      JSR     R4,C$RC FOR REGISTER COMPARE
                                ;*              JSR     R4,C$RW FOR REGISTER WRITE
                                ;*              JSR     R4,C$SC FOR STATUS COMPARE
                                ;*
                                ;*RETURN:       RTS     R4      ERROR RETURN
                                ;*              RTS     R4+2    NORMAL RETURN
                                ;*
                                ;*OBJECT CODE:
                                ;*              JSR     R4      E$RC OR E$RW OR E$SC
                                ;*              REGISTER OR STATUS WORD NUMBER
                                ;*              EXPECTED VALUE OR VALUE TO BE WRITTEN
```

```
                                ;*              MASK (RC & SC ONLY)
                                ;*
                                ;*THIS ROUTINE HAS THREE ENTRY POINTS, ONE FOR EACH
                                ;*OPERATION. THE LINK TO THE PROPER EXECUTION ROUTINE
                                ;*IS GENERATED.
                                ;*
                                ;*EACH PARAMETER IS TAKEN FROM THE COMMAND LINE IF IT IS PROVIDED.
                                ;*IF PROVIDED, IT IS ALSO STORED FOR POSSIBLE LATER USE. IF IT IS NOT
                                ;*SUPPLIED, THE VALUE GIVEN THE LAST TIME IT WAS SPECIFIED IS USED.
                                ;*NOTE THAT THE PARAMETER STORAGE FOR THE REGISTER COMPARE
                                ;*AND THE REGISTER WRITE (REGISTER NUMBER AND VALUE) IS THE SAME. THIS MEANS
                                ;*THAT WHEN ONE COMMAND CHANGES THE VALUE IT IS CHANGED FOR
                                ;*BOTH COMMANDS.
                                ;*
                                ;*THE REGISTER ASSIGNMENT IS:
                                ;*
                                ;*      0       CS1 (CONTROL AND STATUS 1)
                                ;*      1       WC (WORD COUNT)
                                ;*      2       BA (BUFFER ADDRESS)
                                ;*      3       DA (DESIRED TRACK AND SECTOR)
                                ;*      4       CS2 (CONTROL AND STATUS 2)
                                ;*      5       DS (DRIVE STATUS)
                                ;*      6       ER (ERROR REGISTER)
                                ;*      7       ASOF (ATTENTION SUMMARY AND OFFSET)
                                ;*      10      DC (DESIRED CYLINDER)
                                ;*      12      DB (DATA BUFFER)
                                ;*      13      MR1 (MAINTENANCE REGISTER 1)
                                ;*      14      ECC POSITION REGISTER
                                ;*      15      ECC PATTERN REGISTER
                                ;*      16      MR2 (MAINTENANCE REG 2)
                                ;*      17      MR3 (MAINTENANCE REG 3)
                                ;*
                                ;*THE STATUS WORD ASSIGNMENT IS:
                                ;*
                                ;*      0       LINE A WORD 0
                                ;*      1       LINE B WORD 0
                                ;*      2       LINE A WORD 1
                                ;*      3       LINE B WORD 1
                                ;*      4       LINE A WORD 2
                                ;*      5       LINE B WORD 2
                                ;*      6       LINE A WORD 3
                                ;*      7       LINE B WORD 3
                                ;;*****************************************************************
                                
010726  012737  001224  001702  C$RW:   MOV     #REGNUM,TEMP1   ;STORE RW PARAM BASE
010734  012737  000002  001704          MOV     #2,TEMP2        ;STORE NUMBER OF PARAM
010742  000415                  C$RCWS  BR      C$RCWS          ;BRANCH TO COMMON RTE
010744  012737  001224  001702  C$RC:   MOV     #REGNUM,TEMP1   ;STORE RC PARAM BASE
010752  012737  000003  001704          MOV     #3,TEMP2        ;STORE NUMBER OF PARAM
010760  000406                  C$RCWS  BR      C$RCWS          ;BRANCH TO COMMON RTE
010762  012737  001216  001702  C$SC:   MOV     #STATRD,TEMP1   ;STORE SC PARAM BASE
010770  012737  000003  001704          MOV     #3,TEMP2                ;STORE NUMBER OF PARAM
010776                          C$RCWS:
010776  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
011000  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
011002  013704  001704                  MOV     TEMP2,R4        ;SET NUMBER OF PARAM
```

```
3815  011006  013700  001702              MOV     TEMP1,R0        ;SET PARAM BASE
3816
3817  011012  004737  004666      1$:     JSR     PC,SBSCN        ;BUMP R1 TO NEXT PARAM
3818  011016  105711                      TSTB    (R1)            ;TEST FOR NULL (REMAIN PARAM DEFAULT)
3819  011020  001425                      BEQ     3$              ;BRANCH IF YES
3820  011022  121127  000054              CMPB    (R1),#',        ;TEST IF THIS PARAM NULL
3821  011026  001417                      BEQ     2$              ;NULL PARAM, SET FOR NEXT
3822  011030  121127  000067              CMPB    (R1),#67        ;TEST 1ST CHAR OF PARAM
3823  011034  101407                      BLOS    40$             ;LESS THAN 7 - BRANCH
3824  011036  010346                      MOV     R3,-(SP)        ;STORE R3
3825  011040  004737  007054              JSR     PC,RGDCDE       ;GO DECODE REGISTER
3826  011044  011140                      21$                     ;ERROR RETURN
3827  011046  010310                      MOV     R3,(R0)         ;STORE REG NUMBER
3828  011050  012603                      MOV     (SP)+,R3        ;RESTORE R3
3829  011052  000405                      BR      2$
3830  011054  010146              40$:    MOV     R1,-(SP)        ;SET UP TO CONVERT PARAM
3831  011056  004737  030746              JSR     PC,OCTBIN       ;TO BINARY
3832  011062  011132                      20$
3833  011064  012610                      MOV     (SP)+,(R0)      ;STORE PARAM
3834  011066  005720              2$:     TST     (R0)+           ;BUMP R0 TO NEXT PARAM STORE
3835  011070  005304                      DEC     R4              ;DEC PARAM COUNT. TAKE
3836  011072  001347                      BNE     1$              ;ONLY 3 PARAMETERS
3837  011074  013700  001702      3$:     MOV     TEMP1,R0        ;RESET TO BASE FOR PARAM INSERTION
3838  011100  010325                      MOV     R3,(R5)+        ;INSERT JSR R4 CONSTANT
3839  011102  022222                      CMP     (R2)+,(R2)+     ;BUMP R2 TO 4TH WORD
3840  011104  011225                      MOV     (R2),(R5)+      ;INSERT EXECUTE ADDRESS
3841  011106  012025                      MOV     (R0)+,(R5)+     ;INSERT REGISTER OR STATUS WD NULL
3842  011110  012025                      MOV     (R0)+,(R5)+     ;INSERT EXPECTED VALUE
3843  011112  023727  001704  000002      CMP     TEMP2,#2        ;TEST IF RW(ONLY 2 PARAM)
3844  011120  001401                      BEQ     4$              ;IF YES, GET OUT. ELSE
3845  011122  012025                      MOV     (R0)+,(R5)+     ;INSERT MASK
3846  011124  012604              4$:     MOV     (SP)+,R4        ;RESSTORE R4 FOR RETURN
3847  011126  005724                      TST     (R4)+           ;NORMAL RETURN
3848  011130  000407                      BR      25$
3849  011132  104400  002646      20$:    TYPE    ,BADOCT         ;TYPE INVALID OCTAL MESSAGE
3850  011136  000402                      BR      22$
3851  011140  104400  003361      21$:    TYPE    ,BADSEL         ;BAD REG SELECTION MESSAGE
3852  011144  012604              22$:    MOV     (SP)+,R4        ;RESTORE R4 FOR RETURN
3853  011146  011404                      MOV     (R4),R4         ;BAD RETURN
3854  011150                      25$:
3855  011150  012600                      MOV     (SP)+,R0        ;;POP STACK INTO R0
3856  011152  000204                      RTS     R4              ;RETURN
3857
3858                              ;;*****************************************************************
3859                              .SBTTL  BUFFER INITIALIZE PROCESSOR
3860                              ;*ENTRY:        JSR     R4,C$BI OR JSR R4,C$CBI
3861                              ;*RETURN:       RTS     R4      ERROR RETURN
3862                              ;*              RTS     R4+2    NORMAL ROUTINE
3863                              ;*
3864                              ;*OBJECT CODE:
3865                              ;*      JSR     R4,E$BI
3866                              ;*      <EXECUTION LINE COUNT CONTROL><PATTERN NAME>
3867                              ;*
3868                              ;*THIS ROUTINE GENERATES THE LINK TO THE BUFFER INITIALIZE EXECUTION
3869                              ;*THE PATTERN NAME IS ENTERED AS THE LOW ORDER BYTE OF THE PARAMETER.
3870                              ;*BIT 15 IF THE PARAMETER BYTE IS USED TO TELL THE BI EXECUTION
```

```
                                        ;*IF THIS IS AN INTERNALLY GENERATED BI OR A USER COMMAND.  IF
                                        ;*USER COMMAND, BIT 15 IS A ONE.  THIS INDICATOR CONTROLS LINE
                                        ;*COUNT INCREMENT DURING EXECUTION.
                                        ;*
                                        ;*THIS ROUTINE HAS A SPECIAL ENTRY (C$CBI).  THIS ENTRY IS USED
                                        ;*WHEN A DATA TRANSFER OPERATION REQUIRES BUFFER INITIALIZATION.
                                        ;*THE COMPILER ROUTINE INSERTS A JSR TO THE SPECIAL ENTRY INTO THE
                                        ;*OBJECT CODE BEFORE THE DATA TRANSFER JSR (SEE DESCRIPTION OF
                                        ;*SUBSYSTEM FUNCTION PROCESSOR).
                                        ;*
                                        ;*THE RANDOM PATTERN IS GENERATED AND STORED IN THIS ROUTINE IF
                                        ;*PATTERN R IS SELECTED.
                                        ;;*****************************************************************

011154                                  C$BI:
011154  010046                                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
011156  004737  004666                          JSR     PC,SBSCN        ;BUMP R1 TO NEXT PARAM
011162  111100                                  MOVB    (R1),R0         ;MOVE PAT NAME INTO R0
011164  001002                                  BNE     1$              ;IF NOT NULL, BRANCH, ELSE
011166  113700  001236                          MOVB    PATSEL,R0       ;MOVE "A" INTO R0
011172  052700  100000              1$:         BIS     #BIT15,R0       ;SET BIT 15 FOR LINE COUNT CONTROL
011176  000401                                  BR      C$CBII          ;BRANCH AROUND SPECIAL ENTRY
011200                                  C$CBI:
011200  010046                                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
011202  110037  001236              C$CBII:     MOVB    R0,PATSEL       ;STORE PATTERN SELECT
011206  010325                                  MOV     R3,(R5)+        ;INSERT JSR R4 CONSTANT
011210  022222                                  CMP     (R2)+,(R2)+     ;BUMP R2 TO 4TH WORD TO
011212  011225                                  MOV     (R2),(R5)+      ;INSERT EXECUTE ADDRESS
011214  010025                                  MOV     R0,(R5)+        ;INSERT INDEX INTO PATTBL
011216  120027  000122                          CMPB    R0,#'R          ;RANDOM PATTERN?
011222  001015                                  BNE     3$              ;NO - BRANCH OUT
011224  012700  001562                          MOV     #PATR,R0        ;GET ADDRESS OF PATR STORE
011230  004737  033756              2$:         JSR     PC,$RAND        ;GENERATE RANDOM VALUES
011234  013720  034054                          MOV     $HINUM,(R0)+    ;LOAD PATR WITH HI VALUE
011240  013720  034056                          MOV     $LONUM,(R0)+    ;LOAD PATR WITH LO VALUE
011244  022700  001662                          CMP     #PATR+100,R0    ;PATTERN FULL?
011250  001367                                  BNE     2$              ;NO - DO IT AGAIN
011252  105137  001242                          COMB    PATRDF          ;SET PAT R DEFINED SWITCH
011256                                  3$:
011256  012600                                  MOV     (SP)+,R0        ;;POP STACK INTO R0
011260  005724                                  TST     (R4)+           ;SET NORMAL RETURN
011262  000204                                  RTS     R4              ;RETURN

                                        ;;*****************************************************************
                                        .SBTTL  DATA COMPARE PROCESSOR
                                        ;*ENTRY:        JSR     R4,C$DC
                                        ;*RETURN:       RTS     R4      ERROR RETURN
                                        ;*              RTS     R4+2    NORMAL RETURN
                                        ;*OBJECT CODE:
                                        ;*      JSR     R4,E$DC
                                        ;*      COMPARE LENGTH (OCTAL)
                                        ;*
                                        ;*THIS ROUTINE GENERATES THE LINK TO THE DATA COMPARE EXECUTE.
                                        ;*THE COMPARE LENGTH PARAMETER IS TAKEN FROM THE COMSZE
                                        ;*PARAMETER STORAGE LOCATION IF IT IS NOT GIVEN WITH THE COMMAND.
                                        ;*IF IT IS GIVEN, IT IS STORED IN COMSZE FOR POSSIBLE LATER
```

```
                                ;*USE. (COMSZE WILL ALWAYS BE EITHER THE DATA COMPARE
                                ;*LENGTH PARAMETER OR THE WORD COUNT OF THE LAST
                                ;*INPUT DATA TRANSFER.)
                                ;;*****************************************************************
011264                          C$DC:
011264  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
011266  004737  004666                  JSR     PC,SBSCN        ;BUMP R1 TO NEXT PARAM
011272  105711                          TSTB    (R1)            ;TEST IF PARAM NULL
011274  001406                          BEQ     2$              ;BR IF YES
011276  010146                          MOV     R1,-(SP)        ;SET UP FOR CONVERT
011300  004737  030746                  JSR     PC,OCTBIN       ;CONVERT PARAM TO BINARY
011304  011330                          20$                     ;ERROR RETURN
011306  012637  001254                  MOV     (SP)+,COMSZE    ;STORE COMPARE LENGTH
011312  010325                  2$:     MOV     R3,(R5)+        ;INSERT JSR R4 CONSTANT
011314  022222                          CMP     (R2)+,(R2)+     ;BUMP R2 TO WORD 4 AT ICTBL
011316  011225                          MOV     (R2),(R5)+      ;INSERT EXECUTE RTE ADDRESS
011320  013725  001254                  MOV     COMSZE,(R5)+    ;INSERT COMPARE LENGTH
011324  005724                          TST     (R4)+           ;SET FOR NORMAL RETURN
011326  000403                          BR      30$
011330  104400  002646          20$:    TYPE    ,BADOCT         ;TYPE BAD DECIMAL MESSAGE
011334  011404                          MOV     (R4),R4         ;SET FOR ERROR RETURN
011336                          30$:
011336  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
011340  000204                          RTS     R4              ;RETURN

                                ;;*****************************************************************
                                .SBTTL  STALL PROCESSOR
                                ;*ENTRY:        JSR     R4,C$ST
                                ;*RETURN:       RTS     R4      ERROR RETURN
                                ;*              RTS     R4+2    ;NORMAL RETURN
                                ;*
                                ;*OBJECT CODE:
                                ;*      JSR     R4,E$ST
                                ;*      STALL DURATION (OCTAL)
                                ;*
                                ;*THIS ROUTINE GENERATES THE LINK TO THE STALL EXECUTE.
                                ;*THE PARAMETER, IF GIVEN IN THE INPUT COMMAND, IS
                                ;*DECODED FROM ASCII DECIMAL INTO BINARY AND STORED
                                ;*IN THE PARAMETER STORAGE LOCATION "STALL". IT IS THEN
                                ;*PLACED IN THE OBJECT CODE. IF THE DELAY IS NOT
                                ;*SPECIFIED IN THE COMMAND THE OLD VALUE OF
                                ;*"STALL" IS USED.
                                ;;*****************************************************************

011342  004737  004666          C$ST:   JSR     PC,SBSCN        ;BUMP R1 TO NEXT PARAM
011346  105711                          TSTB    (R1)            ;TEST IF PARAM NULL
011350  001406                          BEQ     1$              ;BR IF NULL
011352  010146                          MOV     R1,-(SP)        ;SET UP FOR CONVERT
011354  004737  031102                  JSR     PC,DECBIN       ;CONVERT PARAM TO BINARY
011360  011404                          20$                     ;ERROR RETURN
011362  012637  001252                  MOV     (SP)+,STALL     ;STORE STALL VALUE
011366  010325                  1$:     MOV     R3,(R5)+        ;INSERT JSR R4 CONSTANT
011370  022222                          CMP     (R2)+,(R2)+     ;BUMP R2 TO 4TH WORD ICTBL
011372  011225                          MOV     (R2),(R5)+      ;INSERT EXECUTE ADDRESS
011374  013725  001252                  MOV     STALL,(R5)+     ;INSERT STALL PARAMETER
```

```
 3983  011400  005724                        TST     (R4)+           ;SET UP NORMAL RETURN
 3984  011402  000403                        BR      30$             ;BR TO RETURN
 3985  011404  104400  002520          20$:  TYPE    .BADDEC         ;TYPE BAD DECIMAL MESSAGE
 3986  011410  011404                        MOV     (R4),R4         ;SET UP ERROR RETURN
 3987  011412  000204                  30$:  RTS     R4              ;RETURN
 3988
 3989                                  ;;*****************************************************************
 3990                                  .SBTTL  UNIBUS INITIALIZE PROCESSOR
 3991                                  ;*ENTRY:        JSR     R4,C$UI
 3992                                  ;*RETURN:       RTS     R4+2    NO ERROR RETURN
 3993                                  ;*
 3994                                  ;*OBJECT CODE:
 3995                                  ;*      JSR     R4,E$UI
 3996                                  ;*
 3997                                  ;*THIS ROUTINE GENERATES THE UNIBUS INITIALIZE LINK
 3998                                  ;*INTO THE OJBECT CODE.
 3999                                  ;;*****************************************************************
 4000
 4001  011414  010325                  C$UI:   MOV     R3,(R5)+        ;INSERT JSR R4 CONSTANT
 4002  011416  022222                          CMP     (R2)+,(R2)+     ;BUMP R2 TO 4TH WORD OF TABLE
 4003  011420  011225                          MOV     (R2),(R5)+      ;INSERT EXECUTE ADDRESS
 4004  011422  005724                          TST     (R4)+           ;NORMAL RETURN
 4005  011424  000204                          RTS     R4              ;RETURN
 4006
 4007                                  ;;*****************************************************************
 4008                                  .SBTTL  SUBCOMMAND TABLE
 4009                                  ;*THIS TABLE CONTAINS ALL THE SUBSYSTEM COMMANDS. THE TABLE
 4010                                  ;*FORMAT IS:
 4011                                  ;*      WORD1:  SUBCOMMAND MNEMONIC
 4012                                  ;*      WORD2:  ADDRESS OF EXECUTION SUBROUTINE FOR THIS COMMAND
 4013                                  ;*      WORD3:  THE NUMBER OF PARAMETERS THIS COMMAND REQUIRES
 4014                                  ;;*****************************************************************
 4015                                          .EVEN
 4016  011426  042122                  SCTBL:  .ASCII  /RD/            ;READ DATA
 4017  011430  021652  000005                  .WORD   E$RD,5
 4018  011434  042127                          .ASCII  /WD/            ;WRITE DATA
 4019  011436  021722  000005                  .WORD   E$WD,5
 4020  011442  041527                          .ASCII  /WC/            ;WRITE CHECK
 4021  011444  021736  000005                  .WORD   E$WC,5
 4022  011450  044127                          .ASCII  /WH/            ;WRITE HEADER
 4023  011452  021632  000005                  .WORD   E$WH,5
 4024  011456  044122                          .ASCII  /RH/            ;READ HEADER
 4025  011460  021606  000004                  .WORD   E$RH,4
 4026  011464  045523                          .ASCII  /SK/            ;SEEK
 4027  011466  021620  000004                  .WORD   E$SK,4
 4028  011472  041503                          .ASCII  /CC/            ;CONTROLLER CLEAR
 4029  011474  021500  000001                  .WORD   E$CC,1
 4030  011500  051503                          .ASCII  /CS/            ;CLEAR SUBSYSTEM
 4031  011502  021464  000001                  .WORD   E$CS,1
 4032  011506  041504                          .ASCII  /DC/            ;DRIVE CLEAR
 4033  011510  021524  000001                  .WORD   E$DC,1
 4034  011514  041522                          .ASCII  /RC/            ;RECALIBRATE
 4035  011516  021434  000001                  .WORD   E$RC,1
 4036  011522  051504                          .ASCII  /DS/            ;DRIVE SELECT
 4037  011524  021512  000001                  .WORD   E$DS,1
 4038  011530  040520                          .ASCII  /PA/            ;PACK ACKNOWLEDGE
```

```
 4039  011532  021450  000001                .WORD   ESPA,1
 4040  011536  046125                        .ASCII  /UL/            ;UNLOAD
 4041  011540  021536  000001                .WORD   ESUL,1
 4042  011544  051523                        .ASCII  /SS/            ;START SPINDLE
 4043  011546  021550  000001                .WORD   ESSS,1
 4044  011552  043117                        .ASCII  /OF/            ;OFFSET
 4045  011554  021562  000002                .WORD   ESOF,2
 4046  011560  044101                        .ASCII  /AH/            ;ALL HEADER READ
 4047  011562  021574  000004                .WORD   ESAH,4
 4048  011566  000000                        .WORD   0               ;NULL TO TERMINATE TABLE
 4049
 4050                                ;*******************************************************
 4051                                .SBTTL  SUBSYSTEM FUNCTION PROCESSOR
 4052                                ;*ENTRY:        JSR     R4,CSSF
 4053                                ;*RETURN:       RTS     R4      ERROR RETURN
 4054                                ;*              RTS     R4+2    NORMAL RETURN
 4055                                ;*OBJECT CODE:
 4056                                ;*      JSR     R4,ESXX :WHERE ESXX IS THE COMMAND EXECUTE ROUTINE
 4057                                ;*      OPERATION FLAGS :BIT 1 SET = NO CHECKING
 4058                                ;*                      :BIT 2 SET = BUS ADDRESS INCREMENT INHIBIT
 4059                                ;*                      :BIT 3 SET = 24 SECTOR FORMAT
 4060                                ;*      DRIVE NUMBER    :-1 MEANS USE STORED PARAMETER "DRIVE",
 4061                                ;*                       ELSE THIS IS DRIVE TO BE USED
 4062                                ;*      CYLINDER NUMBER OR OFFSET :DEPENDING ON COMMAND
 4063                                ;*      TRACK NUMBER
 4064                                ;*      SECTOR NUMBER
 4065                                ;*      WORD COUNT
 4066                                ;*
 4067                                ;*THE NUMBER OF PARAMETERS IS VARIABLE DEPENDING ON THE
 4068                                ;*SUBSYSTEM COMMAND. THE NUMBER IS SPECIFIED IN THE THIRD
 4069                                ;*WORD OF THE SUBSYSTEM COMMAND TABLE (SCTBL).
 4070                                ;*
 4071                                ;*THIS ROUTINE PROVIDES THE LINKS TO THE VARIOUS EXECUTION ROUTINES FOR
 4072                                ;*SUBSYSTEM COMMANDS. IT DOES THIS BY PROVIDING THE PROPER DESTINATION
 4073                                ;*ADDRESS FOR THE JSR. THE PARAMETERS REQUIRED BY THE
 4074                                ;*EXECUTION ROUTINE IS PLACED IN THE OBJECT FILE IMMEDIATELY
 4075                                ;*FOLLOWING THE JSR COMMAND.
 4076                                ;*
 4077                                ;*WHEN THE ROUTINE IS ENTERED, THE REGISTERS MUST BE SET
 4078                                ;*AS FOLLOWS:
 4079                                ;*      R0-NA
 4080                                ;*      R1-POINTS TO THE SF MNEMONIC IN THE SOURCE FILE LINE
 4081                                ;*      R2-NA
 4082                                ;*      R3-CONTAINS JSR R4 CONSTANT
 4083                                ;*      R4-NA
 4084                                ;*      R5-POINTS TO THE 1ST UNUSED LOCATION IN OBJECT FILE
 4085                                ;*
 4086                                ;*IF THE SC PARAMETER IS GIVEN IN THE SOURCE LINE THE SC MNEMONIC
 4087                                ;*IS USED TO LOCATE THE PROPER ENTRY IN THE TABLE. IF NONE
 4088                                ;*IS FOUND, AN ERROR IS REPORTED. IF A HIT OCCURS, THE
 4089                                ;*ADDRESS OF THE SECOND WORD OF THE TABLE IF LOADED INTO R2 AND
 4090                                ;*STORED IN SUBCMD AS THE INDICATION OF THE LAST COMMAND
 4091                                ;*SPECIFIED. IF THE SC PARAMETER IS NULL, THE STORED VALUE
 4092                                ;*IN SUBCMD IS PUT IN R2 WHICH EFFECTIVELY LOCATES THE TABLE
 4093                                ;*ENTRY.
 4094                                ;*
```

```
                                ;*THE REMAINING PARAMETERS IN THE SOURCE ARE THEN
                                ;*PROCESSED. IF THE PARAMETER IS NULL, THE LAST SPECIFIED
                                ;*VALUE FOR THAT PARAMETER IS USED. IF THE PARAMETER
                                ;*IS GIVEN, IT IS STORED IN PARAMETER STORAGE AND BECOMES
                                ;*THE LAST VALUE GIVEN.
                                ;*SEVERAL SPECIAL CASES EXIST. THESE ARE:
                                ;*      .PATTERN SELECT SPECIFIED
                                ;*      .NULL DRIVE PARAMETER
                                ;*      .OFFSET COMMAND
                                ;*      .INVALID WORD COUNT
                                ;*
                                ;*SUPPLYING THE PATTERN SELECT PARAMETER REQUIRES OUTPUT BUFFER
                                ;*INITIALIZATION. THIS IS ACCOMPLISHED BY USING THE BUFFER
                                ;*INITIALIZE COMMAND PROCESSOR ROUTINE WITH A SPECIAL
                                ;*ENTRY (JSR R4, C$CBI). THIS GENERATES A LINK TO THE
                                ;*BUFFER INITIALIZE EXECUTION ROUTINE THAT IS IDENTICAL TO
                                ;*A LINK GENERATED BY A BI INTERACTIVE COMMAND. THE
                                ;*LINK TO BI IN THE OBJECT FILE WILL BE INSERTED BEFORE
                                ;*THE LINK TO THE SUBSYSTEM COMMAND.
                                ;*
                                ;*A NULL DRIVE PARAMETER IS SPECIAL BECAUSE IT REQUIRES
                                ;*CONSIDERATION WHEN A COMPILED TEST IS READ FROM PAPER TAPE.
                                ;*TO SUPPORT THIS, THE PARAMETER IS SET TO -1 IF THE DRIVE IS
                                ;*NOT SPECIFIED IN THE COMMAND. THE COMMAND EXECUTION
                                ;*ROUTINE KEYS OFF THIS AND GOES TO THE STORED DRIVE
                                ;*PARAMETER FOR THE DRIVE NUMBER.
                                ;*
                                ;*THE OFFSET COMMAND REQUIRES A UNIQUE PARAMETER (OFFSET)
                                ;*IN THE POSITION OFF THE CYLINDER NUMBER. TO PRESERVE
                                ;*THE STORED CYLINDER NUMBER THIS MUST BE SPECIAL CASE.
                                ;*
                                ;*THE WORD COUNT PARAMETER IS CHECKED TO INSURE THE
                                ;*BUFFER IS LARGE ENOUGH. IF NOT AN ERROR IS PRINTED.
                                ;;*****************************************************************

011570                          C$SF:
011570  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
011572  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
011574  004737  004666                  JSR     PC,SBSCN        ;BUMP R1 TO NEXT PARAM
011600  112137  001702                  MOVB    (R1)+,TEMP1     ;MOVE 1ST CHAR OF SUB CMND.
011604  001403                          BEQ     1$              ;BRANCH IF NULL
011606  111137  001703                  MOVB    (R1),TEMP1+1    ;MOVE 2ND CHAR OF SUB CMND
011612  001015                          BNE     3$              ;BRANCH IF NOT NULL
011614  005301                  1$:     DEC     R1              ;DECREMENT R1 TO INSURE IT DOESN'T
                                                                ;GET PAST THE NULL CHARACTER
011616  013702  001214                  MOV     SUBCMD,R2       ;GET 2ND WORD ADDR FOR LAST CMND
011622  105137  001666                  COMB    SAMDR           ;SET SAME DRIVE SWITCH
011626  024227  043117                  CMP     -(R2),#"OF      ;LAST COMMAND OFFSET?
011632  001002                          BNE     2$              ;IF NO, BRANCH ELSE
011634  105137  001665                  COMB    OFFLAG          ;SET OFFSET FLAG
011640  005722                  2$:     TST     (R2)+           ;BUMP R2 TO SECOND WORD
011642  000137  012234                  JMP     41$             ;GO TO FILL OBJECT
011646  124127  000054          3$:     CMPB    -(R1),#',       ;TEST IF 1ST CHAR COMMA
011652  001423                          BEQ     7$              ;BRANCH IF YES. ELSE
011654  023727  001702  043117  31$:    CMP     TEMP1,#"OF      ;TEST IF OFFSET
011662  001002                          BNE     4$              ;BRANCH IF NO. ELSE
```

```
4151  011664  105137  001665              COMB    OFFLAG          ;SET OFFSET FLAG
4152  011670  012702  011426      4$:     MOV     #SCTBL,R2       ;LOAD ADDRESS OF SUBCMND TABLE
4153  011674  023722  001702      5$:     CMP     TEMP1,(R2)+     ;TEST IF TABLE ENTRY MATCH
4154  011700  001405                      BEQ     6$              ;YES, FOUND A HIT. BRANCH
4155  011702  005742                      TST     -(R2)           ;TEST IF TABLE ENTRY NULL
4156  011704  001502                      BEQ     26$             ;IF YES, NO MATCH IN TABLE, ERROR
4157  011706  062702  000006              ADD     #6,R2           ;BUMP R2 TO NEXT TABLE ENTRY
4158  011712  000770                      BR      5$              ;BRANCH TO TEST NEXT ENTRY
4159  011714  010237  001214      6$:     MOV     R2,SUBCMD       ;STORE ADDRESS 2ND WORD, LAST CMND
4160  011720  000410                      BR      9$
4161  011722  013702  001214      7$:     MOV     SUBCMD,R2       ;GET 2ND WORD ADDR LAST CMND
4162  011726  024227  043117              CMP     -(R2),#"OF      ;TEST IF LAST CMND OFFSET
4163  011732  001002                      BNE     8$              ;IF NO, BRANCH. ELSE
4164  011734  105137  001665              COMB    OFFLAG          ;SET OFFLAG
4165  011740  005722              8$:     TST     (R2)+           ;BUMP R2 BACK TO SECOND WORD
4166  011742  005000              9$:     CLR     R0              ;CLR R0 FOR PARAM COUNTING
4167  011744  004737  004666      10$:    JSR     PC,SBSCN        ;BUMP R1 TO NEXT PARAM
4168  011750  121127  000054              CMPB    (R1),#',        ;TEST IF COMMA
4169  011754  001005                      BNE     11$             ;BR IF NO
4170  011756  005700                      TST     R0              ;TEST IF DRIVE SELECT PARAM
4171  011760  001111                      BNE     18$             ;GO TO BUMP & LOOP
4172  011762  105137  001666              COMB    SAMDR           ;SET SAME DRIVE SWITCH
4173  011766  000506                      BR      18$             ;GO TO BUMP & LOOP
4174  011770  105711              11$:    TSTB    (R1)            ;TEST PARAM NULL
4175  011772  001005                      BNE     12$             ;NOT NULL
4176  011774  005700                      TST     R0              ;TEST IF DRIVE SELECT PARAM
4177  011776  001116                      BNE     41$             ;GO TO FILL OBJECT
4178  012000  105137  001666              COMB    SAMDR           ;SET SAME DRIVE FLAG
4179  012004  000513                      BR      41$             ;GO TO FILL OBJECT
4180  012006  020027  000016      12$:    CMP     R0,#16          ;TEST IF TOO MANY PARAM
4181  012012  002017                      BGE     21$             ;BRANCH TO ERROR
4182  012014  020027  000012              CMP     R0,#12          ;PATTERN SELECT PARAM?
4183  012020  001401                      BEQ     13$             ;IF YES BRANCH, ELSE
4184  012022  000437                      BR      14$             ;GO TO JMP & LOOP
4185  012024  010046              13$:    MOV     R0,-(SP)        ;SAVE R0
4186  012026  010246                      MOV     R2,-(SP)        ;SAVE R2
4187  012030  012702  004016              MOV     #C$CBIS,R2      ;SET R2 WITH ADDRESS OF "BI" EXEC ROUT
4188  012034  111100                      MOVB    (R1),R0         ;LOAD R0 WITH PAT SELECT CHAR
4189  012036  004437  011200              JSR     R4,C$CBI        ;JUMP TO SPECIAL BUFFER INITIALIZE
4190                                                              ;ENTRY. A JSR R4, BI (PATTERN
4191                                                              ;NAME) WILL BE INSERTED INTO
4192                                                              ;OBJECT FILE.
4193  012042  000000                      .WORD   0               ;ERROR RETURN POINT
4194  012044  012602                      MOV     (SP)+,R2        ;RESTORE R2
4195  012046  012600                      MOV     (SP)+,R0        ;RESTORE R0
4196  012050  000455                      BR      18$             ;GO TO BUMP & LOOP
4197  012052  104400  002624      21$:    TYPE    ,IVDPAR         ;TYPE INVALID PARAMETERS
4198  012056  000415                      BR      26$
4199  012060                      22$:
4200  012060  104400  002646      24$:    TYPE    ,BADOCT         ;TYPE BAD OCTAL MESSAGE
4201  012064  000412                      BR      26$
4202  012066  005726              25$:    TST     (SP)+           ;DUMP BAD VALUE
4203  012070  104400  003111              TYPE    ,IVDWCT         ;TYPE INVALID WORD COUNT
4204  012074  000406                      BR      26$
4205  012076  104400  003136      27$:    TYPE    ,IVDNC          ;TYPE INVALID COMMAND IN NO CHECK
4206  012102  000403                      BR      26$
```

```
4207  012104  005726                    28$:   TST     (SP)+           ;DUMP BAD DRIVE NUMBER
4208  012106  104400  003161                   TYPE    ,BADDRV         ;TYPE MESSAGE
4209  012112                            20$:
4210  012112  011404                    26$:   MOV     (R4),R4         ;SET UP ERROR RETURN
4211  012114                            55$:
4212  012114  012602                           MOV     (SP)+,R2        ;;POP STACK INTO R2
4213  012116  012600                           MOV     (SP)+,R0        ;;POP STACK INTO R0
4214  012120  000204                           RTS     R4              ;RETURN
4215  012122  010146                    14$:   MOV     R1,-(SP)        ;PUT ADDRESS OF PARAM ON STACK
4216  012124  004737  030746                   JSR     PC,OCTBIN       ;CONVERT ASCII OCTAL TO BINARY
4217  012130  012060                           24$                     ;BAD CONVERT, NON OCTAL NUMBER
4218  012132  020027  000010                   CMP     R0,#10          ;WORD COUNT PARAM?
4219  012136  001010                           BNE     17$             ;IF NO, BRANCH. ELSE
4220  012140  021637  001700                   CMP     (SP),MAXWDS     ;TEST IF WORD COUNT IF TO BIG
4221  012144  101405                           BLOS    17$             ;NO - SKIP
4222  012146  032737  000004  001244          BIT     #BAII,OPFLGS    ;ELSE TEST IF BAI INHIBIT SET
4223  012154  001001                           BNE     17$             ;YES - BIG WORD COUNT OKAY
4224  012156  000743                           BR      25$             ;ELSE GO REPORT ERROR
4225  012160  020027  000002           17$:   CMP     R0,#2           ;CYL NUM OR OFFSET PARAM
4226  012164  001412                           BEQ     19$             ;IF YES, BRANCH. ELSE STORE
4227  012166  005700                           TST     R0              ;DRIVE PARAMETER?
4228  012170  001003                           BNE     32$             ;NO - SKIP LEGAL DRIVE TEST
4229  012172  021627  000007                   CMP     (SP),#7         ;TEST DRIVE PARAMETER
4230  012176  101342                           BHI     28$             ;TO BIG, ERROR
4231  012200  012660  001176           32$:   MOV     (SP)+,DRIVE(R0) ;CONVERTED VALUE IN PROPER STORAGE
4232  012204  062700  000002           18$:   ADD     #2,R0           ;BUMP R0 TO NEXT PARAM NUM
4233  012210  000655                           BR      10$             ;LOOP FOR NEXT PARAM
4234  012212  105737  001665           19$:   TSTB    OFFLAG          ;TEST IF OFFSET FLAG SET
4235  012216  001003                           BNE     40$             ;IF YES, BRANCH TO STORE OFFSET
4236  012220  012637  001200                   MOV     (SP)+,CYLNUM    ;STORE CYLINDER NUMBER
4237  012224  000767                           BR      18$             ;BR TO BUMP & LOOP
4238  012226  012637  001256           40$:   MOV     (SP)+,LOFFST    ;STORE OFFSET VALUE
4239  012232  000764                           BR      18$             ;BRANCH TO BUMP & LOOP
4240  012234  010325                    41$:   MOV     R3,(R5)+        ;INSERT JSR R4 CONSTANT
4241  012236  012225                           MOV     (R2)+,(R5)+     ;INSERT EXECUTE ADDRESS
4242  012240  011200                           MOV     (R2),R0         ;GET NUM OF PARAM FOR THIS CMND
4243  012242  142737  000010  001244          BICB    #SECT20,OPFLGS  ;CLEAR 24 SECTOR FLAG
4244  012250  022737  000026  001212          CMP     #26,FORMAT      ;CHECK IF THAT IS CORRECT
4245  012256  001403                           BEQ     46$             ;YEP - SKIP
4246  012260  152737  000010  001244          BISB    #SECT20,OPFLGS  ;NOPE - SET THE FLAG
4247  012266  105737  001666           46$:   TSTB    SAMDR           ;SAME DRIVE SWITCH SET?
4248  012272  001405                           BEQ     42$             ;
4249  012274  112725  177777                   MOVB    #-1,(R5)+       ;INSERT DRIVE NUM OF -1
4250  012300  105037  001666                   CLRB    SAMDR           ;CLEAR SAME DRIVE SWITCH
4251  012304  000402                           BR      43$
4252  012306  113725  001176           42$:   MOVB    DRIVE,(R5)+     ;INSERT DRIVE NUM PARAM
4253  012312  113725  001244           43$:   MOVB    OPFLGS,(R5)+    ;INSERT OPERATION FLAGS
4254  012316  005300                           DEC     R0              ;DEC PARAM NUMBER
4255  012320  001435                           BEQ     50$             ;IF NOW ZERO, EXIT
4256  012322  105737  001665                   TSTB    OFFLAG          ;TEST OFFSET FLAG SET
4257  012326  001405                           BEQ     44$
4258  012330  013725  001256                   MOV     LOFFST,(R5)+    ;INSERT OFFSET VALUE
4259  012334  105037  001665                   CLRB    OFFLAG          ;CLEAR OFFSET FLAG
4260  012340  000402                           BR      45$
4261  012342  013725  001200           44$:   MOV     CYLNUM,(R5)+    ;INSERT CYLINDER NUMBER
4262  012346  005300                    45$:   DEC     R0              ;DEC PARAM NUMBER
```

```
4263  012350  001421                          BEQ     50$             ;IF ZERO, EXIT
4264  012352  113725  001202                  MOVB    TRKNUM,(R5)+    ;INSERT TRACK NUMBER PARAM
4265  012356  005300                          DEC     R0              ;DEC PARAM NUMBER
4266  012360  001415                          BEQ     50$             ;IF ZERO, EXIT
4267  012362  113725  001204                  MOVB    SECNUM,(R5)+    ;INSERT SECTOR NUM PARAM
4268  012366  005300                          DEC     R0              ;DEC PARAM NUMBER
4269  012370  001411                          BEQ     50$             ;IF ZERO, EXIT
4270  012372  013725  001206                  MOV     WDCNT,(R5)+     ;INSERT WORD COUNT PARAM
4271  012376  023727  001702  042122          CMP     TEMP1,#"RD      ;TEST IF READ DATA COMMAND
4272  012404  001003                          BNE     50$             ;NO - BRANCH
4273  012406  013737  001206  001254          MOV     WDCNT,COMSZE    ;ELSE STORE WORD COUNT FOR DATA COMPARE
4274  012414  005724                  50$:    TST     (R4)+           ;SET UP NO ERROR RETURN
4275  012416  000636                          BR      55$
4276
4277                                  ;;*****************************************************************
4278                                  .SBTTL  OUTPUT TEST ROUTINE
4279                                  ;*      ENTRY:  JSR     R4,OTRTE
4280                                  ;*      RETURN: RTS     R4              ERROR RETURN
4281                                  ;*              RTS     R4+2            NORMAL RETURN
4282                                  ;*
4283                                  ;*THIS ROUTINE:
4284                                  ;*      1. DETERMINES WHICH FILE IS TO BE PUNCHE (SOURCE OR OBJECT)
4285                                  ;*      2. CHECKS IF THAT FILE HAS VALID CODE
4286                                  ;*      3. COMPUTES THE SIZE OF THAT FILE
4287                                  ;*      4. PUNCHES THE STORED PARAMETERS INCLUDING:
4288                                  ;*              A. THE TYPE OF CODE BEING PUNCHED (SOURCE OR OBJECT)
4289                                  ;*              B. THE SIZE OF THE FILE
4290                                  ;*              C$ WHICH OF THE USER DEFINED TEST PATTERNS HAVE
4291                                  ;*                BEEN DEFINED
4292                                  ;*              D. ALL OF THE STORED TEST SPECIFIC PARAMETERS (DRIVE,
4293                                  ;*                CYLINDER, TRACK, SECTOR, ITERATION COUNT, ETC.)
4294                                  ;*      5. CHECKS WHICH USER DEFINED OR RANDOM TEST PATTERNS HAS BEEN
4295                                  ;*         DEFINED AND PUNCHES THAT PATTERN.
4296                                  ;*      6.  PUNCHES THE SOURCE OR OBJECT FILE
4297                                  ;;*****************************************************************
4298
4299  012420                          OTRTE:
4300  012420  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
4301  012422  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
4302  012424  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
4303  012426  004737  004666                  JSR     PC,SBSCN                ;BUMP R1 TO NEXT PARAMETER
4304  012432  105711                          TSTB    (R1)                    ;TEST IF PARAM NULL
4305  012434  001526                          BEQ     20$                     ;IF YES, BRANCH TO ERROR
4306  012436  121127  000117                  CMPB    (R1),#'O                ;TEST IF PARAMETER IS "O"
4307  012442  001013                          BNE     2$
4308  012444  105737  001235                  TSTB    VLDOBJ                  ;TEST IF VALID OBJECT CODE
4309  012450  001523                          BEQ     21$                     ;BRANCH TO EXIT FOR ERROR
4310  012452  012737  043117  001174          MOV     #"OF,PUCODE             ;STORE OUTPUT DATA TYPE
4311  012460  013703  001712                  MOV     OFPTR,R3                ;GET OFILE POINTER(END OF FILE)
4312  012464  162703  034434                  SUB     #OFILE,R3               ;LENGTH OF FILE IS DIFFERENCE
4313  012470  000412                          BR      3$
4314  012472  105737  001234          2$:     TSTB    SFEMP                   ;TEST SOURCE FILE EMPTY
4315  012476  001113                          BNE     22$                     ;SFEMP SET, ERROR
4316  012500  012737  043123  001174          MOV     #"SF,PUCODE             ;SET PUNCH FILE CODE TO SOURCE
4317  012506  013703  001246                  MOV     SFPTR,R3                ;GET ADDR OF END OF SF+1(1ST EMPTY)
4318  012512  163703  001710                  SUB     IBUFPT,R3               ;DIFFERENCE IS SOURCE FILE LENGTH
```

```
4319  012516  010337  001250          3$:     MOV     R3,PUFLSZ               ;STORE BYTE COUNT IN PU FILE SIZE
4320  012522  012700  001174                  MOV     #PUCODE,R0              ;GET ADDR OF START OF STORED PARAM
4321  012526  012703  000066                  MOV     #66,R3                  ;SET PUNCH COUNT
4322  012532  004437  012752                  JSR     R4,PUNCH                ;GO PUNCH STORED PARAM
4323  012536  012734                          23$                             ;ERROR RETURN
4324  012540  012705  001237                  MOV     #PATXDF,R5              ;GET ADDR OF PAT DEFINED SWITCHES
4325  012544  105725                  TSTB    (R5)+                   ;TEST PAT X DEFINED
4326  012546  001407                          BEQ     4$                      ;IF NOT BRANCH, ELSE
4327  012550  012703  000100                  MOV     #100,R3                 ;SET R3 FOR PUNCH COUNT AND
4328  012554  012700  001262                  MOV     #PATX,R0                ;GET ADDR OF PATX
4329  012560  004437  012752                  JSR     R4,PUNCH                ;PUNCH PATTERN X
4330  012564  012734                          23$
4331  012566  105725                  4$:     TSTB    (R5)+                   ;TEST PAT Y DEFINED
4332  012570  001407                          BEQ     5$                      ;IF NOT SET, BRANCH. ELSE
4333  012572  012700  001362                  MOV     #PATY,R0                ;GET ADDR OF PAT Y AND
4334  012576  012703  000100                  MOV     #100,R3                 ;SET PUNCH COUNT AND
4335  012602  004437  012752                  JSR     R4,PUNCH                ;PUNCH IT
4336  012606  012734                          23$
4337  012610  105725                  5$:     TSTB    (R5)+                   ;TEST PAT Z DEFINED
4338  012612  001407                          BEQ     6$                      ;IF NOT, BRANCH. ELSE
4339  012614  012700  001462                  MOV     #PATZ,R0                        ;GET ADDR OF PAT Z AND
4340  012620  012703  000100                  MOV     #100,R3                 ;SET PUNCH COUNT AND
4341  012624  004437  012752                  JSR     R4,PUNCH                ;PUNCH PAT Z
4342  012630  012734                          23$                             ;ERROR RETURN
4343  012632  105715                  6$:     TSTB    (R5)                    ;TEST RANDOM PATTERN DEFINED
4344  012634  001407                          BEQ     7$                      ;IF NOT, BRANCH. ELSE
4345  012636  012703  000100                  MOV     #100,R3                 ;SET BYTE COUNT
4346  012642  012700  001562                  MOV     #PATR,R0                ;SET ADDRESS OF RANDOM PAT
4347  012646  004437  012752                  JSR     R4,PUNCH                ;PUNCH RANDOM PATTERN
4348  012652  012734                          23$                             ;ERROR RETURN
4349  012654  012700  034434          7$:     MOV     #OFILE,R0               ;GET ADDRESS OF OBJ FILE
4350  012660  023727  001174  043117        CMP     PUCODE,#"OF             ;TEST IF PUNCH CODE SAYS OBJ
4351  012666  001402                          BEQ     8$                      ;IF EQ, R0 IS CORRECT. ELSE
4352  012670  013700  001710                  MOV     IBUFPT,R0               ;LOAD R0 WITH ADDR OF SOURCE
4353  012674  013703  001250          8$:     MOV     PUFLSZ,R3               ;SET PUNCH COUNT
4354  012700  004437  012752                  JSR     R4,PUNCH                ;PUNCH FILE
4355  012704  012734                          23$                             ;ERROR RETURN
4356  012706  005724                          TST     (R4)+           ;SET NORMAL RETURN
4357  012710  000414                          BR      30$                     ;GO TO RETURN
4358  012712  104400  002624          20$:    TYPE    ,IVDPAR                 ;TYPE INVALID PARAM
4359  012716  000410                          BR      25$
4360  012720  104400  002764          21$:    TYPE    ,IVDRUN                 ;TYPE NO OBJECT CODE
4361  012724  000405                          BR      25$
4362  012726  104400  003015          22$:    TYPE    ,NOSRC                  ;TYPE NO SOURCE CODE
4363  012732  000402                          BR      25$
4364  012734  104400  003203          23$:    TYPE    ,PUERR                  ;TYPE PUNCH ERROR
4365  012740  011404                  25$:    MOV     (R4),R4                 ;SET ERROR RETURN
4366  012742                          30$:
4367  012742  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
4368  012744  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
4369  012746  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
4370  012750  000204                          RTS     R4                      ;RETURN
4371                          ;;*****************************************************************
4372                          .SBTTL  PAPER TAPE PUNCH ROUTINE
4373                          ;*      ENTRY:  JSR     R4,PUNCH
4374                          ;*      RETURN: RTS     R4              ERROR RETURN
```

```
                                        ;*              RTS     R4+2            NORMAL RETURN
                                        ;*THIS ROUTINE WILL PUNCH THE NUMBER OF BYTES INDICATED BY THE VALLE
                                        ;*IN R3 FROM THE MEMORY AREA POINTED TO BY R0.
                                        ;;*****************************************************************

 012752  042777  000100  166752 PUNCH:  BIC     #000100,@PTPSR          ;RESET INTERRUPT ENABLE
 012760  032777  100200  166744 1$:     BIT     #100200,@PTPSR          ;TEST FOR ERROR OR READY
 012766  001774                         BEQ     1$                      ;LOOP UNTIL ONE OR THE OTHER
 012770  100406                         BMI     20$                     ;BR TO ERROR EXIT IF ERROR
 012772  112077  166736                 MOVB    (R0)+,@PTPDB            ;LOAD BYTE FOR PUNCH
 012776  005303                         DEC     R3                      ;DEC BYTE COUNT
 013000  001367                         BNE     1$                      ;LOOP IF MORE TO PUNCH
 013002  005724                         TST     (R4)+                   ;SET GOOD RETURN
 013004  000204                         RTS     R4                      ;RETURN
 013006  011404              20$:       MOV     (R4),R4                 ;SET ERROR RETURN
 013010  000204                         RTS     R4                      ;RETURN
                                        ;;*****************************************************************
                                        .SBTTL  INPUT TEST AND INPUT STRING ROUTINE
                                        ;*      ENTRY:  JSR     R4,ITRTE OF ISRTE
                                        ;*      RETURN: RTS     R4              ERPOR RETURN
                                        ;*              RTS     R4+2            NORMAL RETURN
                                        ;*
                                        ;*THIS ROUTINE HAS TWO ENTRY POINTS.  THE ENTRY AT ISRTE SETS THE
                                        ;*CHAIN FLAG.
                                        ;*
                                        ;*FIRST THE STORED PARAMETERS ARE READ IN.  THE USER DEFINED
                                        ;*PATTERN FLAGS ARE CHECKED.  IF PATXDF IS NOT 0, 100(8)
                                        ;*BYTES ARE READ AND STORED IN PATX; IF PATYDF IS NOT 0,
                                        ;*100(8) BYTES ARE READ AND STORED IN PATY; ETC FOR PATTERN Z.
                                        ;*THE PUCODE IS THEN CHECKED FOR OF OR SF AND THE NUMBER OF BYTES
                                        ;*INDICATED IN PUFLSZ IS READ AND STORED IN THE APPROPRIATE
                                        ;*FILE.  THE OUTPUT BUFFER IS THEN INITIALIZED TO THE PATTERN
                                        ;*INDICATED IN PATSEL.
                                        ;;*****************************************************************

 013012  152737  000001  001663 ISRTE:  BISB    #1,CHNFLG               ;SET CHAIN FLAG
 013020                         ITRTE:
 013020  010046                         MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 013022  010346                         MOV     R3,-(SP)        ;;PUSH R3 ON STACK
 013024  010546                         MOV     R5,-(SP)        ;;PUSH R5 ON STACK
 013026  012700  001174                 MOV     #PUCODE,R0              ;GET ADDRESS OF STORED PARAM
 013032  012703  000066                 MOV     #66,R3                  ;SET NUMBER OF BYTES
 013036  004437  013336                 JSR     R4,READ                 ;READ PARAMETERS FROM TAPE
 013042  013314                         20$                             ;ERROR RETURN
 013044  012705  001237                 MOV     #PATXDF,R5              ;GET ADDR OF PATD DEFINED FLAGS
 013050  105725                         TSTB    (R5)+                   ;TEST PAT X DEFINED
 013052  001407                         BEQ     1$                      ;BRANCH IF NOT SET, ELSE
 013054  012703  000100                 MOV     #100,R3                 ;SET BYTE COUNT FOR PAT READ
 013060  012700  001262                 MOV     #PATX,R0                ;SET FOR READ INTO PAT X
 013064  004437  013374                 JSR     R4,RDCONT               ;READ IN PAT X
 013070  013314                         20$                             ;ERROR RETURN
 013072  105725              1$:        TSTB    (R5)+                   ;TEST PAT Y DEFINED
 013074  001407                         BEQ     2$                      ;IF NO, BRANCH. ELSE
 013076  012703  000100                 MOV     #100,R3                 ;SET BYTE COUNT
 013102  012700  001362                 MOV     #PATY,R0                ;SET FOR READ INTO PATY
 013106  004437  013374                 JSR     R4,RDCONT               ;READ IN PAT Y
```

```
4431  013112  013314                        2U$
4432  013114  105725                  2$:   TSTB    (R5)+           ;ERROR RETURN
4433  013116  001407                        BEQ     3$              ;TEST PAT Z DEFINED
4434  013120  012703  000100                MOV     #100,R3         ;IF NOT, BRANCH. ELSE
4435  013124  012700  001462                MOV     #PATZ,R0        ;SET BYTE COUNT
4436  013130  004437  013374                JSR     R4,RDCONT       ;SET FOR READ INTO PAT Z
4437  013134  013314                        2O$                     ;READ IN PAT Z
4438  013136  105715                  3$:   TSTB    (R5)            ;ERROR RETURN
4439  013140  001407                        BEQ     4$              ;TEST RANDOM PATTERN DEFINED
4440  013142  012703  000100                MOV     #100,R3         ;IF NOT, BRANCH. ELSE
4441  013146  012700  001562                MOV     #PATR,R0        ;SET BYTE COUNT
4442  013152  004437  013374                JSR     R4,RDCONT       ;SET FOR READ RANDOM PAT
4443  013156  013314                        2O$                     ;READ RANDOM PATTERN
4444  013160  152737  000377  001234  4$:   BISB    #377,SFEMP      ;ERROR RETURN
4445  013166  012700  034434                MOV     #OFILE,R0       ;SET NO SOURCE SWITCH
4446  013172  023727  001174  043117        CMP     PUCODE,#"OF     ;GET ADDR OF OFILE
4447  013200  001006                        BNE     8$              ;TEST IF CODE IS OBJECT
4448  013202  013737  001250  001712        MOV     PUFLSZ,OFPTR    ;BR IF NO, ELSE
4449  013210  060037  001712                ADD     R0,OFPTR        ;GET FILE SIZE
4450  013214  000415                        BR      5$              ;ADD IN START OF FILE ADDRESS
4451  013216  013700  001710          8$:   MOV     IBUFPT,R0       ;CORRECT R0 AND
4452  013222  105037  001234                CLRB    SFEMP           ;SOURCE FILE TO BE READ
4453  013226  105037  001235                CLRB    VLDOBJ          ;CLEAR VLDOBJ CAUSE SOURCE IS NEW
4454  013232  010003                        MOV     R0,R3           ;LOAD R3 WITH ADD OF SOURCE BUFF
4455  013234  063703  001250                ADD     PUFLSZ,R3       ;COMPUTE SIZE OF SOURCE
4456  013240  010337  001246                MOV     R3,SFPTR        ;STORE END OF SOURCE CODE ADD
4457  013244  105037  001663                CLRB    CHNFLG          ;CLEAR CHAIN FLAG
4458  013250  013703  001250          5$:   MOV     PUFLSZ,R3       ;GET SIZE OF DATA FILE
4459  013254  004437  013374                JSR     R4,RDCONT       ;READ FILE
4460  013260  013314                        2O$                     ;ERROR RETURN
4461  013262  113737  001236  013274        MOVB    PATSEL,6$       ;GET PATTERN SELECT INDEX
4462  013270  004437  020536                JSR     R4,ESBI         ;GO TO BUFFER INITIALIZE
4463  013274  000000            6$:   .WORD   0
4464  013276  105737  001663                TSTB    CHNFLG          ;TEST CHAIN FLAG
4465  013302  001402                        BEQ     7$
4466  013304  004437  010030                JSR     R4,RURTE        ;IF CHAIN FLAG SET GO TO RUN
4467  013310  005724            7$:   TST     (R4)+           ;SET NORMAL RETURN
4468  013312  000405                        BR      3O$
4469  013314  104400  003217          2O$:  TYPE    ,PRERR          ;TYPE READER ERROR
4470  013320  105037  001663                CLRB    CHNFLG          ;CLEAR CHAIN FLAG IF READ ERR
4471  013324  011404                        MOV     (R4),R4         ;SET ERROR RETURN
4472  013326                          3O$:
4473  013326  012605                        MOV     (SP)+,R5        ;;POP STACK INTO R5
4474  013330  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
4475  013332  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
4476  013334  000204                        RTS     R4              ;RETURN
4477                          ;;*****************************************************************
4478                          .SBTTL  PAPER TAPE PUNCH ROUTINE
4479                          ;*      ENTRY:  JSR     R4,READ
4480                          ;*      RETURN: RTS     R4              ERROR RETURN
4481                          ;*              RTS     R4+2            NORMAL RETURN
4482                          ;*
4483                          ;*THIS ROUTINE READS THE NUMBER OF BYTES INDICATED IN R3 AND
4484                          ;*STORES THEM IN CONSECUTIVE MEMORY LOCATIONS STARTING
4485                          ;*AT THE ADDRESS IN R0.
4486                          ;;*****************************************************************
```

```
 4487
 4488  013336  042777  000100  166362  READ:   BIC     #000100,@PTRSR          ;CLEAR INTERRUPT ENABLE
 4489  013344  005277  166356          WSTART: INC     @PTRSR                  ;SET READER GO BIT
 4490  013350  032777  100200  166350  WLOOP:  BIT     #100200,@PTRSR          ;TEST IF ERROR OR DONE
 4491  013356  001774                          BEQ     WLOOP                   ;LOOP
 4492  013360  100422                          BMI     XREAD                   ;GO TO ERROR EXIT IF ERROR
 4493  013362  117710  166342                  MOVB    @PTRDB,(R0)             ;EMPTY DATA BUFFER
 4494  013366  001766                          BEQ     WSTART                  ;IF ZERO GO READ NEXT
 4495  013370  105720                          TSTB    (R0)+                   ;ELSE BUMP R0
 4496  013372  005303                          DEC     R3                      ;COUNT THAT CHARACTER
 4497  013374  005277  166326          RDCONT: INC     @PTRSR                  ;SET READER GO BIT
 4498  013400  032777  100200  166320  RDLOOP: BIT     #100200,@PTRSR          ;TEST FOR ERROR OR DONE
 4499  013406  001774                          BEQ     RDLOOP                  ;LOOP IF NO BITS SET
 4500  013410  100406                          BMI     XREAD                   ;IF BIT IS SET GO TO ERROR
 4501  013412  117720  166312                  MOVB    @PTRDB,(R0)+            ;ELSE STORE BYTE READ
 4502  013416  005303                          DEC     R3                      ;DEC BYTE COUNT
 4503  013420  001365                          BNE     RDCONT                  ;IF NOT ZERO, LOOP
 4504  013422  005724                          TST     (R4)+                   ;SET NORMAL RETURN
 4505  013424  000204                          RTS     R4                      ;RETURN
 4506  013426  011404                  XREAD:  MOV     (R4),R4                 ;SET ERROR RETURN
 4507  013430  000204                          RTS     R4                      ;RETURN
 4508                                  ;*ENTRY:        JSR     R4,E$PM
 4509                                  ;*              (MESSAGE)
 4510                                  ;*RETURN:       RTS     R4
 4511                                  ;*THIS ROUTINE PRINTS THE MESSAGE FOUND IN THE OBJECT CODE
 4512                                  ;*FOLLOWING THE JSR TO E$PM. AFTER THE MESSAGE IS PRINTED R4 IS
 4513                                  ;*ADJUSTED TO SKIP OVER THE MESSAGE CONTENTS.
 4514                                  ;*ROUTINES CALLED:
 4515                                  ;*      TYPE
 4516                                  ;*
 4517                                  ;;*****************************************************************
 4518          013516                          .=13516
 4519  013516  005237  001116          E$PM:   INC     LINNUM                  ;INCREMENT PSUEDO LINE COUNT
 4520  013522  023737  001232  001714          CMP     ITCNT,CNTSTR            ;CHECK IF FIRST PASS?
 4521  013530  001011                          BNE     2$                      ;NO, SKIP PRINT
 4522  013532  005737  001722                  TST     LPCNT1                  ;TEST IF LOOP COUNT HAS COUNT
 4523  013536  001006                          BNE     2$                      ;YES - DON'T PRINT MESSAGE
 4524  013540  010437  013546                  MOV     R4,1$                   ;GET ADDRESS OF MESSAGE
 4525  013544  104400                          TYPE                            ;TYPE MESSAGE
 4526  013546  000000                  1$:     .WORD
 4527  013550  104400  001171                  TYPE    ,$CRLF                  ;TYPE CARRIAGE RETURN
 4528  013554  105724                  2$:     TSTB    (R4)+                   ;TEST FOR NULL (END OF MESSAGE)
 4529  013556  001376                          BNE     2$                      ;LOOP IF NOT THE END
 4530  013560  105714                          TSTB    (R4)                    ;TEST FOR SECOND NULL
 4531  013562  001001                          BNE     3$                      ;IF ONLY ONE NULL, EXIT
 4532  013564  105724                          TSTB    (R4)+                   ;BUMP PAST 2ND NULL
 4533                                                                          ;T[illegible] MAY BE PRESENT TO
 4534                                                                          ;INSURE WORD ALIGNMENT
 4535  013566  000204                  3$:     RTS     R4                      ;RETURN
 4536                                  ;;*****************************************************************
 4537                                  .SBTTL  COMMON DRIVER CALL
 4538                                  ;*ENTRY JSR     PC, DRVCAL WITH R5 POINTING TO ACTIVE PARAM BLK
 4539                                  ;*RETURN        RTS     PC
 4540                                  ;*
 4541                                  ;*THIS ROUTINE CLEARS THE DONE FLAG, CALLS THE DRIVER WITH
 4542                                  ;*THE ACTIVE PARAMETER BLOCK, AND CALLS THE WATCH DOG
```

```
                                        ;*TIMER.
                                        ;*
                                        ;*IT THEN LOOPS WAITING FOR "DONE" (INDICATING AN INTERRUPT
                                        ;*WITH OR WITHOUT ERROR) OR WAITING FOR CONTROLLER READY
                                        ;*(IF THE NO CHECK BIT IS SET IN THE ACTIVE PARAM
                                        ;*BLOCK). WHEN EITHER OCCURS THE ROUTINE RETURNS TO
                                        ;*THE CALLER.
                                        ;;*****************************************************************
013570  105037  001667          DRVCAL: CLRB    DONE            ;CLEAR DONE FLAG
013574  010537  013604                  MOV     R5,1$           ;GET PARAM BLOCK ADDRESS
013600  004737  027052                  JSR     PC,C.INIT       ;CALL DRIVER
013604  000000                  1$:     .WORD                   ;PARAM BLK ADDRESS PARAMETER
013606  004737  023422          2$:     JSR     PC,W.WTCH       ;CALL TIMER
013612  105737  001667                  TSTB    DONE            ;TEST IF DONE SET
013616  001011                          BNE     3$              ;IF YES, INT OCCURRED, OPCOMP.
013620  032765  000400  000014          BIT     #NOCHK,P.PRST(R5) ;TEST IF NO CHECK OPERATION
013626  001767                          BEQ     2$              ;NO CHECK OFF, MUST GET DONE
013630  013702  023342                  MOV     RKBAS,R2        ;GET RKBASE ADDRESS
013634  105762  000000                  TSTB    RKCS1(R2)       ;TEST FOR READY
013640  100362                          BPL     2$              ;NOT READY, LOOP
013642  000207                  3$:     RTS     PC              ;RETURN
                                        ;;*****************************************************************
                                        .SBTTL  ERROR FREE RETURN FROM DRIVER
                                        ;*ENTRY JSR     PC, ERRFRE WITH R5 POINTING TO ACTIVE PARAM BLK
                                        ;*RETURN        RTS     PC
                                        ;*THIS ROUTINE SETS THE DONE FLAG AND RETURNS TO
                                        ;*THE INTERRUPT HANDLER.
                                        ;;*****************************************************************
013644  105137  001667          ERRFRE: COMB    DONE            ;SET DONE FLAG
013650  000207                          RTS     PC              ;RETURN
                                        ;;*****************************************************************
                                        .SBTTL  ERROR RETURN FROM DRIVER (DRIVE ERROR)
                                        ;*ENTRY JSR     PC, DRVERR WITH R5 POINTING TO ACTIVE PARAM BLK
                                        ;*RETURN        RTS     PC
                                        ;*
                                        ;*THIS ROUTINE IS CALLED BY THE DRIVER WHEN AN ERROR
                                        ;*OCCURS IN THE SUBSYSTEM OPERATION OR WHEN A TIMEOUT
                                        ;*OCCURS.
                                        ;*
                                        ;*ALL SUBSYSTEM ERROR REPORTS ARE GENERATED IN THIS ROUTINE.
                                        ;*THE ERROR REPORTS WILL CONTAIN PREVIOUS OPERATION
                                        ;*INFORMATION AS WELL AS INFORMATION ABOUT THE FAILING
                                        ;*OPERATION. FAILURE INFORMATION IS TAKEN FROM THE
                                        ;*ACTIVE PARAM BLOCK AND PREVIOUS OPERATION INFORMATION
                                        ;*IS TAKEN FROM THE INACTIVE BLOCK.
                                        ;*
                                        ;*THE WORD IN LOCATION 72 OF THE PARAMETER BLOCK IS
                                        ;*A PRINT CONTROL WORD. THIS WORD  IS SET UP WHEN THE
                                        ;*PARAMETER BLOCK IS LOADED BEFORE THE CALL TO THE
                                        ;*DRIVER AND TELLS THE ERROR REPORT WHICH GROUPS OF
                                        ;*DATA IN THE PARAMETER BLOCK IS PERTINENT TO THIS
                                        ;*OPERATION. THIS ALLOWS THE REPORT TO BE CUSTOMIZED
                                        ;*AND SPECIFIC FOR THIS ERROR.
                                        ;*
                                        ;*EACH BIT IN THE PRINT CONTROL WORD INDICATES THAT A
                                        ;*GROUP OF WORDS OR BYTES IS OR IS NOT TO BE INCLUDED IN
```

```
4599                                    ;*THE REPORT. IF SET THAT GROUP IS INCLUDED. THE PARAMETER
4600                                    ;*BLOCK ELEMENTS TO PRINT CONTROL WORD BIT ASSIGNMENTS ARE
4601                                    ;*AS FOLLOWS:
4602                                    ;*          BIT          PARAMATER BLOCK ELEMENT
4603                                    ;*          ---          ------------------------
4604                                    ;*           0                   WORD 2
4605                                    ;*                               BYTE 4,5,&6
4606                                    ;*           1                   BYTE 7
4607                                    ;*                               WORD 10 & 12
4608                                    ;*           2                   WORD 16 & 20
4609                                    ;*           3                   WORD 22 & 24
4610                                    ;*           4                   WORD 26 & 30
4611                                    ;*           5                   WORD 32
4612                                    ;*           6                   WORD 34 THRU 62
4613                                    ;*
4614                                    ;*THE ERROR REPORT CAN BE ABBREVIATED TO THE FAILING
4615                                    ;*LINE NUMBER, THE DRIVE, AND COMMAND BY SETTING
4616                                    ;*SWITCH REGISTER 0. THE REST OF THE ERROR REPORT
4617                                    ;*IS SUPPRESSED, THE OTHER ERROR REPORTING CONTROLS
4618                                    ;*ARE STANDARD
4619                                    ;*****************************************************************
4620  013652  041412  051125  042522  CUROP:  .ASCIZ  <12>/CURRENT OPERATION:/<15><12>
4621  013660  052116  047440  042520
4622  013666  040522  044524  047117
4623  013674  006472  000012
4624  013700  050012  042522  044526  PREVOP: .ASCIZ  <12>/PREVIOUS OPERATION:/<15><12>
4625  013706  052517  020123  050117
4626  013714  051105  052101  047511
4627  013722  035116  005015     000
4628  013727     012  050101  046120  APRES:  .ASCIZ  <12>/APPLICABLE REGS:/<15><12>
4629  013734  041511  041101  042514
4630  013742  051040  043505  035123
4631  013750  005015     000
4632  013753     012  040520  040522  PARMGVN: .ASCIZ <12>/PARAMETERS GIVEN:/<15><12>
4633  013760  042515  042524  051522
4634  013766  043440  053111  047105
4635  013774  006472  000012
4636  014000  051412  041125  054523  SSTO:   .ASCIZ  <12>/SUBSYSTEM TIMEOUT/<15><12>
4637  014006  052123  046505  052040
4638  014014  046511  047505  052125
4639  014022  005015     000
4640  014025     012  052523  051502  SDETER: .ASCIZ  <12>/SUBSYSTEM DETECTED ERROR/<15><12>
4641  014032  051531  042524  020115
4642  014040  042504  042524  052103
4643  014046  042105  042440  051122
4644  014054  051117  005015     000
4645  014061     075  046503  042116  PLINE:  .ASCIZ  /=CMND LINE NUM /<15><12>
4646  014066  046040  047111  020105
4647  014074  052516  020115  005015
4648  014102     000
4649  014103     104  044522  042526  FDRIVE: .ASCIZ  /DRIVE=/
4650  014110  000075
4651  014112  046503  042116  000075  FCMND:  .ASCIZ  /CMND=/
4652  014120  054503  047114  051104  FPARM1: .ASCIZ  /CYLNDR  SECTOR  TRACK   OFFSET  /
4653  014126  020040  042523  052103
4654  014134  051117  020040  051124
```

```
 4655  014142  041501  020113  020040
 4656  014150  043117  051506  052105
 4657  014156  020040     000
 4658  014161     102  044101  020040  FPARM2: .ASCIZ  /BAH     BAL     WDC/
 4659  014166  020040  041040  046101
 4660  014174  020040  020040  053440
 4661  014202  041504     000
 4662  014205     103  030523  020040  FPARM3: .ASCIZ  /CS1     CS2     /
 4663  014212  020040  041440  031123
 4664  014220  020040  020040  000040
 4665  014226  041527  020040  020040  FPARM4: .ASCIZ  /WC      BA      /
 4666  014234  020040  040502  020040
 4667  014242  020040  020040     000
 4668  014247     104  020101  020040  FPARM5: .ASCIZ  /DA      DC      /
 4669  014254  020040  042040  020103
 4670  014262  020040  020040  000040
 4671  014270  051501  043117     000  FPARM6: .ASCIZ  /ASOF/
 4672  014275     105  020122  020040  FPARM7: .ASCIZ  /ER      DS      A0      B0/<15><12>
 4673  014302  020040  042040  020123
 4674  014310  020040  020040  040440
 4675  014316  020060  020040  020040
 4676  014324  041040  006460  000012
 4677  014332  030501  020040  020040  FPARM8: .ASCII  &A1      B1      A2      B2      A3      B3      &
 4678  014340  020040  030502  020040
 4679  014346  020040  020040  031101
 4680  014354  020040  020040  020040
 4681  014362  031102  020040  020040
 4682  014370  020040  031501  020040
 4683  014376  020040  020040  031502
 4684  014404  020040  020040  020040
 4685  014412  041505  027503  047520          .ASCIZ  &ECC/POS ECC/PAT&<15><12>
 4686  014420  020123  041505  027503
 4687  014426  040520  006524  000012
 4688  014434  042012  052101  020101  FDATC:  .ASCIZ  <12>/DATA COMPARE ERR ON WORD /
 4689  014442  047503  050115  051101
 4690  014450  020105  051105  020122
 4691  014456  047117  053440  051117
 4692  014464  020104     000
 4693  014467     075  047507  042117  GDDAT:  .ASCIZ  /=GOOD WORD/<15><12>
 4694  014474  053440  051117  006504
 4695  014502  000012
 4696  014504  041075  042101  053440  BDDAT:  .ASCIZ  /=BAD WORD/<15><12>
 4697  014512  051117  006504  000012
 4698  014520  051412  040524  052524  SCERR:  .ASCIZ  <12>/STATUS COMPARE ERROR/<15><12>
 4699  014526  020123  047503  050115
 4700  014534  051101  020105  051105
 4701  014542  047522  006522  000012
 4702  014550  051475  040524  052524  STWDNM: .ASCIZ  /=STATUS WD NUM/<15><12>
 4703  014556  020123  042127  047040
 4704  014564  046525  005015     000
 4705  014571     075  040515  045523  MSKLAB: .ASCIZ  /=MASK/<15><12>
 4706  014576  005015     000
 4707  014601     012  042522  044507  RCERR:  .ASCIZ  <12>/REGISTER COMPARE ERROR/<15><12>
 4708  014606  052123  051105  041440
 4709  014614  046517  040520  042522
 4710  014622  042440  051122  051117
```

```
 4711  014630  005015     000
 4712  014633     075  042522  020107  REGLAB: .ASCIZ  /=REG NUM/<15><12>
 4713  014640  052516  006515  000012
 4714  014646  051104  053111  020105  HARDER: .ASCIZ  /DRIVE HAS HARD ERROR/<15><12>
 4715  014654  040510  020123  040510
 4716  014662  042122  042440  051122
 4717  014670  051117  005015     000
 4718  014675     123  040524  052524  SCNCLR: .ASCIZ  /STATUS CHANGE NOT CLEARED/<15><12>
 4719  014702  020123  044103  047101
 4720  014710  042507  047040  052117
 4721  014716  041440  042514  051101
 4722  014724  042105  005015     000
 4723  014731     116  020117  051104  DSCNOT: .ASCIZ  /NO DRIVE STATUS CHANGE/<15><12>
 4724  014736  053111  020105  052123
 4725  014744  052101  051525  041440
 4726  014752  040510  043516  006505
 4727  014760  000012
 4728  014762  042523  055111  042105  STILSZ: .ASCIZ  /SEIZED OTHER PORT/<15><12>
 4729  014770  047440  044124  051105
 4730  014776  050040  051117  006524
 4731  015004  000012
 4732  015006  047125  054105  042520  BADATT: .ASCIZ  /UNEXPECTED ATTENTION/<15><12>
 4733  015014  052103  042105  040440
 4734  015022  052124  047105  044524
 4735  015030  047117  005015     000
 4736          015036                  .EVEN
 4737  015036                  DRVERR:
 4738  015036  010046          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 4739  015040  010146          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 4740  015042  010246          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
 4741  015044  010346          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
 4742  015046  010446          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
 4743  015050  105037  001671  CLRB    RPSWIT          ;CLEAR REPORT PASS SWITCH
 4744  015054  152737  000377  001667  BISB    #377,DONE       ;SET DONE FLAG
 4745  015062  032777  002000  164050  BIT     #SW10,@SWR      ;BELL ON ERROR?
 4746  015070  001402          BEQ     1$              ;NO - BRANCH
 4747  015072  104400  001164  TYPE    ,$BELL          ;RING BELL
 4748  015076  032777  020000  164034  1$:     BIT     #SW13,@SWR      ;INHIBIT ERROR REPORT?
 4749  015104  001407          BEQ     56$
 4750  015106  013702  023342          MOV     RKBAS,R2        ;GET BASE
 4751  015112  012762  100000  000000  MOV     #CCLR,RKCS1(R2)  ;RESET IE AND DO CONT CLEAR
 4752  015120  000137  015524          JMP     36$             ;JUMP TO EXIT
 4753  015124  105737  023240  56$:    TSTB    CEFLG                   ;TEST CONTROLLER ERROR FLAG
 4754  015130  001052          BNE     3$                      ;SKIP TO TYPE OUT
 4755  015132  032765  000010  000014  BIT     #UEXATT,P.PRST(R5)  ;TEST UNEXPECTED ATTENTION
 4756  015140  001405          BEQ     55$
 4757  015142  104400  015006          TYPE    ,BADATT         ;UNEXPECTED ATTENTION REPORT
 4758  015146  012765  000044  000064  MOV     #44,PRTCON(R5)  ;SET PRINT CONTROL TO LIMIT REPORT
 4759  015154  032765  000020  000014  55$:    BIT     #DRVHRD,P.PRST(R5) ;TEST DRIVE HARD ERROR
 4760  015162  001402          BEQ     50$             ;NO - BRANCH
 4761  015164  104400  014646          TYPE    ,HARDER         ;HARD ERROR REPORT
 4762  015170  032765  000040  000014  50$:    BIT     #DRVDSC,P.PRST(R5) ;TEST STATUS NOT CLEARED ERROR
 4763  015176  001402          BEQ     51$             ;NO - BRANCH
 4764  015200  104400  014675          TYPE    ,SCNCLR         ;REPORT STATUS NOT CLEARED ERROR
 4765  015204  032765  004000  000014  51$:    BIT     #NODSC,P.PRST(R5)  ;TEST NO STATUS CHANGE
 4766  015212  001402          BEQ     52$             ;NO - BRANCH
```

```
 4767  015214  104400  014731                  TYPE    ,DSCNOT             ;REPORT NO STATUS CHANGE ERROR
 4768  015220  032765  010000  000014  52$:     BIT     #DRVSZD,P.PRST(R5) ;TEST DRIVE SEIZED
 4769  015226  001402                          BEQ     54$                 ;NO - BRANCH
 4770  015230  104400  014762                  TYPE    ,STILSZ             ;REPORT STILL SEIZED ERROR
 4771  015234  032765  010100  000014  54$:     BIT     #CMDTO!DRVSZD,P.PRST(R5) ;TEST TIMEOUT
 4772  015242  001403                          BEQ     2$                  ;BR IF NO
 4773  015244  104400  014000                  TYPE    ,SSTO               ;TYPE SUBSYSTEM TIMEOUT
 4774  015250  000402                          BR      3$                  ;
 4775  015252  104400  014025          2$:     TYPE    ,SDETER             ;TYPE SUBSYSTEM DET. ERR.
 4776  015256  005002          3$:     CLR     R2                  ;CLEAR R2 (BYTE PRINT SWITCH)
 4777  015260  013746  001116                  MOV     LINNUM,-(SP)        ;PUT LINE NUMBER ON STACK
 4778  015264  104404                          TYPDS                       ;PRINT IT
 4779  015266  104400  014061                  TYPE    ,PLINE              ;TYPE LINE NUM MESSAGE
 4780  015272  010500          19$:    MOV     R5,R0               ;SET R0 TO BEGINNING OF PARAM BLK
 4781  015274  104400  014103                  TYPE    ,FDRIVE             ;TYPE "DRIVE"
 4782  015300  005102                          COM     R2                  ;SET R2 FOR BYTE OPERATION
 4783  015302  104413                          FPRINT                      ;PRINT FIRST LINE (DRIVE NUM)
 4784  015304  104400  001171                  TYPE    ,$CRLF
 4785  015310  104400  014112                  TYPE    ,FCMND              ;TYPE "COMMAND"
 4786  015314  104413                          FPRINT                      ;PRINT 2ND LINE (COMMAND)
 4787  015316  104400  001171                  TYPE    ,$CRLF
 4788  015322  032765  000100  000014          BIT     #CMDTO,P.PRST(R5) ;TEST TIMEOUT
 4789  015330  001402                          BEQ     25$
 4790  015332  004737  016120                  JSR     PC,READRG           ;GO READ REGISTERS
 4791  015336  105737  001671          25$:    TSTB    RPSWIT              ;TEST 2ND PASS
 4792  015342  001006                          BNE     17$                 ;YES - SKIP (SHORT FORM TEST)
 4793  015344  032777  000001  163566          BIT     #SWO,@SWR           ;TEST IF SHORT REPORT SWITCH SET
 4794  015352  001062                          BNE     30$                 ;YES - EXIT
 4795  015354  104400  013652                  TYPE    ,CUROP              ;TYPE "CURRENT OPERATION"
 4796  015360  016503  000064          17$:    MOV     PRTCON(R5),R3       ;GET PRINT CONTROL WORD
 4797  015364  032703  000003                  BIT     #3,R3               ;TEST IF GROUP 0 OR 1 TO BE PRINTED
 4798  015370  001001                          BNE     4$                  ;YES SKIP TO PRINTING
 4799  015372  000443                          BR      6$                  ;SKIP TO NEXT REPORT GROUP TEST
 4800  015374  104400  013753          4$:     TYPE    ,PARMG/N            ;TYPE HEADING "PARAM GIVEN"
 4801  015400  032703  000001                  BIT     #BIT0,R3            ;TEST FOR GROUP 0
 4802  015404  001402                          BEQ     46$
 4803  015406  104400  014120                  TYPE    ,FPARM1             ;TYPE GROUP 0 HEADINGS
 4804  015412  032703  000002          46$:    BIT     #BIT1,R3            ;TEST FOR GROUP 1
 4805  015416  001402                          BEQ     45$
 4806  015420  104400  014161                  TYPE    ,FPARM2             ;TYPE GROUP 1 HEADINGS
 4807  015424  104400  001171          45$:    TYPE    ,$CRLF
 4808  015430  032703  000001                  BIT     #BIT0,R3            ;TEST IF MUST PRINT GROUP 0
 4809  015434  001406                          BEQ     5$
 4810  015436  005002                          CLR     R2                  ;CLEAR BYTE REPORT CONTROL
 4811  015440  104413                          FPRINT                      ;PRINT LINE (CYLINDER)
 4812  015442  005102                          COM     R2                  ;SET BYTE CONTROL
 4813  015444  104413                          FPRINT                      ;PRINT LINE (SECTOR)
 4814  015446  104413                          FPRINT                      ;PRINT LINE (TRACK)
 4815  015450  104413                          FPRINT                      ;PRINT LINE (OFFSET)
 4816  015452  010500          5$:     MOV     R5,R0               ;SET TABLE POINTER FOR GROUP 1
 4817  015454  062700  000007                  ADD     #P.BAHI,R0          ;SET ADDRESS
 4818  015460  032703  000002                  BIT     #BIT1,R3            ;TEST IF MUST PRINT GROUP 1
 4819  015464  001406                          BEQ     6$                  ;NO, SKIP TO NEXT GROUP TEST
 4820  015466  052702  000001                  BIS     #1,R2               ;MAKE SURE R2 SAYS BYTE
 4821  015472  104413                          FPRINT                      ;PRINT LINE (BUS ADD HI)
 4822  015474  005002                          CLR     R2                  ;R2 SAYS WORD
```

```
 4823  015476  104413                        FPRINT                  ;PRINT LINE (BUS ADD LO)
 4824  015500  104413                        FPRINT                  ;PRINT LINE (WORD COUNT)
 4825  015502  104400  001171          6$:   TYPE    ,$CRLF
 4826  015506  104400  001172                TYPE    ,$LF
 4827  015512  105737  001671                TSTB    @PSWIT          ;TEST 2ND PASS
 4828  015516  001412                        BEQ     31$             ;NO - BRANCH AROUND EXIT
 4829  015520  004737  016670          30$:  JSR     PC,PRLPCT               ;PRINT LOOP COUNT
 4830  015524                          36$:
 4831  015524  012604                        MOV     (SP)+,R4        ;;POP STACK INTO R4
 4832  015526  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
 4833  015530  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
 4834  015532  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
 4835  015534  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
 4836  015536  004737  016622                JSR     PC,SWCONT       ;GO TO SWR LOOP CONTROL
 4837  015542  000207                        RTS     PC              ;RETURN
 4838  015544  104400  013727          31$:  TYPE    ,APRES          ;TYPE "APPLICABLE REGS"
 4839  015550  005002                        CLR     R2              ;MAKE SURE R2 SAYS WORD
 4840  015552  032703  000004                BIT     #BIT2,R3        ;TEST IF MUST PRINT GROUP 2
 4841  015556  001402                        BEQ     40$
 4842  015560  104400  014205                TYPE    ,FPARM3         ;TYPE GROUP 2 HEADINGS
 4843  015564  032703  000010          40$:  BIT     #BIT3,R3        ;TEST FOR GROUP 3
 4844  015570  001402                        BEQ     41$
 4845  015572  104400  014226                TYPE    ,FPARM4         ;TYPE GROUP 3 HEADINGS
 4846  015576  032703  000020          41$:  BIT     #BIT4,R3        ;TEST FOR GROUP 4
 4847  015602  001402                        BEQ     42$
 4848  015604  104400  014247                TYPE    ,FPARM5         ;TYPE GROUP 4 HEADINGS
 4849  015610  032703  000040          42$:  BIT     #BIT5,R3        ;TEST FOR GROUP 5
 4850  015614  001402                        BEQ     43$
 4851  015616  104400  014270                TYPE    ,FPARM6         ;TYPE GROUP 5 HEADINGS
 4852  015622  104400  001171          43$:  TYPE    ,$CRLF
 4853  015626  032703  000004                BIT     #BIT2,R3        ;TEST IF MUST PRINT GROUP 2
 4854  015632  001405                        BEQ     7$              ;NO - BRANCH
 4855  015634  010500                        MOV     R5,R0           ;SET TABLE PTR TO GROUP 2 VALUES
 4856  015636  062700  000016                ADD     #P.CS1,R0       ;SET ADDRESS
 4857  015642  104413                        FPPINT                  ;PRINT LINE (CS1)
 4858  015644  104413                        FPRINT                  ;PRINT LINE (CS2)
 4859  015646  032703  000010          7$:   BIT     #BIT3,R3        ;TEST IF MUST PRINT GROUP 3
 4860  015652  001405                        BEQ     8$              ;NO - BRANCH
 4861  015654  010500                        MOV     R5,R0           ;SET TABLE PTR TO GROUP 3 VALUES
 4862  015656  062700  000022                ADD     #P.WCR,R0       ;SET ADDRESS
 4863  015662  104413                        FPRINT                  ;PRINT LINE (WORD COUNT)
 4864  015664  104413                        FPRINT                  ;PRINT LINE (BUFFER ADD)
 4865  015666  032703  000020          8$:   BIT     #BIT4,R3        ;TEST IF MUST PRINT GROUP 4
 4866  015672  001405                        BEQ     9$              ;NO - BRANCH
 4867  015674  010500                        MOV     R5,R0           ;SET TABLE PTR TO GROUP 4 VALUES
 4868  015676  062700  000026                ADD     #P.DTS,R0       ;SET ADDRESS
 4869  015702  104413                        FPRINT                  ;PRINT LINE (DESIRED TRACK/SEC)
 4870  015704  104413                        FPRINT                  ;PRINT LINE (DESIRED CYLINDER)
 4871  015706  032703  000040          9$:   BIT     #BIT5,R3        ;TEST IF MUST PRINT GROUP 5
 4872  015712  001404                        BEQ     10$             ;NO - BR
 4873  015714  010500                        MOV     R5,R0           ;SET PTR TO GROUP 5 VALUE
 4874  015716  062700  000032                ADD     #P.ASOF,R0      ;SET ADDRESS
 4875  015722  104413                        FPRINT                  ;PRINT LINE (ATTN SUM & OFFSET)
 4876  015724  104400  001171          10$:  TYPE    ,$CRLF
 4877  015730  104400  001172                TYPE    ,$LF
 4878  015734  032703  000100                BIT     #BIT6,R3        ;TEST IF MUST PRINT GROUP 6
```

```
 4879  015740  001427                          BEQ     15$             ;NO - BRANCH
 4880  015742  012704  000004                  MOV     #4,R4           ;SET COL COUNT TO 4
 4881  015746  104400  014275                  TYPE    ,FPARM7         ;TYPE GROUP 6 HEADINGS
 4882  015752  010500                          MOV     R5,R0           ;SET TABLE POINTER TO GROUP 6 VALUES
 4883  015754  062700  000034                  ADD     #P.ER,R0        ;SET ADDRESS
 4884  015760  104413                  11$:    FPRINT                  ;PRINT COLUMN
 4885  015762  005304                          DEC     R4              ;DEC COUNT, LOOP UNTIL ZERO
 4886  015764  001375                          BNE     11$
 4887  015766  104400  001171                  TYPE    ,$CRLF
 4888  015772  012704  000010                  MOV     #10,R4          ;SET COL COUNT TO 10
 4889  015776  104400  014332                  TYPE    ,FPARM8         ;TYPE GROUP 7 HEADINGS
 4890  016002  104413                  12$:    FPRINT                  ;PRINT COL
 4891  016004  005304                          DEC     R4              ;DEC COUNT, LOOP UNTIL ZERO
 4892  016006  001375                          BNE     12$
 4893  016010  104400  001171                  TYPE    ,$CRLF
 4894  016014  104400  001172                  TYPE    ,$LF
 4895  016020  023727  001116  000001  15$:    CMP     LINNUM,#1       ;TEST IF FIRST LINE IS ERROR
 4896  016026  101421                          BLOS    18$             ;IF YES BYPASS 2ND PASS (PREV OP)
 4897  016030  105737  001671                  TSTB    RPSWIT          ;TEST IF SECOND PASS IN REPORT
 4898  016034  001016                          BNE     18$             ;YES - EXIT
 4899  016036  105137  001671                  COMB    RPSWIT          ;SET SWITCH
 4900  016042  012705  002142                  MOV     #PARM0,R5       ;SET R5 TO PARAM 0
 4901  016046  105737  001670                  TSTB    PBSW            ;WAS THAT RIGHT?
 4902  016052  001002                          BNE     16$             ;YES - SKIP
 4903  016054  012705  002230                  MOV     #PARM1,R5       ;NO - SET TO PARAM 1
 4904  016060  104400  013700          16$:    TYPE    ,PREVOP         ;TYPE "PREVIOUS OP" HDR
 4905  016064  005002                          CLR     R2
 4906  016066  000137  015272                  JMP     19$             ;GO PRINT PREVIOUS OPS.
 4907  016072  105037  001671          18$:    CLRB    RPSWIT          ;RESET 2ND PASS SWITCH
 4908  016076  012705  002142                  MOV     #PARM0,R5       ;SET R5 TO PARAM 0
 4909  016102  105737  001670                  TSTB    PBSW            ;TEST PB SWITCH. WAS THAT RIGHT?
 4910  016106  001402                          BEQ     20$             ;YES - SKIP
 4911  016110  012705  002230                  MOV     #PARM1,R5       ;NO - SET R5 TO PARAM 1
 4912  016114  000137  015520          20$:    JMP     30$             ;GO TO EXIT
 4913                                  ;;**********************************************************************
 4914                                  ;:**********************************************************************
 4915                                  .SBTTL  READ RK611 AND DRIVE STATUS REGISTER ROUTINE
 4916                                  ;*ENTRY:        JSR     PC,READRG
 4917                                  ;*RETURN:       RTS     PC
 4918                                  ;*
 4919                                  ;*THIS ROUTINE READS ALL THE RK611 REGISTERS AND ENTERS THEM INTO THE
 4920                                  ;*PARAMETER BLOCK. IT THEN TRIES TO READ THE DRIVE STATUS REGISTERS
 4921                                  ;*AND IF SUCCESSFUL, PUTS THEM INTO THE BLOCK. IF AND ERROR OCCURS
 4922                                  ;*WHILE READING DRIVE REGISTERS, A MESSAGE IS PRINTED TO ALERT THE
 4923                                  ;*USER THAT THE DRIVE STATUS IS NOT VALID.
 4924                                  ;:**********************************************************************
 4925  016120                          READRG:
 4926  016120  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 4927  016122  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
 4928  016124  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
 4929  016126  012705  002142                  MOV     #PARM0,R5       ;GET ADDRESS OF PARM0
 4930  016132  105737  001670                  TSTB    PBSW            ;IS PARM0 PRESENTLY SELECTED?
 4931  016136  001402                          BEQ     8$              ;YES - SKIP
 4932  016140  012705  002230                  MOV     #PARM1,R5       ;ELSE GET ADDRESS OF PARM 1
 4933  016144  010500                  8$:     MOV     R5,R0           ;GET PARAM BLOCK ADDRESS
 4934  016146  013702  023342                  MOV     RKBAS,R2        ;GET RK BASE ADDRESS
```

```
4935  016152  062700  000016                    ADD     #P.CS1,R0       ;BUMP BASE TO REG RETURN ENTRIES
4936  016156  016220  000000                    MOV     RKCS1(R2),(R0)+ ;GET CS1
4937  016162  016220  000010                    MOV     RKCS2(R2),(R0)+ ;GET CS2
4938  016166  016220  000002                    MOV     RKWC(R2),(R0)+  ;GET WORD COUNT
4939  016172  016220  000004                    MOV     RKBA(R2),(R0)+  ;GET BUFFER ADDRESS
4940  016176  016220  000006                    MOV     RKDA(R2),(R0)+  ;GET DESIRED ADDRESS
4941  016202  016220  000020                    MOV     RKDC(R2),(R0)+  ;GET DESIRED CYLINDER
4942  016206  016220  000016                    MOV     RKASOF(R2),(R0)+ ;GET ATTENTION
4943  016212  016220  000014                    MOV     RKER(R2),(R0)+  ;GET ERROR REG
4944  016216  016220  000012                    MOV     RKDS(R2),(R0)+  ;RET STATUS REG
4945  016222  005065  000014                    CLR     P.PRST(R5)      ;CLEAR PROG STATUS
4946  016226  012737  016416  023352            MOV     #6$,A.ABNL      ;SET TIME OUT RETURN
4947  016234  016203  000010                    MOV     RKCS2(R2),R3    ;GET CS2 TO STORE DRIVE NUM
4948  016240  012762  100000  000000            MOV     #CCLR,RKCS1(R2) ;SET CLEAR CONTROLLER
4949  016246  042703  177770                    BIC     #177770,R3      ;CLEAR ALL BUT DRIVE NUMBER
4950  016252  050362  000010                    BIS     R3,RKCS2(R2)    ;INSERT DRIVE NUMBER
4951  016256  005003                            CLR     R3
4952  016260  110362  000026          3$:       MOVB    R3,RKMR1(R2)    ;SET DESIRED STATUS BYTE SELECT
4953  016264  012762  000001  000000            MOV     #1,RKCS1(R2)    ;DO DRIVE SELECT
4954  016272  013737  023366  023420            MOV     W.SEC,W.DRV     ;SET UP TIMEOUT DURATION
4955  016300  152737  000377  023404            BISB    #377,W.TIME     ;GIVE WATCH DOG SOMETHING TO TIME
4956  016306  032762  000200  000000  1$:       BIT     #RDY,RKCS1(R2)  ;TEST READY
4957  016314  001007                            BNE     2$
4958  016316  004737  023422                    JSR     PC,W.WTCH       ;CALL HIM
4959  016322  032765  000100  000052            BIT     #CMDTO,P.PRST(R5)       ;DID HE TIME OUT?
4960  016330  001033                            BNE     5$              ;YES- TYPE WARNING AND EXIT
4961  016332  000765                            BR      1$              ;NO - GO TEST READY AGAIN
4962  016334  005762  000000          2$:       TST     RKCS1(R2)       ;TEST CS1 FOR ERROR
4963  016340  100427                            BMI     5$              ;YES - SKIP
4964  016342  032762  000001  000012            BIT     #DRA,RKDS(R2)   ;TEST IF DRIVE AVAILABLE
4965  016350  001423                            BEQ     5$              ;NO - PRINT WARNING
4966  016352  016220  000034                    MOV     RKMR2(R2),(R0)+ ;STORE BYTE A
4967  016356  016220  000036                    MOV     RKMR3(R2),(R0)+ ;STORE BYTE B
4968  016362  020327  000003                    CMP     R3,#3           ;TEST IF WORD 3 RETRIEVED
4969  016366  001402                            BEQ     4$              ;YES - EXIT
4970  016370  005203                            INC     R3              ;BUMP R3 TO NEXT WORD NUMBER
4971  016372  000732                            BR      3$              ;GO GET IT
4972  016374  012765  000177  000064  4$:       MOV     #177,PRTCON(R5) ;SET PRINT CONTROL
4973  016402  012737  015036  023352  7$:       MOV     #DRVERR,A.ABNL  ;RESTORE DRIVER RETURN
4974  016410  012603                            MOV     (SP)+,R3        ;;POP STACK INTO R3
4975  016412  012602                            MOV     (SP)+,R2        ;;POP STACK INTO R2
4976  016414  012600                            MOV     (SP)+,R0        ;;POP STACK INTO R0
4977  016416  000207                    6$:     RTS     PC
4978  016420  104400  003234          5$:       TYPE    ,STNTVD         ;TYPE WARNING ABOUT STATUS VALID
4979  016424  012765  000077  000064            MOV     #77,PRTCON(R5)  ;DO NOT PRINT STATUS WORDS
4980  016432  005046                            CLR     -(SP)           ;;PUT NEW PS ON STACK
4981  016434  012746  016442                    MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
4982  016440  000002                            RTI                     ;;POP NEW PC AND PS
4983  016442                            64$:
4984  016442  000757                            BR      7$
4985                                    .SBTTL  ERROR RETURN FROM DRIVER (CONTROLLER ERROR)
4986                                    ;*ENTRY: JSR    PC,CONERR WITH R5 POINTING TO ACTIVE PARAM BLK
4987                                    ;*RETURN: RTS   PC
4988                                    ;*
4989                                    ;*THIS ROUTINE IS CALLED BY THE DRIVER WHEN AN ERROR
4990                                    ;*OCCURS WITH THE CONTROLLER ERROR BIT SET.
```

```
 4991                                   ;;*****************************************************************
 4992  016444                           CERR:
 4993  016444  010046                           MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 4994  016446  010146                           MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 4995  016450  152737  000377  001667           BISB    #377,DONE       ;SET DONE
 4996  016456  032777  002000  162454           BIT     #SW10,@SWR      ;BELL ON ERROR?
 4997  016464  001402                           BEQ     1$              ;NO - BRANCH
 4998  016466  104400  001164                   TYPE    ,$BELL          ;RING BELL
 4999  016472  032777  020000  162440  1$:      BIT     #SW13,@SWR      ;INHIBIT PRINT?
 5000  016500  001024                           BNE     2$              ;YES - SKIP TO EXIT
 5001  016502  013700  023356                   MOV     E.CONT,R0       ;GET ERROR WORD
 5002  016506  100002                           BPL     3$
 5003  016510  104400  014025                   TYPE    ,$DETER
 5004  016514  005001                  3$:      CLR     R1
 5005  016516  006300                  8$:      ASL     R0              ;SHIFT ERROR WORD LEFT TO TEST BIT 15
 5006  016520  103405                           BCS     9$              ;IF CARRY TRUE GO REPORT ERROR
 5007  016522  020127  000040                   CMP     R1,#40          ;TEST IF ALL BITS TESTED
 5008  016526  001402                           BEQ     9$              ;GO REPORT NO ERROR ENTRY
 5009  016530  005721                           TST     (R1)+           ;BUMP R1 BY 2
 5010  016532  000771                           BR      8$              ;LOOP
 5011  016534  016137  023242  016544  9$:      MOV     ETABL(R1),10$   ;GET ADDRESS OF ERROR MESSAGE
 5012  016542  104400                           TYPE                    ;TYPE MESSAGE
 5013  016544  000000                  10$:     .WORD                   ;ADDRESS OF MESSAGE GOES HERE
 5014  016546  004737  016120                   JSR     PC,READRG       ;GO GET RK611 REGISTERS
 5015  016552  005037  023356                   CLR     E.CONT          ;CLEAR CONTROLLER ERROR WORD
 5016  016556  152737  000001  023240  2$:      BISB    #1,CEFLG        ;SET CONTROLLER ERROR FLAG
 5017  016564  004737  015036                   JSR     PC,DRVERR       ;GO REPORT
 5018  016570  105037  023240                   CLRB    CEFLG           ;CLEAR ERROR FLAG
 5019  016574  012601                           MOV     (SP)+,R1        ;;POP STACK INTO R1
 5020  016576  012600                           MOV     (SP)+,R0        ;;POP STACK INTO R0
 5021  016600  012762  000040  000010           MOV     #SCLR,RKCS2(R2) ;CLEAR CONTROLLER
 5022  016606  012762  000100  000000           MOV     #IE,RKCS1(R2)   ;SET IE AGAIN
 5023  016614  004737  016622                   JSR     PC,SWCONT       ;GO TO SWITCH CONTROL
 5024  016620  000207                           RTS     PC              ;RETURN
 5025                                   ;;*****************************************************************
 5026                                   .SBTTL  SWITCH CONTROL TEST
 5027                                   ;*ENTRY: JSR     PC,SWCONT
 5028                                   ;*RETURN: RTS    PC              IF NO LOOP OR HALT
 5029                                   ;*
 5030                                   ;*       MOV     $LPERR,(SP)     IF LOOP ON ERROR
 5031                                   ;*       RTS     PC
 5032                                   ;*
 5033                                   ;*       HALT                    IF HALT ON ERROR
 5034                                   ;*       PRESS CONTINUE TO EXECUTE NEXT COMMAND OR LOOP ON
 5035                                   ;*       ERROR (SWR 9).
 5036                                   ;*****************************************************************
 5037  016622  032777  100000  162310  SWCONT: BIT     #SW15,@SWR      ;TEST HALT ON ERROR
 5038  016630  001401                           BEQ     1$
 5039  016632  000000                           HALT
 5040  016634  032777  001000  162276  1$:      BIT     #SW9,@SWR       ;TEST LOOP ON ERROR
 5041  016642  001411                           BEQ     2$
 5042  016644  152737  000001  001103           BISB    #1,$ERFLG       ;SET ERFLG TO LOOP ON TEST FOREVER
 5043  016652  013716  001110                   MOV     $LPERR,(SP)     ;FUDGE RETURN FOR LOOP
 5044  016656  005046                           CLR     -(SP)           ;;PUT NEW PS ON STACK
 5045  016660  012746  016666                   MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
 5046  016664  000002                           RTI                     ;;POP NEW PC AND PS
```

```
 5047  016666                                  64$:
 5048  016666  000207                          2$:     RTS     PC                      ;RETURN
 5049                                          ;;**************************************************************
 5050                                          .SBTTL  PRINT LOOP COUNTER IF NOT ZERO ROUTINE
 5051                                          ;;**************************************************************
 5052  016670  005737  001722                  PRLPCT: TST     LPCNT1                  ;TEST IF LOOP COUNTER ZERO
 5053  016674  001003                                  BNE     1$                      ;IF ZERO - EXIT
 5054  016676  005737  001724                          TST     LPCNT2
 5055  016702  001412                                  BEQ     3$
 5056  016704  012746  001722                  1$:     MOV     #LPCNT1,-(SP)           ;GET ADDRESS OF LOOP COUNTERS
 5057  016710  004737  033466                          JSR     PC,@#$DB2D              ;CALL CONVERSION
 5058  016714  012637  016722                          MOV     (SP)+,2$                ;STORE RESULTS FOR PRINT
 5059  016720  104400                                  TYPE
 5060  016722  000000                          2$:     .WORD                           ;RESULTS GO HERE
 5061  016724  104400  003415                          TYPE    ,LPLABL
 5062  016730  000207                          3$:     RTS     PC
 5063                                          ;;**************************************************************
 5064                                          .SBTTL  ERROR REPORT PRINT ROUTINE
 5065                                          ;*ENTRY:  FPRINT     (TRAP CALL):WITH R2 A SWITCH SUCH THAT
 5066                                          ;*                               IF NON-ZERO THE OCTAL TYPE
 5067                                          ;*                               OUT IS A BYTE AND IF ZERO IT
 5068                                          ;*                               IS A WORD.
 5069                                          ;*                              :WITH R0 CONTAINING THE
 5070                                          ;*                               ADDRESS OF THE DATA TO
 5071                                          ;*                               BE TYPED.
 5072                                          ;*RETURN: RTI
 5073                                          ;;**************************************************************
 5074  016732  005702                          FPRINT: TST     R2                      ;TEST IF BYTE OPERATION
 5075  016734  001403                                  BEQ     1$                      ;BRANCH IF NO, ELSE
 5076  016736  005046                                  CLR     -(SP)                   ;CLEAR WORD ON STACK
 5077  016740  112016                                  MOVB    (R0)+,(SP)              ;MOVE BYTE FOR PRINT ON STACK
 5078  016742  000401                                  BR      2$
 5079  016744  012046                          1$:     MOV     (R0)+,-(SP)             ;MOVE WORD ON STACK
 5080  016746  104401                          2$:     TYPOC                           ;TYPE BYTE OR WORD
 5081  016750  104400  003540                          TYPE    ,SPACE2
 5082  016754  000002                                  RTI                             ;RETURN
 5083                                          ;;**************************************************************
 5084                                          .SBTTL  STATUS COMPARE EXECUTION
 5085                                          ;*ENTRY: JSR     R4,E$SC
 5086                                          ;*       <STATUS WORD NUMBER>
 5087                                          ;*       <EXPECTED VALUE>
 5088                                          ;*       <MASK>
 5089                                          ;*RETURN: RTS   R4
 5090                                          ;*
 5091                                          ;*THIS ROUTINE FIRST GOES TO THE DRIVER PARAMETER BLOCK TO CHECK
 5092                                          ;*IF THE STATUS HAS ALREADY BEEN RETRIEVED AND STORED THERE.  IF
 5093                                          ;*IT HAS, THE STATUS IN THE PARAMETER BLOCK IS USED IN THE
 5094                                          ;*TEST.  IF STATUS HAS NOT BEEN STORED, A DRIVE SELECT
 5095                                          ;*SPECIAL IS ISSUED TO THE DRIVE TO GET ALL THE STATUS
 5096                                          ;*WORDS.
 5097                                          ;*
 5098                                          ;*THE COMPARE ONLY TESTS THE BITS THAT CORRESPOND TO ONES
 5099                                          ;*IN THE MASK.  IF A MISCOMPARE OCCURS THE NUMBER OF THE
 5100                                          ;*STATUS WORD TESTED, THE EXPECTED VALUE, THE RECEIVED VALUE,
 5101                                          ;*AND THE MASK ARE REPORTED.
 5102                                          ;;**************************************************************
```

```
5103            017610                          .=17610
5104  017610    012705  002142        ESSC:     MOV     #PARM0,R5       ;SET R5 TO BLOCK 0
5105  017614    005237  001116                  INC     LINNUM          ;INCREMENT LINE COUNT
5106  017620    105737  001670                  TSTB    PBSW            ;TEST PARAM BLOCK SWITCH
5107  017624    001402                          BEQ     1$              ;IF BLOCK 0 SELECTED BRANCH
5108  017626    012705  002230                  MOV     #PARM1,R5       ;ELSE SET R5 TO BLOCK 1
5109  017632    032765  001000  000014  1$:     BIT     #PBSVAL,P.PRST(R5) ;TEST STATUS VALID BIT
5110  017640    001005                          BNE     2$              ;STATUS VALID, GO TO TEST
5111  017642    112765  000141  000001          MOVB    #RDSTAT,P.CMND(R5) ;SET TO DO READ STATUS
5112  017650    004737  013570                  JSR     PC,DRVCAL       ;GO TO DRIVER CALL
5113  017654    012400                  2$:     MOV     (R4)+,R0        ;STORE STATUS WORD NUMBER
5114  017656    012401                          MOV     (R4)+,R1        ;STORE EXPECTED VALUE
5115  017660    012402                          MOV     (R4)+,R2        ;STORE MASK
5116  017662    006300                          ASL     R0              ;SHIFT ST WD FOR CORRECT INDEX
5117  017664    062700  000040                  ADD     #P.AOO,R0       ;ADD BLOCK OFFSET FOR STATUS WORDS
5118  017670    060500                          ADD     R5,R0           ;COMPUTE STATUS WORD ADDRESS
5119  017672    011003                          MOV     (R0),R3         ;STORE IT
5120  017674    005102                          COM     R2              ;COMPLIMENT MASK
5121  017676    040201                          BIC     R2,R1           ;CLEAR UNTESTED BITS IN
5122  017700    040203                          BIC     R2,R3           ;EXPECTED & RECEIVED VALUES
5123  017702    020103                          CMP     R1,R3           ;COMPARE FOR EQUAL
5124  017704    001447                          BEQ     3$              ;EQUAL - EXIT
5125  017706    032777  002000  161234          BIT     #SW10,@SWR      ;BELL ON ERROR?
5126  017714    001402                          BEQ     4$              ;NO - BRANCH
5127  017716    104400  001164                  TYPE    ,$BELL
5128  017722    032777  020000  161710  4$:     BIT     #SW13,@SWR      ;INHIBIT PRINT?
5129  017730    001033                          BNE     5$              ;YES - BRANCH
5130  017732    162704  000006                  SUB     #6,R4           ;BACK UP R4 TO PARAMETERS
5131  017736    013746  001116                  MOV     LINNUM,-(SP)    ;PUT LINE NUMBER ON STACK
5132  017742    104434                          TYPDS                   ;TYPE IT
5133  017744    104400  014061                  TYPE    ,PLINE          ;LABEL IT
5134  017750    104400  014520                  TYPE    ,SCERR          ;TYPE STATUS COMPARE ERR
5135  017754    012446                          MOV     (R4)+,-(SP)     ;GET STATUS WORD
5136  017756    104401                          TYPOC                   ;TYPE IT
5137  017760    104400  014550                  TYPE    ,STWDNM         ;TYPE LABEL
5138  017764    012446                          MOV     (R4)+,-(SP)     ;GET GOOD WORD
5139  017766    104401                          TYPOC                   ;TYPE IT
5140  017770    104400  014467                  TYPE    ,GDDAT          ;TYPE LABEL
5141  017774    011046                          MOV     (R0),-(SP)      ;GET BAD WORD
5142  017776    104401                          TYPOC                   ;TYPE IT
5143  020000    104400  014504                  TYPE    ,BDDAT          ;TYPE LABEL
5144  020004    012446                          MOV     (R4)+,-(SP)     ;GET MASK
5145  020006    104401                          TYPOC                   ;TYPE IT
5146  020010    104400  014571                  TYPE    ,MSKLAB         ;TYPE LABEL
5147  020014    004737  016670                  JSR     PC,PRLPCT
5148  020020    004737  016622          5$:     JSR     PC,SWCONT       ;GO TO SWITCH CONTROL FOR LOOP
5149  020024    000204                  3$:     RTS     R4              ;RETURN
5150                                    ;;*****************************************************************
5151                                    .SBTTL  REGISTER COMPARE EXECUTION
5152                                    ;*ENTRY: JSR    R4      ESREGC
5153                                    ;*       <REGISTER NUMBER>
5154                                    ;*       <EXPECTED VALUE>
5155                                    ;*       <MASK>
5156                                    ;*RETURN: RTS   R4
5157                                    ;*
5158                                    ;*THIS ROUTINE DIRECTLY ACCESSES THE DESIRED RK611 REGISTER AND COMPARES
```

```
                                        ;*IT TO THE EXPECTED VALUE.  ONLY THE BITS THAT CORRESPOND TO ONES
                                        ;*IN THE MASK ARE COMPARED.
                                        ;*
                                        ;*THE REGISTER NUMBER APPEARING IN THE PARAMETER IS AN OCTAL
                                        ;*VALUE.  THIS VALUE IS USED AS AN INDEX TO SELECT THE DESIRED
                                        ;*REGISTER.  SINCE THE REGISTERS ARE ON WORD BOUNDARIES, THE VALUE
                                        ;*MUST BE SHIFTED LEFT ONE BIT BEFORE IT IS USED AS THE INDEX.
                                        ;*
                                        ;*IF A COMPARE PRODUCES AN ERROR, THE REGISTER NUMBER, THE EXPECTED VALUE, THE
                                        ;*VALUE READ, AND THE MASK ARE PRINTED ON THE CONSOLE.
                                        ;*
                                        ;;*****************************************************************
020026  005237  001116          E$REGC: INC     LINNUM          ;INCREMENT PSUEDO LINE CNT
020032  012400                          MOV     (R4)+,R0        ;GET REGISTER NUMBER
020034  012401                          MOV     (R4)+,R1        ;GET EXPECTED VALUE
020036  012402                          MOV     (R4)+,R2        ;GET MASK
020040  006300                          ASL     R0              ;MAKE REGNUM CORRECT FOR INDEX
020042  063700  023342                  ADD     RKBAS,R0        ;COMPUTE ADDRESS
020046  011003                          MOV     (R0),R3         ;GET CONTENTS
020050  010300                          MOV     R3,R0           ;STORE AGAIN FOR REPORT IF ERROR
020052  005102                          COM     R2              ;COMPLEMENT MASK
020054  040201                          BIC     R2,R1           ;CLEAR UNTESTED BITS IN EXP. VAL.
020056  040200                          BIC     R2,R0           ;CLEAR UNTESTED BITS FROM REG
020060  020001                          CMP     R0,R1           ;COMP TWO VALUES
020062  001447                          BEQ     1$              ;IF ZERO, NO ERROR. BRANCH
020064  032777  002000  161046          BIT     #SW10,@SWR      ;BELL ON ERROR?
020072  001402                          BEQ     3$              ;NO - BRANCH
020074  104400  001164                  TYPE    ,$BELL          ;RING BELL
020100  032777  020000  161032  3$:     BIT     #SW13,@SWR      ;INHIBIT PRINT?
020106  001033                          BNE     2$              ;YES - BRANCH
020110  162704  000006                  SUB     #6,R4           ;REPOSITION R4 FOR REPORT DATA
020114  013746  001116                  MOV     LINNUM,-(SP)    ;PUT LINE NUMBER ON STACK
020120  104404                          TYPDS                   ;TYPE IT
020122  104400  014061                  TYPE    ,PLINE          ;LABEL IT
020126  104400  014601                  TYPE    ,RCERR          ;TYPE REG COMP ERROR
020132  012446                          MOV     (R4)+,-(SP)     ;GET REGISTER NUMBER
020134  104401                          TYPOC                   ;TYPE IT
020136  104400  014633                  TYPE    ,REGLAB         ;LABEL IT
020142  012446                          MOV     (R4)+,-(SP)     ;GET GOOD WORD
020144  104401                          TYPOC                   ;TYPE IT
020146  104400  014467                  TYPE    ,GDDAT          ;LABEL IT
020152  010346                          MOV     R3,-(SP)        ;GET BAD WORD
020154  104401                          TYPOC                   ;TYPE IT
020156  104400  014504                  TYPE    ,BDDAT          ;LABEL IT
020162  012446                          MOV     (R4)+,-(SP)     ;GET MASK
020164  104401                          TYPOC                   ;TYPE IT
020166  104400  014571                  TYPE    ,MSKLAB         ;LABEL IT
020172  004737  016670                  JSR     PC,PRLPCT
020176  004737  016622          2$:     JSR     PC,SWCONT       ;GO TO SWITCH CONTROL FOR LOOP
020202  000204                  1$:     RTS     R4
                                        ;;*****************************************************************
                                        .SBTTL  DATA COMPARE EXECUTION
                                        ;*ENTRY: JSR    R4,E.DATC
                                        ;*      <NUMBER OF WORDS>
                                        ;*RETURN: RTS   R4
                                        ;*THIS ROUTINE WILL COMPARE THE CONTENTS OF THE INPUT BUFFER
```

```
                                        ;*TO THE CONTENTS OF THE OUTPUT BUFFER.  THE NUMBER OF
                                        ;*WORDS COMPARED IS SPECIFIED AS THE PARAMETER.
                                        ;*
                                        ;*IF A MISCOMPARE OCCURS, THE GOOD AND BAD WORD IS
                                        ;*REPORTED.  THE NUMBER OF THE WORD THAT MISCOMPARED IS
                                        ;*ALSO PRINTED.  ONLY THE FIRST 10 MISCOMPARES WILL BE
                                        ;*PRINTED UNLESS SWITCH REGISTER 1 IS SET.  IN THAT
                                        ;*CASE ALL MISCOMPARES ARE PRINTED.
                                        ;*****************************************************************
020204                                  ESDATC:
020204  010546                                  MOV     R5,-(SP)        ;;PUSH R5 ON STACK
020206  005237  001116                          INC     LINNUM          ;INCREMENT PSUEDO LINE COUNTER
020212  013700  001710                          MOV     IBUFPT,R0       ;START OF INPUT BUFFER
020216  013701  001706                          MOV     OBUFPT,R1       ;START OF OUTPUT BUFFER
020222  011402                                  MOV     (R4)+,R2        ;COMPARE COUNT
020224  005003                                  CLR     R3              ;ERROR COUNT
020226  005005                                  CLR     R5              ;CLEAR COUNT FOR REPORT
020230  022021                          1$:     CMP     (R0)+,(R1)+     ;COMPARE DATA
020232  001004                                  BNE     2$              ;ERROR
020234  005205                          4$:     INC     R5              ;COUNT FOR PRINT
020236  005302                                  DEC     R2              ;COUNT FOR COMPARE LENGTH
020240  001453                                  BEQ     11$             ;EXIT - OK
020242  000772                                  BR      1$              ;LOOP
020244  032777  002000  160666          2$:     BIT     #SW10,@SWR      ;BELL ON ERROR?
020252  001402                                  BEQ     5$              ;NO - BRANCH
020254  104400  001164                          TYPE    ,$BELL          ;RING BELL
020260  032777  000004  160652          5$:     BIT     #SW2,@SWR       ;INHIBIT PRINT?
020266  001053                                  BNE     12$             ;YES - SKIP PRINT
020270  005703                                  TST     R3              ;ANY ERRORS YET COUNTED?
020272  001005                                  BNE     6$              ;YES - DON'T PRINT LINE NUMBER
020274  013746  001116                          MOV     LINNUM,-(SP)    ;PUT LINE NUMBER ON STACK
020300  104404                                  TYPDS                   ;TYPE IT
020302  104400  014061                          TYPE    ,PLINE          ;LABEL IT
020306  020327  000012                  6$:     CMP     R3,#12          ;TEST FOR TO MANY ERRORS
020312  101404                                  BLOS    3$              ;NO - SKIP SWR TEST
020314  032777  000002  160616                  BIT     #SW1,@SWR       ;TEST SWITCH 1 SET
020322  001417                                  BEQ     13$             ;EXIT WITH ERROR
020324  104400  014434                  3$:     TYPE    ,FDATC          ;TYPE DATA COMPARE ERROR
020330  010546                                  MOV     R5,-(SP)        ;ERROR WORD POSITION
020332  104401                                  TYPOC                   ;PRINT IT
020334  104400  001171                          TYPE    ,$CRLF          ;RETURN CARRIAGE
020340  024041                                  CMP     -(R0),-(R1)     ;BACK UP DATA POINTERS
020342  012146                                  MOV     (R1)+,-(SP)     ;GOOD WORD
020344  104401                                  TYPOC                   ;TYPE IT
020346  104400  014467                          TYPE    ,GDDAT          ;TYPE LABEL
020352  012046                                  MOV     (R0)+,-(SP)     ;BAD WORD
020354  104401                                  TYPOC                   ;TYPE IT
020356  104400  014504                          TYPE    ,BDDAT          ;LABEL IT
020362  005203                          13$:    INC     R3              ;COUNT ERROR
020364  000723                                  BR      4$
020366  005703                          11$:    TST     R3              ;TEST IF ANY ERRORS OCCURED
020370  001414                                  BEQ     14$             ;NO - EXIT
020372  032777  020000  160540                  BIT     #SW13,@SWR      ;CHECK IF INHIBIT TYPE OUT
020400  001006                                  BNE     12$             ;YES, RETURN
020402  010346                                  MOV     R3,-(SP)        ;PUT ERROR COUNT ON STACK
020404  104401                                  TYPOC                   ;TYPE IT
```

```
 5271  020406  104400  003267                  TYPE    ,TOTMSC         ;LABEL IT
 5272  020412  004737  016670                  JSR     PC,PRLPCT
 5273  020416  004737  016622          12$:    JSR     PC,SWCONT       ;GO TO SWITCH CONTROL
 5274  020422                         14$:
 5275  020422  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
 5276  020424  000204                          RTS     R4              ;RETURN TO OBJECT CODE
 5277                                 ;;*****************************************************************
 5278                                 .SBTTL  REGISTER WRITE EXECUTION
 5279                                 ;*ENTRY: JSR    R4,E$RW
 5280                                 ;*      <REG NUMBER>
 5281                                 ;*      <VALUE>
 5282                                 ;*RETURN: RTS   R4
 5283                                 ;*
 5284                                 ;*THIS ROUTINE PLACES THE VALUE GIVEN IN THE REGISTER
 5285                                 ;*SPECIFIED.  NO ATTEMPT IS MADE TO PROTECT THE USER
 5286                                 ;*AGAINST WRITING READ-ONLY BITS OR AGAINST CAUSING
 5287                                 ;*UNEXPECTED INTERRUPTS.
 5288                                 ;;*****************************************************************
 5289  020426  005237  001116         E$RW:   INC     LINNUM          ;INCREMENT PSUEDO LN CNT
 5290  020432  012400                          MOV     (R4)+,R0        ;GET REG NUM
 5291  020434  006300                          ASL     R0              ;ALIGN R0 (REGISTER NUM) FOR INDEX
 5292  020436  063700  023342                  ADD     RKBAS,R0        ;COMPUTE ADDRESS
 5293  020442  012410                          MOV     (R4)+,(R0)      ;LOAD VALUE
 5294  020444  000204                          RTS     R4              ;RETURN
 5295                                 ;;*****************************************************************
 5296                                 .SBTTL  STALL EXECUTION
 5297                                 ;*ENTRY: JSR    R4,E$ST
 5298                                 ;*      <NUM OF MILLISECONDS>
 5299                                 ;*RETURN: RTS   R4
 5300                                 ;*
 5301                                 ;*THIS ROUTINE WILL DELAY THE PROGRAM EXECUTION FOR
 5302                                 ;*THE TIME SPECIFIED.  THE DRIVER WATCHDOG TIMER IS
 5303                                 ;*USED FOR THE DELAY TIMING.
 5304                                 ;;*****************************************************************
 5305  020446  005237  001116         E$ST:   INC     LINNUM          ;INCREMENT LINE COUNT
 5306  020452  012737  000340  177776          MOV     #PR7,PS         ;LOCK OUT ANY INTERRUPTS
 5307  020460  012437  023420                  MOV     (R4)+,W.DRV     ;LOAD MS DELAY INTO TIMER
 5308  020464  112737  000177  023404          MOVB    #177,W.TIME     ;FAKE TIMER TO THINK IT IS
 5309                                                                  ;WATCHING A DRIVE
 5310  020472  010637  001704                  MOV     SP,TEMP2        ;STORE STACK POINTER
 5311  020476  012737  020512  023352          MOV     #2$,A.ABNL      ;SET UP TIMER RETURN
 5312  020504  004737  023422          1$:     JSR     PC,W.WTCH       ;CALL TIMER
 5313  020510  000775                          BR      1$              ;LOOP ON TIMER
 5314  020512  013706  001704          2$:     MOV     TEMP2,SP        ;RESTORE STACK POINTER
 5315  020516  012737  015036  023352          MOV     #DRVERR,A.ABNL  ;RESTORE ABNORMAL RETURN
 5316  020524  012737  000000  177776          MOV     #PR0,PS         ;ALLOW ALL INTERRUPTS
 5317  020532  000204                          RTS     R4              ;RETURN
 5318                                 ;;*****************************************************************
 5319                                 .SBTTL  BUFFER INITIALIZE EXECUTION
 5320                                 ;*ENTRY: JSR    R4,E$BI
 5321                                 ;*      <EXECUTION LINE COUNTER CONTROL> <PATTERN>
 5322                                 ;*RETURN: RTS   R4
 5323                                 ;*
 5324                                 ;*THIS ROUTINE FIRST CHECKS LINE COUNT CONTROL AND IF A ONE
 5325                                 ;*INCREMENTS LINNUM.
 5326                                 ;*
```

```
                                        ;*THE SELECTED PATTERN IS THEN GENERATED AND THE ENTIRE OUTPUT
                                        ;*BUFFER IS INITIALIZED.
                                        ;;*****************************************************************
                020536                      .=20536
020536  005714                E$BI:   TST     (R4)            ;TEST IF LINE COUNT TO BE INC.
020540  100002                        BPL     1$              ;BRANCH IF NO
020542  005237  001116                INC     LINNUM          ;INC LINE COUNT
020546  013737  001706  001704  1$:   MOV     OBUFPT,TEMP2    ;COMPUTE AND STORE
020554  063737  001700  001704        ADD     MAXWDS,TEMP2    ;THE LAST USABLE BUFFER LOC
020562  063737  001700  001704        ADD     MAXWDS,TEMP2    ;DO AGAIN FOR LAST BUFF ADDRESS
020570  013702  001706                MOV     OBUFPT,R2       ;SET BUFFER POINTER
020574  012401                        MOV     (R4)+,R1        ;GET PARAMETER
020576  120127  000122                CMPB    R1,#'R          ;RANDOM PATTERN?
020602  001004                        BNE     2$              ;NO
020604  012701  001562                MOV     #PATR,R1        ;GET ADDRESS OF STORED PAT
020610  000137  021240                JMP     26$             ;GO INITIALIZE WITH PATTERN
020614  120127  000130        2$:     CMPB    R1,#'X          ;PAT X?
020620  001004                        BNE     3$              ;NO
020622  012701  001262                MOV     #PATX,R1        ;GET ADDRESS OF STORED PAT
020626  000137  021240                JMP     26$             ;GO INITIALIZE WITH PATTERN
020632  120127  000131        3$:     CMPB    R1,#'Y          ;PAT Y?
020636  001003                        BNE     4$              ;NO
020640  012701  001362                MOV     #PATY,R1        ;GET ADDRESS
020644  000575                        BR      26$             ;GO INITIALIZE WITH PATTERN
020646  120127  000132        4$:     CMPB    R1,#'Z          ;PAT Z?
020652  001003                        BNE     5$              ;NO
020654  012701  001462                MOV     #PATZ,R1        ;GET ADDRESS
020660  000567                        BR      26$             ;GO INITIALIZE WITH PATTERN
020662  120127  000101        5$:     CMPB    R1,#'A          ;TEST IF PAT A
020666  001013                        BNE     8$
020670  012701  177777                MOV     #177777,R1      ;1ST AND LAST WORD OF PATTERN
020674  005003                        CLR     R3              ;OTHER 30 WORDS
020676  012700  000036        6$:     MOV     #36,R0          ;SET COUNT FOR OTHER 30 WORDS
020702  010122                        MOV     R1,(R2)+        ;PUT 1ST WORD IN BUFFER
020704  010322                7$:     MOV     R3,(R2)+        ;PUT IN OTHER 30 LOOP
020706  005300                        DEC     R0
020710  001375                        BNE     7$
020712  010122                        MOV     R1,(R2)+        ;PUT LAST WORD OF PAT IN BUFF
020714  000761                        BR      28$             ;GO SPREAD THRU BUFFER
020716  120127  000102        8$:     CMPB    R1,#'B          ;PAT B?
020722  001004                        BNE     9$              ;NO
020724  005001                        CLR     R1              ;1ST AND LAST WORD OF PATTERN
020726  012703  177777                MOV     #177777,R3      ;OTHER 30 WORDS
020732  000761                        BR      6$              ;GO INITIALIZE WITH PATTERN
020734  120127  000103        9$:     CMPB    R1,#'C          ;PAT C?
020740  001004                        BNE     10$
020742  012701  125252                MOV     #125252,R1      ;1ST AND LAST WORD OF PATTERN
020746  010103                        MOV     R1,R3           ;OTHER 30 THE SAME
020750  000752                        BR      6$              ;GO INITIALIZE WITH PATTERN
020752  120127  000104        10$:    CMPB    R1,#'D          ;PAT D?
020756  001004                        BNE     11$
020760  012701  052525                MOV     #052525,R1      ;1ST AND LAST WORD
020764  010103                        MOV     R1,R3           ;OTHER 30 THE SAME
020766  000743                        BR      6$              ;GO INITIALIZE WITH PATTERN
020770  120127  000105        11$:    CMPB    R1,#'E          ;PAT E?
020774  001023                        BNE     15$
```

```
5383  020776  012701  000001            MOV   #1,R1          ;BASE WORD FOR PATTERN GEN
5384  021002  012700  000020            MOV   #20,R0         ;SET A LOOP CNTR
5385  021006  010122            12$:    MOV   R1,(R2)+       ;BASE WORD INTO BUFFER
5386  021010  005300                    DEC   R0             ;DEC COUNTER
5387  021012  001403                    BEQ   13$            ;1ST HALF GENERATED, EXIT
5388  021014  006301                    ASL   R1             ;TWO OPERATIONS TO
5389  021016  005201                    INC   R1             ;CONTINUE THE PATTERN
5390  021020  000772                    BR    12$            ;LOOP
5391  021022  042701  100000    13$:    BIC   #100000 ,R1    ;SET BASE FOR SECOND HALF
5392  021026  012700  000020            MOV   #20,R0         ;SET A COUNT
5393  021032  010122            14$:    MOV   R1,(R2)+       ;PUT WORD INTO BUFFER
5394  021034  005300                    DEC   R0             ;DEC COUNTER
5395  021036  001410                    BEQ   28$            ;PATTERN GENERATED, GO SPREAD IT
5396  021040  006201                    ASR   R1             ;SHIFT FOR NEXT WORD OF PAT
5397  021042  000773                    BR    14$            ;LOOP
5398  021044  120127  000106    15$:    CMPB  R1,#'F         ;PAT F?
5399  021050  001016                    BNE   18$
5400  021052  012701  177777            MOV   #177777,R1     ;BASE WORD FOR PATTERN
5401  021056  012700  000021            MOV   #21,R0         ;SET COUNTER
5402  021062  010122            16$:    MOV   R1,(R2)+       ;PUT IN BUFFER
5403  021064  005300                    DEC   R0             ;DEC COUNT
5404  021066  001402                    BEQ   17$            ;1ST HALF GENERATED, EXIT
5405  021070  006301                    ASL   R1             ;SHIFT FOR NEXT WD OF PAT
5406  021072  000773                    BR    16$
5407  021074  052701  100000    17$:    BIS   #100000,R1     ;SET BASE FOR 2ND HALF
5408  021100  012700  000017            MOV   #17,R0         ;SET COUNT
5409  021104  000752                    BR    14$            ;GO MAKE USE OF PREVIOUS
5410                                                         ;PATTERN GENERATION OP.
5411  021106  120127  000107    18$:    CMPB  R1,#'G         ;PAT G?
5412  021112  001021                    BNE   22$
5413  021114  012701  177777            MOV   #177777,R1     ;BASE WORD FOR PATTERN G
5414  021120  006101                    ROL   R1             ;THIS SETS CARRY AND RESETS BIT 0
5415  021122  012700  000020            MOV   #20,R0         ;SET COUNT
5416  021126  010122            19$:    MOV   R1,(R2)+       ;PUT IN BUFFER
5417  021130  005300                    DEC   R0             ;DEC COUNT
5418  021132  001402                    BEQ   20$            ;EXIT 1ST HALF IF ZERO
5419  021134  006101                    ROL   R1             ;SHIFT PATTERN
5420  021136  000773                    BR    19$            ;LOOP
5421  021140  012700  000020    20$:    MOV   #20,R0         ;SET 2ND HALF COUNT
5422  021144  006001            21$:    ROR   R1             ;SHIFT THE PATTERN
5423  021146  010122                    MOV   R1,(R2)+       ;PUT IN BUFFER
5424  021150  005300                    DEC   R0             ;DEC COUNT
5425  021152  001442                    BEQ   28$            ;
5426  021154  000773                    BR    21$            ;LOOP
5427  021156  120127  000110    22$:    CMPB  R1,#'H         ;PAT H?
5428  021162  001017                    BNE   24$
5429  021164  012701  000001            MOV   #1,R1          ;BASE WORD FOR PATTERN
5430  021170  012700  000020            MOV   #20,R0         ;SET COUNT
5431  021174  010122            31$:    MOV   R1,(R2)+       ;PUT IN BUFFER
5432  021176  005300                    DEC   R0             ;DEC COUNT
5433  021200  001402                    BEQ   23$            ;EXIT 1ST HALF IF ZERO
5434  021202  006301                    ASL   R1             ;SHIFT PATTERN
5435  021204  000773                    BR    31$            ;LOOP
5436  021206  010122            23$:    MOV   R1,(R2)+       ;PUT LAST WD GENERATED IN AGAIN
5437  021210  012701  040000            MOV   #40000,R1      ;SET NEW BASE FOR 2ND HALF
5438  021214  012700  000017            MOV   #17,R0         ;SET COUNT
```

```
5439  021220  000734                          BR      14$             ;GO USE PREV GENERATE OPERATION
5440  021222  120127  000111          24$:    CMPB    R1,#'I          ;PAT I
5441  021226  001000                          BNE     25$             ;NO BUT FORCE USE OF PAT I
5442  021230  012701  155555          25$:    MOV     #155555,R1      ;WORST CASE PAT
5443  021234  010103                          MOV     R1,R3           ;ALL WORDS THE SAME
5444  021236  000617                          BR      6$              ;GO INITIALIZE WITH PATTERN
5445  021240  012700  000040          26$:    MOV     #40,R0          ;SET COUNT
5446  021244  012122                  27$:    MOV     (R1)+,(R2)+     ;PUT IN BUFFER
5447  021246  020237  001704                  CMP     R2,TEMP2        ;CHECK BUFFER FULL
5448  021252  001405                          BEQ     30$             ;YES, EXIT
5449  021254  005300                          DEC     R0              ;DEC COUNT
5450  021256  001372                          BNE     27$             ;NO - LOOP
5451  021260  013701  001706          28$:    MOV     OBUFPT,R1       ;RESET R1 TO START OF
5452                                                                  ;OUTPUT BUFFER.  FROM THIS
5453                                                                  ;POINT ON THE INITIALIZE
5454                                                                  ;PATTERN IS SPREAD THROUGH THE
5455                                                                  ;OBUFF.
5456  021264  000765                          BR      26$             ;LOOP
5457  021266  000204                  30$:    RTS     R4              ;RETURN
5458                                  ;;*****************************************************************
5459                                  .SBTTL  PARAMETER BLOCK SELECTION
5460                                  ;*ENTRY: JSR     PC,PBSEL
5461                                  ;*RETURN: RTS    PC
5462                                  ;*
5463                                  ;*THIS ROUTINE SELECTS THE PARAMETER BLOCK TO BE USED IN THE
5464                                  ;*NEXT OPERATION AND SETS THE PARAMETER BLOCK SWITCH ACCORDINGLY.
5465                                  ;*R5 IS LOADED WITH THE PARAM BLOCK ADDRESS.
5466                                  ;*
5467                                  ;*IN ADDITION, THIS ROUTINE TAKES CARE OF INSERTING THE DRIVE
5468                                  ;*NUMBER AND NO CHECK INDICATOR IN THAT PARAM BLOCK.
5469                                  ;*IT ALSO INCREMENTS THE LINNUM (PSUEDO LINE COUNT).
5470                                  ;;*****************************************************************
5471  021270  005237  001116          PBSEL:  INC     LINNUM          ;INCREMENT LINE NUMBER COUNT
5472  021274  012705  002142                  MOV     #PARM0,R5       ;SET R5 WITH PARAM0
5473  021300  105737  001670                  TSTB    PBSW            ;IS PARAM0 TO BE USED NEXT
5474  021304  001002                          BNE     1$              ;YES (PARAM1 WAS USED LAST)
5475  021306  012705  002230                  MOV     #PARM1,R5       ;NO - SET TO PARAM1
5476  021312  105137  001670          1$:     COMB    PBSW            ;SET PBSW
5477  021316  111465  000000                  MOVB    (R4),P.DRVN(R5) ;MOV IN DRIVE PARAM
5478  021322  105714                          TSTB    (R4)            ;WAS IT A DRIVE NUM
5479  021324  100403                          BMI     2$              ;NO? GO CHANGE IT
5480  021326  112437  001176                  MOVB    (R4)+,DRIVE     ;YES? PUT IT IN PARAM STORAGE TOO.
5481  021332  000404                          BR      3$
5482  021334  113765  001176  000000  2$:     MOVB    DRIVE,P.DRVN(R5) ;GET STORED DRIVE NUMBER
5483  021342  105724                          TSTB    (R4)+           ;BUMP R4
5484  021344  005065  000014          3$:     CLR     P.PRST(R5)      ;CLEAR PROG STAT WORD
5485  021350  132714  000004                  BITB    #BAII,(R4)              ;TEST BAI INHIBIT SWITCH
5486  021354  001403                          BEQ     4$              ;NOT SET, SKIP
5487  021356  052765  100000  000014          BIS     #DTBAII,P.PRST(R5)  ;SET FOR INHIBIT INCREMENT
5488  021364  132714  000002          4$:     BITB    #NOCK,(R4)      ;TEST IF NO CHECK
5489  021370  001403                          BEQ     5$              ;NO - SKIP
5490  021372  052765  000400  000014          BIS     #NOCHK,P.PRST(R5)  ;SET NO CHECK
5491  021400  142765  000020  000007  5$:     BICB    #B.CFMT,P.CS1H(R5)  ;CLEAR 24 SECTOR MODE
5492  021406  132724  000010                  BITB    #SECT20,(R4)+   ;WAS THAT RIGHT?
5493  021412  001403                          BEQ     6$              ;YES - SKIP TO EXIT
5494  021414  152765  000020  000007          BISB    #B.CFMT,P.CS1H(R5)  ;NO - SET THE 24 SECTOR MODE BIT
```

```
                                        ;S$:     RTS     PC              ;RETURN
                                        ;;*****************************************************************
                                        ;.SBTTL  UNIBUS INITIALIZE EXECUTE
                                        ;*ENTRY:         JSR     R4,E$UI
                                        ;*RETURN:        RTS     R4
                                        ;*
                                        ;*THIS ROUTINE EXECUTES A UNIBUS INITIALIZE
                                        ;*
                                        ;;*****************************************************************
021422  000005                          E$UI:   RESET                           ;UNIBUS RESET
021424  005237  001116                          INC     LINNUM                  ;INCREMENT LINE NUMBER
021430  000204                          ;       RTS     R4                      ;RETURN
                                        ;;*****************************************************************
                                        ;.SBTT_  SUBSYSTEM COMMANDS EXECUTION
                                        ;*ENTRY: JSR    R4,E$XX WHERE XX IS THE SUBSYSTEM CMND
                                        ;*       <NO CHECK> <DRIVE>
                                        ;*       <CYL NUM> OR <OFFSET>
                                        ;*       <TRACK> <SECTOR>
                                        ;*       <WORD COUNT>
                                        ;*RETURN: RTS   R4
                                        ;*
                                        ;*THIS ROUTINE HAS MANY ENTRY POINTS, ONE FOR EACH SUBSYSTEM
                                        ;*COMMAND.  EACH ENTRY IS SPECIAL IN THAT THE PROPER FUNCTION
                                        ;*CODE IS PUT INTO THE PARAMETER BLOCK ALONG WITH THE
                                        ;*APPLICABLE PARAMETERS.
                                        ;*
                                        ;*THE PRINT CONTROL WORD IS PUT INTO THE LAST WORD (WORD 66)
                                        ;*OF THE PARAMETER BLOCK. THIS WORD CONTROLS WHICH INFORMATION
                                        ;*IN THE PARAMETER BLOCK IS PRINTED IF AN ERROR OCCURS.
                                        ;*REFER TO DRVERR FOR A DESCRIPTION OF THE WORD AND
                                        ;*THE RESULTING DATA REPORTED.
                                        ;;*****************************************************************
021434  112703  000113                  E$RC:   MOVB    #RECAL,R3       ;RECALIBRATE ENTRY
021440  012700  000104                          MOV     #104,R0         ;REPORT FORMAT ENTRY
021444  000137  022072                          JMP     ONEP
021450  112703  000103                  E$PA:   MOVB    #PACK,R3        ;PACK ACK ENTRY
021454  012700  000104                          MOV     #104,R0         ;REPORT FORMAT
021460  000137  022072                          JMP     ONEP
021464  112703  000177                  E$CS:   MOVB    #SUBCLR,R3      ;SUBSYSTEM CLEAR ENTRY
021470  012700  000104                          MOV     #104,R0         ;REPORT FORMAT
021474  000137  022072                          JMP     ONEP
021500  112703  000176                  E$CC:   MOVB    #CONCLR,R3      ;CONTROLLER CLEAR ENTRY
021504  012700  000104                          MOV     #104,R0         ;REPORT FORMAT
021510  000570                                  BR      ONEP
021512  112703  000101                  E$DS:   MOVB    #SELDRV,R3      ;DRIVE SELECT ENTRY
021516  012700  000104                          MOV     #104,R0         ;REPORT FORMAT
021522  000563                                  BR      ONEP
021524  112703  000105                  E$DC:   MOVB    #CLEAR,R3       ;DRIVE CLEAR ENTRY
021530  012700  000104                          MOV     #104,R0         ;REPORT FORMAT
021534  000556                                  BR      ONEP
021536  112703  000107                  E$UL:   MOVB    #UNLOAD,R3      ;UNLOAD ENTRY
021542  012700  000104                          MOV     #104,R0         ;REPORT FORMAT
021546  000551                                  BR      ONEP
021550  112703  000111                  E$SS:   MOVB    #SRTSPL,R3      ;START SPINDLE ENTRY
021554  012700  000104                          MOV     #104,R0         ;REPORT FORMAT
021560  000544                                  BR      ONEP
```

```
 5551  021562  112703  000115          ESOF:   MOVB    #OFFSET,R3      ;OFFSET ENTRY
 5552  021566  012700  000165                  MOV     #165,R0         ;REPORT FORMAT
 5553  021572  000527                          BR      TWOP
 5554  021574  112703  000164          ESAH:   MOVB    #RDALHD,R3      ;READ ALL HEADERS
 5555  021600  012700  000165                  MOV     #165,R0         ;SET REPORT FORMAT
 5556  021604  000501                          BR      THREEP
 5557  021606  112703  000125          ESRH:   MOVB    #RDHEAD,R3      ;READ HEADER ENTRY
 5558  021612  012700  000165                  MOV     #165,R0         ;REPORT FORMAT
 5559  021616  000474                          BR      THREEP
 5560  021620  112703  000117          ESSK:   MOVB    #SEEK,R3        ;SEEK ENTRY
 5561  021624  012700  000165                  MOV     #165,R0         ;REPORT FORMAT
 5562  021630  000467                          BR      THREEP
 5563  021632  004737  021270          ESWH:   JSR     PC,PBSEL
 5564  021636  112765  000127  000001          MOVB    #WRHEAD,P.CMND(R5)
 5565  021644  004737  022242                  JSR     PC,BLDHDR
 5566  021650  000437                          BR      SETADD
 5567  021652  010046          ESRD:   MOV     R0,-(SP)        ;STORE R0
 5568  021654  010146                  MOV     R1,-(SP)        ;AND R1
 5569  021656  013700  001710                  MOV     IBUFPT,R0       ;SET R0 AT START OF INPUT BUFF
 5570  021662  013701  001700                  MOV     MAXWDS,R1       ;SET R1 WITH NUMBER OF WORDS
 5571  021666  005020          1$:     CLR     (R0)+           ;CLEAR BUFFER LOCATION
 5572  021670  005301                  DEC     R1              ;DECREMENT COUNT
 5573  021672  001375                  BNE     1$              ;LOOP UNTIL DONE
 5574  021674  012601                  MOV     (SP)+,R1        ;RESTORE R1
 5575  021676  012600                  MOV     (SP)+,R0        ;AND R0
 5576  021700  004737  021270                  JSR     PC,PBSEL        ;READ DATA, GET PB ASSIGNMENT
 5577  021704  112765  000121  000001          MOVB    #RDDATA,P.CMND(R5) ;LOAD FUNCTION CODE
 5578  021712  013765  001710  000010          MOV     IBUFPT,P.BALO(R5) ;LOAD BUFFER ADDRESS
 5579  021720  000416                          BR      FOURP
 5580  021722  004737  021270          ESWD:   JSR     PC,PBSEL        ;WRITE DATA, GET PB ASSIGNMENT
 5581  021726  112765  000123  000001          MOVB    #WRDATA,P.CMND(R5) ;LOAD FUNCTION CODE
 5582  021734  000405                          BR      SETADD
 5583  021736  004737  021270          ESWC:   JSR     PC,PBSEL        ;WRITE CHECK,GET PB ASSIGNMENT
 5584  021742  112765  000131  000001          MOVB    #WRTCHK,P.CMND(R5) ;LOAD FUNCTION CODE
 5585  021750  013765  001706  000010  SETADD: MOV     OBUFPT,P.BALO(R5) ;LOAD BUFFER ADDRESS FOR WRITES
 5586  021756  012700  000177          FOURP:  MOV     #177,R0         ;REPORT FORMAT
 5587  021762  012465  000002                  MOV     (R4)-,P.CYLN(R5) ;LOAD CYL ADDRESS
 5588  021766  112465  000005                  MOVB    (R4)+,P.TRCK(R5) ;LOAD TRACK ADDRESS
 5589  021772  112465  000004                  MOVB    (R4)+,P.SECT(R5) ;LOAD SECTOR ADDRESS
 5590  021776  012465  000012                  MOV     (R4)+,P.WC(R5) ;LOAD WORD COUNT
 5591  022002  005465  000012                  NEG     P.WC(R5)        ;MAKE WORD COUNT 2'S COMP
 5592  022006  000435                          BR      GODRV
 5593  022010  004737  021270          THREEP: JSR     PC,PBSEL        ;GET PB ASSIGNMENT
 5594  022014  022703  000164                  CMP     #RDALHD,R3      ;TEST IF READ ALL HEADERS COMMAND
 5595  022020  001003                          BNE     1$              ;NO - SKIP
 5596  022022  012765  001736  000010          MOV     #HDBUFF,P.BALO(R5)  ;LOAD ADDRESS OF HEADER BUFFER
 5597  022030  110365  000001          1$:     MOVB    R3,P.CMND(R5) ;LOAD FUNCTION CODE
 5598  022034  012465  000002                  MOV     (R4)+,P.CYLN(R5) ;LOAD CYL ADDRESS
 5599  022040  112465  000005                  MOVB    (R4)+,P.TRCK(R5) ;LOAD TRACK ADD
 5600  022044  112465  000004                  MOVB    (R4)+,P.SECT(R5) ;LOAD SECTOR ADDRESS
 5601  022050  000414                          BR      GODRV
 5602  022052  004737  021270          TWOP:   JSR     PC,P3SEL        ;GET PB ASSIGNMENT
 5603  022056  110365  000001                  MOVB    R3,P.CMND(R5) ;LOAD FUNCTION CODE
 5604  022062  112465  000006                  MOVB    (R4)+,P.OFST(R5) ;LOAD OFFSET
 5605  022066  105724                          TSTB    (R4)+           ;BUMP R4 TO NXT COMD
 5606  022070  000404                          BR      GODRV
```

```
5607  022072  004737  021270          ONEP:   JSR     PC,PBSEL        ;GET PB ASSIGNMENT
5608  022076  110365  000001                  MOVB    R3,P.CMND(R5) ;LOAD FUNCTION CODE
5609  022102  010065  000064          GODRV:  MOV     R0,PRTCON(R5)   ;INSERT FORMAT WORD
5610  022106  004737  013570                  JSR     PC,DRVCAL       ;CALL DRIVER
5611  022112  000204                          RTS     R4              ;RETURN
5612                                  ;;*****************************************************************
5613                                  .SBTTL  COPY TAPE ROUTINE
5614                                  ;*THIS ROUTINE WILL PUNCH A TAPE THAT IS IDENTICAL TO THE
5615                                  ;*TAPE BEING READ, EITHER SOURCE OR OBJECT.
5616
5617  022114  104411                  CTRTE:  SAVREG
5618  022116  042777  000100  157602          BIC     #000100,@PTRSR  ;RESET IE, PTR
5619  022124  042777  000100  157600          BIC     #000100,@PTPSR  ;RESET IE, PTP
5620  022132  005277  157570                  INC     @PTRSR  ;SET READER ENABLE
5621  022136  032777  100200  157562  1$:     BIT     #100200,@PTRSR  ;CHECK PTR DONE OR ERROR
5622  022144  001774                          BEQ     1$              ;NO - LOOP
5623  022146  100425                          BMI     20$             ;ERROR - EXIT
5624  022150  032777  100200  157554  2$:     BIT     #100200,@PTPSR  ;CHECK PTP DONE OR ERROR
5625  022156  001774                          BEQ     2$              ;NO - LOOP
5626  022160  100423                          BMI     30$             ;ERROR - EXIT
5627  022162  005277  157540          5$:     INC     @PTRSR          ;START READ
5628  022166  032777  100200  157532  3$:     BIT     #100200,@PTRSR  ;TEST DONE OR ERROR
5629  022174  001774                          BEQ     3$              ;NEITHER - LOOP
5630  022176  100411                          BMI     20$             ;ERROR - EXIT
5631  022200  117777  157524  157526          MOVB    @PTRDB,@PTPDB   ;MOVE DATA TO PUNCH
5632  022206  032777  100200  157516  4$:     BIT     #100200,@PTPSR  ;CHECK IF DONE OR ERROR
5633  022214  001774                          BEQ     4$              ;NEITHER - LOOP
5634  022216  100404                          BMI     30$             ;ERROR -EXIT
5635  022220  000760                          BR      5$              ;DO NEXT READ
5636  022222  104400  003217          20$:    TYPE    ,PRERR          ;READER ERROR
5637  022226  000402                          BR      40$
5638  022230  104400  003203          30$:    TYPE    ,PUERR          ;PUNCH ERROR
5639  022234  104412                  40$:    RESREG
5640  022236  005724                          TST     (R4)+           ;GOOD RETURN
5641  022240  000204                          RTS     R4              ;RETURN
5642                                  ;;*****************************************************************
5643                                  .SBTTL  ROUTINE TO BUILD HEADERS
5644                                  ;*THIS ROUTINE WILL BUILD HEADERS AS THEY ARE EXPECTED TO BE FOUND
5645                                  ;*ON THE PACK WHEN THE SECTOR FIELD OF A WRITE HEADER COMMAND IS
5646                                  ;*SPECIFIED AS ZERO. IF ANY VALUE OTHER THAN 0 IS ENTERED AS THE
5647                                  ;*SECTOR VALUE THE ROUTINE GOES TO THE SPECIAL DATA PATTERN BUFFERS
5648                                  ;*AND TRANSFERS THE DATA FOUND IN PAT X, PAT Y, AND THE FIRST TWO
5649                                  ;*WORDS OF PAT Z INTO THE OUPUT BUFFER TO BE WRITTEN AS THE HEADERS.
5650                                  ;;*****************************************************************
5651  022242  104411                  BLDHDR: SAVREG
5652  022244  012700  000026                  MOV     #26,R0          ;PRESET FOR 26 SECTORS
5653  022250  005037  001702                  CLR     TEMP1           ;CLEAR LOCATION TO BE USED AS FLAG
5654  022254  132765  000020  000007          BITB    #B.CFMT,P.CS1H(R5)  ;TEST IF THAT IS RIGHT
5655  022262  001405                          BEQ     5$              ;YES - SKIP
5656  022264  012700  000024                  MOV     #24,R0          ;NO - SET FOR 24 SECTORS
5657  022270  052737  001000  001702          BIS     #BIT9,TEMP1
5658  022276  013705  001706          5$:     MOV     OBUFPT,R5       ;GET BUFFER ADDRESS
5659  022302  012401                          MOV     (R4)+,R1        ;GET CYL NUMBER PARAMETER
5660  022304  112402                          MOVB    (R4)+,R2        ;GET TRACK PAREMETER
5661  022306  112403                          MOVB    (R4)+,R3        ;GET SECTER PARAMETER
5662  022310  001025                          BNE     2$              ;IF SECTOR NOT 0, GO GETHDRS FROM PAT X,Y,Z BUF
```

```
5663  022312  006302                        ASL     R2              ;ADJUST TRACK FOR CORRECT POSITION
5664  022314  006302                        ASL     R2
5665  022316  006302                        ASL     R2
5666  022320  006302                        ASL     R2
5667  022322  006302                        ASL     R2
5668  022324  052702  140000                BIS     #140000,R2      ;SET NO BAD SECTOR BITS
5669  022330  053702  001702                BIS     TEMP1,R2        ;INSERT FORMAT BIT
5670  022334  010103                1$:     MOV     R1,R3           ;COMPUTE VRC
5671  022336  010204                        MOV     R2,R4
5672  022340  040104                        BIC     R1,R4
5673  022342  040203                        BIC     R2,R3
5674  022344  050403                        BIS     R4,R3
5675  022346  010125                        MOV     R1,(R5)+        ;INSEPT WORD 1
5676  022350  010225                        MOV     R2,(R5)+        ;INSERT WORD 2
5677  022352  010325                        MOV     R3,(R5)+        ;INSERT WORD 3
5678  022354  005202                        INC     R2              ;BUMP SECTOR COUNT
5679  022356  005300                        DEC     R0              ;DECREMENT COUNTER
5680  022360  001365                        BNE     1$              ;LOOP IF NOT YET ZERO
5681  022362  000410                        BR      4$              ;EXIT
5682  022364  012701  001262          2$:   MOV     #PATX,R1        ;GET ADDRESS OF PAT X BUFFER
5683  022370  010003                        MOV     R0,R3           ;MULTIPLY SECTOR COUNT BY 3
5684  022372  006300                        ASL     R0              ;TO GEY THE NUMBER OF WORDS REQUIRED
5685  022374  060300                        ADD     R3,R0
5686  022376  012125                3$:     MOV     (R1)+,(R5)+     ;MOVE HEADER WORD
5687  022400  005300                        DEC     R0              ;DEC COUNT
5688  022402  001375                        BNE     3$              ;LOOP UNTIL DONE
5689  022404  104412                4$:     RESREG
5690  022406  000207                        RTS     PC
5691                                  ;;*****************************************************************
5692                                  .SBTTL  ERROR MESSAGES FOR CONTROLLER ERROR RETURN
5693  022410  046103  040505  020122  CERR0:  .ASCIZ  /CLEAR CONTROLLER DID NOT CLEAR ERROR/<15><12>
5694  022416  047503  052116  047522
5695  022424  046114  051105  042040
5696  022432  042111  047040  052117
5697  022440  041440  042514  051101
5698  022446  042440  051122  051117
5699  022454  005015     000
5700  022457     116  020117  052101  CERR1:  .ASCIZ  /NO ATTENTION IS ATTENTION SUMMARY REGISTER/<15><12>
5701  022464  042524  052116  047511
5702  022472  020116  051511  040440
5703  022500  052124  047105  044524
5704  022506  047117  051440  046525
5705  022514  040515  054522  051040
5706  022522  043505  051511  042524
5707  022530  006522  000012
5708  022534  047125  047523  044514  CERR2:  .ASCIZ  /UNSOLICITED ATTENTION/<15><12>
5709  022542  044503  042524  020104
5710  022550  052101  042524  052116
5711  022556  047511  006516  000012
5712  022564  047125  054105  042520  CERR3:  .ASCIZ  /UNEXPECTED DATA TYPE ERROR/<15><12>
5713  022572  052103  042105  042040
5714  022600  052101  020101  054524
5715  022606  042520  042440  051122
5716  022614  051117  005015     000
5717  022621     101  052124  047105  CERR4:  .ASCIZ  /ATTENTION DID NOT RESET WITH DRIVE CLEAR/<15><12>
5718  022626  044524  047117  042040
```

```
5719  022634  042111  047040  052117
5720  022642  051040  051505  052105
5721  022650  053440  052111  020110
5722  022656  051104  053111  020105
5723  022664  046103  040505  006522
5724  022672  000012
5725  022674  052101  042524  052116  CERR5:  .ASCIZ  /ATTENTION DID NOT CLEAR WITH SUS-SYSTEM CLEAR/<15><12>
5726  022702  047511  020116  044504
5727  022710  020104  047516  020124
5728  022716  046103  040505  020122
5729  022724  044527  044124  051440
5730  022732  051525  051455  051531
5731  022740  042524  020115  046103
5732  022746  040505  006522  000012
5733  022754  046111  042514  040507  CERR6:  .ASCIZ  /ILLEGAL DRIVER COMMAND/<15><12>
5734  022762  020114  051104  053111
5735  022770  051105  041440  046517
5736  022776  040515  042116  005015
5737  023004     000
5738  023005     104  052101  020101  CERR8:  .ASCIZ  /DATA LATE WHEN UNLOADING HEADER/<15><12>
5739  023012  040514  042524  053440
5740  023020  042510  020116  047125
5741  023026  047514  042101  047111
5742  023034  020107  042510  042101
5743  023042  051105  005015     000
5744  023047     103  047117  051124  CERR9:  .ASCIZ  /CONTROLLER ERROR DURING DRIVE SERVICING/<15><12>
5745  023054  046117  042514  020122
5746  023062  051105  047522  020122
5747  023070  052504  044522  043516
5748  023076  042040  044522  042526
5749  023104  051440  051105  044526
5750  023112  044503  043516  005015
5751  023120     000
5752  023121     104  044522  042526  CERR10: .ASCIZ  /DRIVE PARITY WHILE GATHERING STATUS/<15><12>
5753  023126  050040  051101  052111
5754  023134  020131  044127  046111
5755  023142  020105  040507  044124
5756  023150  051105  047111  020107
5757  023156  052123  052101  051525
5758  023164  005015     000
5759  023167     115  046125  044524  CERR15: .ASCIZ  /MULTIPLE DRIVE SELECT/<15><12>
5760  023174  046120  020105  051104
5761  023202  053111  020105  042523
5762  023210  042514  052103  005015
5763  023216     000
5764  023217     116  020117  051105  CERR7:  .ASCIZ  /NO ERROR ENTRY/<15><12>
5765  023224  047522  020122  047105
5766  023232  051124  006531  000012
5767                                  .EQUIV  CERR7,CERR11
5768                                  .EQUIV  CERR7,CERR12
5769                                  .EQUIV  CERR7,CERR13
5770                                  .EQUIV  CERR7,CERR14
5771                                  .EQUIV  CERR7,CERR16
5772  023240     000             CEFLG:' .BYTE   0               ;CONTROLLER ERROR FLAG
5773          023242                     .EVEN
5774  023242  023167             ETABL:  .WORD   CERR15
```

```
5775  023244  023217                          .WORD   CERR14
5776  023246  023217                          .WORD   CERR13
5777  023250  023217                          .WORD   CERR12
5778  023252  023217                          .WORD   CERR11
5779  023254  023121                          .WORD   CERR10
5780  023256  023047                          .WORD   CERR9
5781  023260  023005                          .WORD   CERR8
5782  023262  023217                          .WORD   CERR7
5783  023264  022754                          .WORD   CERR6
5784  023266  022674                          .WORD   CERR5
5785  023270  022621                          .WORD   CERR4
5786  023272  022564                          .WORD   CERR3
5787  023274  022534                          .WORD   CERR2
5788  023276  022457                          .WORD   CERR1
5789  023300  022410                          .WORD   CERR0
5790  023302  023217                          .WORD   CERR16
5791
5792                                  .SBTTL  TEMPORARY CONTROLLER REGISTER STORAGE
5793
5794  023304  000000                T.CS1:    .WORD   0             ;TEMPORARY STORAGE FOR COMMAND AND STATUS
5795                                                                ;  REGISTER 1
5796  023306  000000                T.CS2:    .WORD   0             ;TEMPORARY STORAGE FOR COMMAND AND STATUS
5797                                                                ;  REGISTER 2
5798  023310  000000                T.WCR:    .WORD   0             ;TEMPORARY STORAGE FOR WORD COUNT REGISTER
5799  023312  000000                T.BA:     .WORD   0             ;TEMPORARY STORAGE FOR BUS ADDRESS REGISTER
5800  023314  000000                T.DA:     .WORD   0             ;TEMPORARY STORAGE FOR DISK TRACK AND SECTOR
5801  023316  000000                T.DC:     .WORD   0             ;TEMPORARY STORAGE FOR DRIVE CYLINDER
5802  023320  000000                T.ASOF:   .WORD   0             ;TEMPORARY STORAGE FOR ATTENTION SUMMARY
5803                                                                ;  AND OFFSET
5804  023322  000000                T.ER:     .WORD   0             ;TEMPORARY STORAGE FOR ERROR REGISTER
5805  023324  000000                T.DS:     .WORD   0             ;TEMPORARY STORAGE FOR DRIVE STATUS REGISTER
5806  023326  000000                T.MR1:    .WORD   0             ;TEMPORARY STORAGE FOR MAINTENANCE REGISTER 1
5807  023330  000000                T.MR2:    .WORD   0             ;TEMPORARY STORAGE FOR MAINTENANCE REGISTER 2
5808  023332  000000                T.MR3:    .WORD   0             ;TEMPORARY STORAGE FOR MAINTENANCE REGISTER 3
5809  023334  000000                T.POS:    .WORD   0             ;TEMPORARY STORAGE FOR ECC POSITION
5810  023336  000000                T.PAT:    .WORD   0             ;TEMPORARY STORAGE FOR ECC PATTERN
5811  023340  000000                T.DB:     .WORD   0             ;TEMPORARY STORAGE FOR DATA BUFFER REGISTER
5812
5813                                  .SBTTL  DRIVER PARAMERTERS
5814
5815  023342  177440                RKBAS:    .WORD   177440        ;ADDRESS OF RK611 UNIBUS ADDRESS BLOCK
5816  023344  000210                RKVEC:    .WORD   210           ;ADDRESS OF R611 VECTOR
5817  023346  000240                RKPRI:    .WORD   PR5           ;RK611 INTERRUPT PRIORITY
5818  023350  013644                A.NORM:   ERRFRE                ;ADDRESS OF NORMAL RETURN FROM DRIVER
5819  023352  015036                A.ABNL:   DRVERR                ;ADDRESS OF ABNORMAL RETURN FROM DRIVER
5820  023354  016444                A.CONT:   CONERR                ;ADDRESS OF CONTROLLER ERROR RETURN
5821  023356  000000                E.CONT:   .WORD   0             ;CONTROLLER ERROR STATUS
5822                                                                ;  THIS LOCATION IS CLEARED WHEN EVERY COMMAND
5823                                                                ;  IS INITIATED.  IF A CONTROLLER ERROR
5824                                                                ;  OCCURS THE FOLLOWING BIT ASSIGNMENT IS
5825                                                                ;  USED:
5826                                                                ;
5827          000001                E.CCLR=   BIT0                ;CLEAR CONTROLLER DID NOT CLEAR ERROR
5828          000002                E.NOAT=   BIT1                ;NO ATTENTION IN ATTENTION SUMMARY REG
5829          000004                E.UATT=   BIT2                ;UNSOLICATED ATTENTION (SEQUENTIAL ONLY)
5830          000010                E.UDAT=   BIT3                ;UNEXPECTED DATA TYPE ERROR
```

```
                        000020          E.CLAT= BIT4                    ;ATTENTION DID NOT RESET WITH CLEAR
                        000040          E.SCLR= BIT5                    ;SUBSYSTEM CLEAR DID NOT CLEAR DRIVE
                                                                        ;  ATTENTION
                        000100          E.ILLD= BIT6                    ;ILLEGAL DRIVER COMMAND
                        000400          E.DLT=  BIT8                    ;DATA LATE WHEN UNLOADING HEADER
                        001000          E.CERR= BIT9                    ;CONTROLLER ERROR DURING DRIVER SERVICING
                        002000          E.DPAR= BIT10                   ;DRIVE DETECTED PARITY ERROR
                        040000          E.CMTO= BIT14                   ;CONTROLLER COMMAND TIME OUT (QUEUED ONLY)
                        100000          E.MDS=  BIT15                   ;MULTIPLE DRIVE SELECT

023360  000000          O.WAIT: .WORD   0               ;PARAMETER BLOCK OF THE DRIVE
                                                        ;WAITING FOR COMMAND COMPLETION
023362  000400          W.MTIM: .WORD   400             ;LOOP COUNTER FOR MILISECOND SCAN OF DRIVE
023364  000400          W.MILI: .WORD   400             ;16 MILISECOND TIME FOR PROGRAM
                                                        ;
                                                        ;       CPU     VALUE
                                                        ;       ---     -----
                                                        ;
                                                        ;       11/05   100
                                                        ;       11/10
                                                        ;       11/20
                                                        ;       11/34
                                                        ;       11/40
                                                        ;       11/45   400
                                                        ;       11/50
                                                        ;       11/70
023366  000100          W.SEC:  .WORD   100             ;SECOND COUNT COUNT FOR ALL COMMANDS
                                                        ;  EXCEPT START SPINDLE
023370  001000          W.BSEC: .WORD   1000            ;8 SECOND FOR DRIVE CYCLE DOWN
023372  010000          W.MIN:  .WORD   10000           ;MINUTE TIME FOR START SPINDLE
023374  000000          HDR.AD: .WORD   0               ;ADDRESS USED FOR READ ALL HEADERS
023376  000000          HDR.CT: .WORD   0               ;NUMBER OF HEADERS LEFT TO READ FOR READ
                                                        ;  ALL HEADERS
023400      000         I.ISRL: .BYTE   0               ;INTERRUPT OR RELEASED COMMAND ISSUED
023401      002 004 010 H.HEAD: .BYTE   2,4,10          ;HEAD DECODES
023404      000         W.TIME: .BYTE   0               ;DRIVES BEING WATCH-DOG TIMED

                        .SBTTL  INTERRUPT MASKS

023405      000         INTMSK: .BYTE   0               ;INTERRUPT MASKS FOR DRIVE IN PARAMETER BLOCK

                        ;       INTERRUPT MASK TABLE

023406      001         I.DRV:  .BYTE   1               ;INTERRUPT MASK FOR DRIVE 0
023407      002                 .BYTE   2               ;INTERRUPT MASK FOR DRIVE 1
023410      004                 .BYTE   4               ;INTERRUPT MASK FOR DRIVE 2
023411      010                 .BYTE   10              ;INTERRUPT MASK FOR DRIVE 3
023412      020                 .BYTE   20              ;INTERRUPT MASK FOR DRIVE 4
023413      040                 .BYTE   40              ;INTERRUPT MASK FOR DRIVE 5
023414      100                 .BYTE   100             ;INTERRUPT MASK FOR DRIVE 6
023415      200                 .BYTE   200             ;INTERRUPT MASK FOR DRIVE 7

                        .SBTTL  PARAMETER BLOCK TABLE

023416  002142          PBLKT:  PARM0                   ;ADDRESS OF PARAMETER BLOCK GIVEN WITH
                                                        ;DRIVE CALL.  MUST BE LOADED INTO PBLKT
```

```
 5887
 5888
 5889                                .SBTTL  TIME FOR WATCH-DOG TIMER
 5890
 5891  023420  000000                W.DRV:  .WORD   0               ;TIME FOR INSTRUCTION IN PARAMETER BLOCK
```

```
                                .SBTTL  RK611/RK06 UNIBUS DRIVER FOR SEQUENTIAL OPERATIONS (REV. 0.08)

                                ;*COPYRIGHT (C) 1975
                                ;*DIGITAL EQUIPMENT CORP.
                                ;*MAYNARD, MA. 01754
                                ;*AUTHOR: ROY SPITZER

                                .SBTTL   *WATCH-DOG TIMER

                                ;;*****************************************************************
                                ;*
                                ;*      THE WATCH-DOG TIMER DOES A PSEUDO-TIMING OF RK06 UNIBUS
                                ;*      SUBSYSTEM COMMAND. SINCE ONE CAN NOT GUARANTEE THAT A
                                ;*      REAL-TIME CLOCK (KW11-P OR KW11-L) IS ON THE SYSTEM
                                ;*      THE RK06 DRIVER WILL USE THE LOCATION W.MTIM FOR
                                ;*      MILLI-SECOND TIMING. WHEN W.MTIM REACHES ZERO THE
                                ;*      WATCH-DOG TIMER WILL SCAN THE DRIVES IN USE AS
                                ;*      DETERMINED BY THE LOCATION W.TIME. THE TIMER COUNTS
                                ;*      (ONE FOR EACH DRIVE) ARE KEPT IN THE TABLE W.DRV.
                                ;*      IF ANY COUNT IN THE TABLE W.DRV REACHES ZERO A COMMAND
                                ;*      TIME-OUT WILL BE DESIGNATED IN THE PROGRAM DEVICE STATUS
                                ;*      REGISTER OF THAT DRIVE'S PARAMETER BLOCK.
                                ;*
                                ;*      THE DRIVER WILL USE THE LOCATION W.MIN AS THE NUMBER
                                ;*      OF MILLISECONDS FOR AN UNLOAD OR START SPINDLE COMMAND.
                                ;*      THE DRIVER WILL USE THE LOCATION W.SEC AS THE TIME
                                ;*      LIMIT FOR ALL OTHER COMMANDS.
                                ;*
                                ;*      FOR QUEUED OPERATIONS THE WATCH-DOG TIMER WILL
                                ;*      WATCH UP TO 8 OPERATIONS SIMULATEOUSLY. FOR SEQUENTIAL
                                ;*      OPERATIONS ONLY ONE OPERATION WILL BE WATCHED.
                                ;*
                                ;*CALL   JSR     PC,W.WTCH
                                ;*      RETURN          IF NO DRIVE ORDER EXCEEDED ITS TIME LIMIT
                                ;*
                                ;*      OTHERWISE AN ABNORMAL RETURN TO THE ROUTINE ADDRESS
                                ;*      BY LOCATION A.ABNL WILL OCCUR AND THE CMDTO FLAG
                                ;*      IN THE PROGRAM DEVICE STATUS REGISTER OF THE
                                ;*      APPROPRIATE PARAMETER BLOCK WILL BE SET.
                                ;*
                                ;;*****************************************************************

023422  010546                  W.WTCH: MOV     R5,-(SP)        ;SAVE R5 ON THE STACK
023424  010446                          MOV     R4,-(SP)        ;SAVE R4 ON THE STACK
023426  010346                          MOV     R3,-(SP)        ;SAVE R3 ON THE STACK
023430  010246                          MOV     R2,-(SP)        ;SAVE R2 ON STACK
023432  013746  177776                  MOV     PS,-(SP)        ;SAVE PROGRAM STATUS WORD ON STACK
023436  005337  023362                  DEC     W.MTIM          ;DECREMENT MILLISECOND TIMER
023442  001034                          BNE     20$             ;IF NOT ZERO RETURN
023444  013737  023364  023362          MOV     W.MILI,W.MTIM   ;REINITIALIZE MILLISECOND TIMER
023452  105737  023404                  TSTB    W.TIME          ;CHECK IF DRIVE IS BEING TIMED
023456  001426                          BEQ     20$             ;NO, RETURN
023460  013737  023346  177776          MOV     RKPRI,PS        ;LOCK OUT RK06 INTERRUPTS
023466  013702  023342                  MOV     RKBAS,R2        ;LOAD BASE OF RK06 REGISTERS
023472  005337  023420                  DEC     W.DRV           ;DECREMENT COMMAND TIMER
023476  001016                          BNE     20$             ;RETURN IF NO TIME OUT
```

```
 5948  023500  105037  023404                    CLRB    W.TIME          ;RESET TIMING INDICATOR
 5949  023504  013705  023416                    MOV     PBLKT,R5        ;LOAD ADDRESS OF PARAMETER BLOCK
 5950                                                                    ;  TABLE FOR INDEXING
 5951  023510  052765  000100  000014            BIS     #CMDTO,P.PRST(R5) ;SET COMMAND TIME OUT
 5952  023516  020537  023360                    CMP     R5,O.WAIT       ;CHECK IF DRIVER IS WAITING FOR
 5953                                                                    ;  COMMAND COMPLETION
 5954  023522  001002                            BNE     5$              ;NO, DO NOT ALTER WAITING FOR
 5955                                                                    ;  COMMAND COMPLETION
 5956  023524  005037  023360                    CLR     O.WAIT          ;CLEAR WAIT FOR COMMAND COMPLETION
 5957  023530  004737  027004          5$:       JSR     PC,R.ABNL       ;BRANCH TO ERROR ROUTINE
 5958  023534  012637  177776          20$:      MOV     (SP)+,PS        ;RESTORE PSW
 5959  023540  012602                            MOV     (SP)+,R2        ;RESTORE R2
 5960  023542  012603                            MOV     (SP)+,R3        ;RESTORE R3
 5961  023544  012604                            MOV     (SP)+,R4        ;RESTORE R4
 5962  023546  012605                            MOV     (SP)+,R5        ;RESTORE R5
 5963  023550  000207                            RTS     PC              ;RETURN
```

```
                                .SBTTL   *RK06 INTERRUPT SERVICE ROUTINE

                                ;;*****************************************************************
                                ;*
                                ;*      THIS ROUTINE WILL SERVICE ALL RK06 INTERRUPTS.
                                ;*
                                ;*      UPON RECEIVING AN INTERRUPT, THIS ROUTINE WILL
                                ;*      PERFORM ONE OF THE FOLLOWING SERVICES:
                                ;*
                                ;*      1.) SERVICE PORT WAS SEIZED BY OTHER PORT
                                ;*      2.) SERVICE DRIVER IS WAIT FOR COMMAND COMPLETION
                                ;*      3.) SERVICE POSITIONING COMPLETION
                                ;*      4.) REQUEUE COMMAND IF DRIVE WAS RELEASED
                                ;*          FOR THE QUEUED RK06 DRIVER.
                                ;*      5.) IF NO SERVICE IS REQUIRED, THE COMMAND WILL BE ISSUED
                                ;*          FOR THE QUEUED RK06 DRIVER.
                                ;*
                                ;*      THREE LINKS ARE PROVIDED TO THE DRIVING PROGRAM.
                                ;*      THEY ARE:
                                ;*
                                ;*      1.) A.NORM ADDRESS OF NORMAL RETURN (SUCESSFUL COMPLETION OF COMMAND)
                                ;*      2.) A.ABNL ADDRESS OF ABNORMAL RETURN (UNSUCESSFUL COMPLETION OF COMMAND)
                                ;*      3.) A.CONT ADDRESS OF CONTROL ERROR RETURN
                                ;*
                                ;*      FOR NORMAL AND ABNORMAL RETURNS, THE ADDRESS OF THE APPROPRIATE
                                ;*      PARAMETER BLOCK WILL BE IN R5.
                                ;*
                                ;*      FOR THE CONTROLLER ERROR RETURN, THE LOCATION E.CONT CONTAINS
                                ;*      THE REASON FOR THE CONTROLLER ERROR.
                                ;*
                                ;*      ROUTINES USED:
                                ;*              C.OPT   (QUEUED ONLY)
                                ;*              Q.PUSH (QUEUED ONLY)
                                ;*              Q.RMOV (QUEUED ONLY)
                                ;*              R.CONT (SEQUENTIAL ONLY)
                                ;*              R.NORM (SEQUENTIAL ONLY)
                                ;*              R.ABNL (SEQUENTIAL ONLY)
                                ;*              I.CSTS
                                ;*              I.STAT
                                ;*              I.ISSU
                                ;*              I.CCLR
                                ;*
                                ;;*****************************************************************

023552  010546          I.INTR: MOV     R5,-(SP)        ;STORE R5 ON THE STACK
023554  010446                  MOV     R4,-(SP)        ;STORE R4 ON THE STACK
023556  010346                  MOV     R3,-(SP)        ;STORE R3 ON THE STACK
023560  010246                  MOV     R2,-(SP)        ;STORE R2 ON THE STACK
023562  010146                  MOV     R1,-(SP)        ;STORE R1 ON THE STACK
023564  010046                  MOV     R0,-(SP)        ;STORE R0 ON THE STACK
023566  013702  023342          MOV     RKBAS,R2        ;LOAD R2 TO ADDRESS RK06 REGISTER
023572  016237  000010  023306  MOV     RKCS2(R2),T.CS2 ;STORE CS2
023600  032737  001000  023306  BIT     #MDS,T.CS2      ;CHECK IF MULTIPLE DRIVE SELECT
023606  001407                  BEQ     1$              ;NO, CONTINUE PROCESSING
023610  052737  100000  023356  BIS     #E.MDS,E.CONT   ;SET MULTIPLE DRIVE SELECT
023616  004737  027030          JSR     PC,R.CONT       ;REPORT ERROR
```

```
6020  023622  000137  026002                    JMP     I.RTRN          ;RETURN
6021
6022  023626  105737  023400          1$:       TSTB    I.ISRL          ;CHECK IF INTERRUPT OR RELEASE
6023  023632  001410                            BEQ     6$              ;NO, CHECK IF DRIVE AVAILABLE
6024  023634  100403                            BMI     5$              ;CHECK IF RELEASE COMMAND
6025  023636  105037  023400                    CLRB    I.ISRL          ;YES,CLEAR FLAG
6026  023642  000473                            BR      I.IOO           ;CONTINUE PROCESSING INTERRUPT
6027
6028  023644  105037  023400          5$:       CLRB    I.ISRL          ;CLEAR FLAG
6029  023650  000137  024764                    JMP     I.ATTN          ;GO PROCESS DRIVE ATTENTIONS
6030
6031  023654  032737  010400  023306  6$:       BIT     #NED!UFE,T.CS2  ;CHECK FOR NON-EXISTENT DRIVE OR
6032                                                                    ;  UNIT FIELD ERROR
6033  023662  001413                            BEQ     7$              ;NO, WAIT FOR DUAL ACCESS INTERRUPT
6034  023664  013704  023306                    MOV     T.CS2,R4        ;LOAD R4 FOR DRIVE NUMBER
6035  023670  042704  177770                    BIC     #^C<DRVMSK>,R4  ;KEEP DRIVE BITS
6036  023674  013705  023416                    MOV     PBLKT,R5        ;STORE PARAMETER BLOCK ADDRESS
6037  023700  016237  000000  023304            MOV     RKCS1(R2),T.CS1 ;LOAD TEMPORARY CS1 FOR STATUS REPORT
6038  023706  000137  024174                    JMP     I.ERRC          ;REPORT ERROR
6039
6040  023712  016237  000012  023324  7$:       MOV     RKDS(R2),T.DS   ;STORE STATUS REGISTER FOR COMPARISON
6041  023720  032737  000001  023324            BIT     #DRA,T.DS       ;CHECK IF DRIVE SEIZED BY OTHER
6042                                                                    ;  PORT
6043  023726  001041                            BNE     I.IOO           ;NO, CONTINUE PROCESSING INTERRUPT
6044
6045                                            ;CHECK IF ANY DATA TRANSFER ERROR EXISTS
6046  023730  032737  164000  023306            BIT     #DLT!WCE!UPE!NEM,T.CS2
6047
6048  023736  001007                            BNE     10$             ;INDICATE ERROR
6049  023740  016237  000014  023322            MOV     RKER(R2),T.ER   ;STORE ERROR REGISTER
6050
6051                                  ;         CHECK FOR DATA TRANSFER ERROR TYPE ERROR
6052  023746  032737  125700  023322            BIT     #DCK!OPI!WLE!COE!HVRC!BSE!ECH,T.ER
6053
6054  023754  001407                            BEQ     11$             ;NO, WAIT FOR RELEASE OF RK06 DRIVE
6055
6056  023756  052737  000010  023356  10$:      BIS     #E.UDAT,E.CONT  ;SET UNEXPECTED DATA TYPE ERROR
6057  023764  004737  027030                    JSR     PC,R.CONT       ;REPORT ERROR
6058  023770  000137  026002                    JMP     I.RTRN          ;RESTORE REGISTERS
6059
6060  023774  105037  023404          11$:      CLRB    W.TIME          ;RESET TIMING ON THIS DRIVE
6061  024000  005037  023420                    CLR     W.DRV           ;CLEAR TIMING COUNT FOR THIS DRIVE
6062  024004  013705  023416                    MOV     PBLKT,R5        ;LOAD R5 WITH PARAMETER BLOCK
6063                                                                    ;  ADDRESS
6064  024010  052765  010000  000014            BIS     #DRVSZD,P.PRST(R5) ;SET DRIVE SEIZED IN THE
6065                                                                    ;  PROGRAM DRIVE STATUS REGISTER
6066  024016  005037  023360                    CLR     0.WAIT          ;CLEAR WAIT FOR COMMAND COMPLETION
6067  024022  004737  027004                    JSR     PC,R.ABNL       ;INDICATE ABNORMAL TERMINATION
6068  024026  000137  026002                    JMP     I.RTRN          ;GO RESTORE REGISTERS
6069
6070  024032  013705  023360          I.IOO:    MOV     0.WAIT,R5       ;LOAD PARAMETER BLOCK ADDRESS INTO R5
6071  024036  001002                            BNE     2$              ;IS COMMAND WAITING PROCESSING
6072                                                                    ;  YES, DO PROCESSING
6073  024040  000137  024764                    JMP     I.ATTN          ;NO, PROCESS ATTENTION
6074
6075  024044  013704  023306          2$:       MOV     T.CS2,R4        ;STORE RKCS2 FOR DRIVE NUMBER
```

```
 6076  024050  042704  177770                    BIC     #^C<DRVMSK>,R4  ;MASK OUT UNNECESSARY BITS
 6077
 6078
 6079  024054  126504  000000                    CMPB    P.DRVN(R5),R4   ;CHECK IF DRIVE NUMBER IS EXPECTED
 6080  024060  001401                            BEQ     3$              ;YES, CONTINUE
 6081  024062  000000                            HALT                    ;NO, DRIVER ERROR
 6082  024064  122765  000164  000001  3$:      CMPB    #RDALHD,P.CMND(R5) ;CHECK IF READ ALL HEADERS
 6083  024072  001002                            BNE     10$             ;NO, EXECUTE NORMAL DATA TRANSFER
 6084  024074  000137  024432                    JMP     I.HDAL          ;GO EXECUTE SPECIAL HEADER SEQUENCE
 6085
 6086  024100  005037  023360          10$:     CLR     O.WAIT          ;CLEAR WAIT FOR COMMAND COMPLETION
 6087  024104  005037  023420                    CLR     W.DRV           ;CLEAR WATCH-DOG TIME
 6088  024110  105037  023404                    CLRB    W.TIME          ;RESET TIMING ON THIS DRIVE
 6089  024114  016237  000000  023304            MOV     RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REGISTER 1
 6090  024122  032737  100000  023304            BIT     #CERR,T.CS1     ;CHECK IF CONTROLLER ERROR
 6091  024130  001021                            BNE     I.ERRC          ;YES, PROCESS ERROR
 6092  024132  016237  000016  023320            MOV     RKASOF(R2),T.ASOF ;STORE ATTENTION SUMMARY
 6093  024140  133737  023405  023321            BITB    INTMSK,T.ASOF+1 ;CHECK IF DRIVE ATTENTION SET
 6094  024146  001004                            BNE     15$             ;YES, REPORT ERROR
 6095  024150  004737  027016                    JSR     PC,R.NORM       ;INDICATE NORMAL RETURN
 6096  024154  000137  026002                    JMP     I.RTRN          ;RESTORE REGISTERS
 6097
 6098  024160  052765  000010  000014  15$:      BIS     #UEXATT,P.PRST(R5) ;SET UNEXPECTED ATTENTION
 6099
 6100  024166  004737  026452          I.ERRA:  JSR     PC,I.CSTS       ;STORE CONTROLLER STATUS
 6101  024172  000405                            BR      I.ERR           ;STORE PATTERN AND POSITION INFORMATION
 6102
 6103  024174  013765  023304  000016  I.ERRC:  MOV     T.CS1,P.CS1(R5) ;GET ERROR RKCS1
 6104  024202  004737  026474                    JSR     PC,I.CST1       ;GET REST OF CONTROLLER STATUS
 6105  024206  016265  000032  000062  I.ERR:   MOV     RKECPT(R2),P.EPAT(R5) ;STORE ECC PATTERN
 6106  024214  016265  000030  000060            MOV     RKECPS(R2),P.EPOS(R5) ;STORE ECC POSITION
 6107  024222  004037  026020                    JSR     R0,I.CCLR       ;CLEAR CONTROLLER
 6108  024226  026002                            I.RTRN                  ;ERROR RETURN
 6109  024230  032765  010400  000020            BIT     #NED!UFE,P.CS2(R5) ;CHECK IF IT WAS NON-EXISTENT DRIVE OR
 6110                                                                    ;  UNIT FIELD ERROR
 6111  024236  001046                            BNE     5$              ;YES, REPORT ERROR
 6112  024240  004037  026556                    JSR     R0,I.STAT       ;GATHER DRIVE STATUS
 6113  024244  026002                            I.RTRN                  ;ERROR RETURN
 6114  024246  112737  000005  023304            MOVB    #DR.CLR,T.CS1   ;LOAD COMMAND
 6115  024254  004037  026102                    JSR     R0,I.ISSU       ;ISSUE DRIVE CLEAR
 6116  024260  026002                            I.RTRN                  ;ERROR RETURN
 6117  024262  133737  023405  023321            BITB    INTMSK,T.ASOF+1 ;CHECK IF ATTENTION RESET
 6118  024270  001407                            BEQ     2$              ;NO, INDICATE DRIVE ERROR
 6119  024272  052737  000020  023356            BIS     #E.CLAT,E.CONT  ;SET ATTENTION DID NOT RESET
 6120                                                                    ;  WITH CLEAR
 6121  024300  004737  027030                    JSR     PC,R.CONT       ;REPORT CONTROLLER ERROR
 6122  024304  000137  026002                    JMP     I.RTRN          ;GO RESTORE REGISTERS
 6123
 6124  024310  032737  040000  023330  2$:      BIT     #S.DSC,T.MR2    ;CHECK IF DRIVE STATUS CHANGE CLEARED
 6125  024316  001403                            BEQ     3$              ;YES, CHECK FAULT
 6126  024320  052765  000040  000014            BIS     #DRVDSC,P.PRST(R5) ;SET DSC DID NOT CLEAR
 6127  024326  032737  001000  023332  3$:      BIT     #S.PAR,T.MR3    ;CHECK IF DRIVE PARITY ERROR
 6128  024334  001407                            BEQ     5$              ;NO, INDICATE ABNORMAL TERMINATION
 6129  024336  052737  002000  023356            BIS     #E.DPAR,E.CONT  ;SET DRIVE PARITY ERROR
 6130  024344  004737  027030                    JSR     PC,R.CONT       ;INDICATE CONTROLLER ERROR
 6131  024350  000137  026002                    JMP     I.RTRN          ;RETURN
```

```
6132
6133  024354  032765  000020  000014  5$:     BIT     #DRVHRD,P.PRST(R5) ;CHECK IF HARD DRIVE ERROR
6134  024362  001017                          BNE     10$                ;YES, GO REPORT ERROR
6135  024364  032737  020000  023330          BIT     #S.PIP,T.MR2       ;CHECK IF DRIVE IS CYCLING DOWN
6136  024372  001413                          BEQ     10$                ;NO, REPORT ERROR
6137  024374  052765  020000  000014          BIS     #E.UNLD,P.PRST(R5) ;SET DRIVE UNLOADING
6138  024402  113737  023405  023404          MOVB    INTMSK,W.TIME      ;SET UP 8 SECONDS FOR DRIVE TO CYCLE UP
6139  024410  013737  023370  023420          MOV     W.BSEC,W.DRV
6140  024416  000137  026002                  JMP     I.RTRN             ;GO RESTORE REGISTERS
6141
6142  024422  004737  027004          10$:    JSR     PC,R.ABNL          ;GO REPORT ERROR
6143  024426  000137  026002                  JMP     I.RTRN             ;GO RESTORE REGISTERS
6144
6145                                  .SBTTL   *READ ALL HEADERS INTERRUPT SEQUENCE
6146
6147  024432  016237  000000  023304  I.HDAL: MOV     RKCS1(R2),T.CS1 ;STORE CS1 TO CHECK CONTROLLER
6148                                                                  ;  ERROR
6149  024440  032737  100000  023304          BIT     #CERR,T.CS1        ;CHECK IF CONTROLLER ERROR
6150  024446  001422                          BEQ     5$                 ;NO, CHECK FOR ATTENTION
6151
6152  024450  005037  023360                  CLR     0.WAIT             ;CLEAR WAITING FOR COMMAND COMPLETE
6153  024454  105037  023404                  CLRB    W.TIME             ;RESET TIMING ON DRIVE
6154  024460  005037  023420                  CLR     W.DRV              ;CLEAR TIME OUT COUNT
6155  024464  013765  023304  000016          MOV     T.CS1,P.CS1(R5)    ;STORE ERROR RKCS1
6156  024472  004737  026474                  JSR     PC,I.CST1          ;STORE CONTROLLER REGISTERS
6157  024476  004037  026020                  JSR     R0,I.CCLR          ;CLEAR CONTROLLER
6158  024502  026002                          I.RTRN                     ;ERROR RETURN
6159  024504  004737  027004                  JSR     PC,R.ABNL          ;INDICATE ERROR RETURN
6160  024510  000137  026002                  JMP     I.RTRN             ;RESTORE REGISTERS
6161
6162  024514  016537  000016  023320  5$:     MOV     RKASOF(R5),T.ASOF ;STORE ATTENTION SUMMARY
6163  024522  133737  023405  023321          BITB    INTMSK,T.ASOF+1    ;CHECK IF DRIVE ATTENTION IS SET
6164  024530  001410                          BEQ     7$                 ;NO, CHECK IF READ ALL HEADERS
6165  024532  005037  023360                  CLR     0.WAIT             ;CLEAR WAITING FOR COMMAND COMPLETION
6166  024536  105037  023404                  CLRB    W.TIME             ;RESET TIMING ON DRIVE
6167  024542  005037  023420                  CLR     W.DRV              ;CLEAR TIME OUT COUNT
6168  024546  000137  024166                  JMP     I.ERRA             ;GO REPORT ERROR
6169
6170  024552  013701  023374          7$:     MOV     HDR.AD,R1          ;GET MAIN MEMORY ADDRESS
6171  024556  016221  000024                  MOV     RKDB(R2),(R1)+     ;GET FIRST WORD OF HEADER
6172  024562  016221  000024                  MOV     RKDB(R2),(R1)+     ;GET SECOND WORD OF HEADER
6173  024566  016221  000024                  MOV     RKDB(R2),(R1)+     ;GET THIRD WORD OF HEADER
6174  024572  010137  023374                  MOV     R1,HDR.AD          ;STORE ADDRESS FOR NEXT HEADER
6175  024576  016237  000010  023306          MOV     RKCS2(R2),T.CS2 ;STORE CS2 TO CHECK FOR DATA LATE
6176  024604  032737  100000  023306          BIT     #DLT,T.CS2         ;CHECK FOR DATA LATE
6177  024612  001055                          BNE     35$                ;YES, REPORT ERROR
6178  024614  005337  023376                  DEC     HDR.CT             ;DECREMENT NUMBER OF HEADER YET TO READ
6179  024620  001026                          BNE     25$                ;IF NON-ZERO, GO ISSUE NEXT READ HEADER
6180  024622  005037  023360                  CLR     0.WAIT             ;CLEAR DRIVER WAITING FOR COMMAND COMPLETION
6181  024626  005037  023420                  CLR     W.DRV              ;CLEAR TIME OUT COUNT FOR THIS DRIVE
6182  024632  105037  023404                  CLRB    W.TIME             ;CLEAR WATCH DOG TIME ON THIS DRIVE
6183  024636  012762  000003  000026          MOV     #3,RKMR1(R2)       ;LOAD MAINTENANCE REGISTER FOR SECTOR COUNT
6184  024644  112737  000001  023304          MOVB    #DR.SEL,T.CS1      ;LOAD SELECT COMMAND
6185  024652  004037  026102                  JSR     R0,I.ISSU          ;GET SECTOR COUNT
6186  024656  026002                          I.RTRN                     ;ERROR RETURN
6187  024660  013765  023332  000056          MOV     T.MR3,P.B11(R5)    ;LOAD SECTOR COUNT
```

```
6188  024666  004737  027016                  JSR     PC,R.NORM       ;INDICATE NORMAL TERMINATION
6189  024672  000137  026002                  JMP     I.RTRN          ;RESTORE REGISTERS
6190
6191  024676  016562  000002  000020  25$:    MOV     P.CYLN(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS REGISTER
6192  024704  016562  000004  000006          MOV     P.SECT(R5),RKDA(R2) ;LOAD SECTOR AND TRACK
6193  024712  116565  000007  000017          MOVB    P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
6194  024720  042765  165777  000016          BIC     #^C<CDT!CFMT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT FORMAT AND
6195                                                                  ;  DRIVE TYPE
6196  024726  112765  000125  000016          MOVB    #RDHEAD,P.CS1(R5)  ;STORE COMMAND ISSUED
6197  024734  016562  000016  000000          MOV     P.CS1(R5),RKCS1(R2)  ;ISSUE READ HEADER
6198  024742  000137  026002                  JMP     I.RTRN          ;RESTORE REGISTERS
6199
6200  024746  052737  000400  023356  35$:    BIS     #E.DLT,E.CONT   ;SET DATA LATE WHILE UNLOADING HEADER
6201  024754  004737  027030                  JSR     PC,R.CONT       ;REPORT ERROR
6202  024760  000137  026002                  JMP     I.RTRN          ;RESTORE REGISTERS
6203
6204                                  .SBTTL   *DRIVE ATTENTION SCANNER
6205
6206  024764  016237  000000  023304  I.ATTN: MOV     RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS
6207                                                                  ;  REGISTER 1 FOR COMPARISON
6208  024772  032737  100000  023304          BIT     #CERR,T.CS1     ;CHECK IF CONTROLLER ERROR OCCURRED
6209  025000  001441                          BEQ     5$              ;NO, CHECK IF ATTENTION
6210
6211                                          ;CHECK IF ANY DATA TRANSFER TYPE ERROR EXISTS
6212  025002  032737  164000  023306          BIT     #DLT!WCE!UPE!NEM,T.CS2
6213
6214  025010  001007                          BNE     1$              ;INDICATE ERROR
6215  025012  016237  000014  023322          MOV     RKER(R2),T.ER   ;STORE ERROR REGISTER
6216
6217                                  ;       CHECK FOR DATA TRANSFER ERROR TYPE
6218  025020  032737  125700  023322          BIT     #DCK!OPI!WLE!COE!HVRC!BSE!ECH,T.ER
6219
6220  025026  001407                          BEQ     2$              ;NO DATA TRANSFER ERROR
6221
6222  025030  052737  000010  023356  1$:     BIS     #E.UDAT,E.CONT  ;SET UNEXPECTED DATA TYPE ERROR
6223  025036  004737  027030                  JSR     PC,R.CONT       ;REPORT ERROR
6224  025042  000137  026002                  JMP     I.RTRN          ;RESTORE REGISTERS
6225
6226  025046  013704  023306          2$:     MOV     T.CS2,R4        ;SAVE CS2 FOR REGISTER NUMBER
6227  025052  042704  177770                  BIC     #^C<DRVMSK>,R4  ;STRIP OFF JUNK
6228  025056  105037  023404                  CLRB    W.TIME          ;CLEAR WATCH DOG TIMER
6229  025062  005037  023420                  CLR     W.DRV           ;RESET TIMER VALUE
6230  025066  013705  023416                  MOV     PBLKT,R5        ;STORE PARAMETER BLOCK ADDRESS IN R5
6231
6232                                  ;       CLEAR DRIVE POSITIONING AND DRIVE POSITIONED FOR DATA TRANSFER
6233                                  ;         IN PROGRAM DEVICE STATUS REGISTER
6234  025072  042765  000006  000014          BIC     #DRVPOS!DRVPDT,P.PRST(R5)
6235
6236  025100  000137  024174                  JMP     I.ERRC          ;GO REPORT ERROR
6237
6238  025104  032737  040000  023304  5$:     BIT     #DI,T.CS1       ;CHECK IF ANY DRIVE ATTENTION
6239  025112  001002                          BNE     6$              ;YES, PROCESS INTERRUPT
6240  025114  000137  026002                  JMP     I.RTRN          ;RESTORE REGISTERS
6241
6242  025120  016237  000016  023320  6$:     MOV     RKASOF(R2),T.ASOF ;STORE ATTENTION SUMMARY
6243  025126  105737  023321                  TSTB    T.ASOF+1        ;CHECK IF ANY ATTENTIONS SET
```

```
6244  025132  001007                        BNE     7$                ;YES GO PROCESS INTERRUPT
6245  025134  052737  000002  023356        BIS     #E.NOAT,E.CONT    ;SET NO ATTENTION IN ATTENTION SUMMARY
6246  025142  004737  027030                JSR     PC,R.CONT         ;GO REPORT ERROR
6247  025146  000137  026002                JMP     I.RTRN            ;GO RESTORE REGISTERS
6248
6249  025152  133737  023405  023321  7$:   BITB    INTMSK,T.ASOF+1   ;CHECK IF DESIRED INTERRUPT
6250  025160  001007                        BNE     8$                ;YES, GO PROCESS IT
6251  025162  052737  000004  023356        BIS     #E.UATT,E.CONT    ;SET UNSOLICATED ATTENTION
6252  025170  004737  027030                JSR     PC,R.CONT         ;GO REPORT ERROR
6253  025174  000137  026002                JMP     I.RTRN            ;GO RESTORE REGISTERS
6254
6255  025200  013705  023416        8$:     MOV     PBLKT,R5          ;STORE PARAMETER BLOCK TABLE
6256  025204  116504  000000                MOVB    P.DRVN(R5),R4     ;STORE DRIVE NUMBER
6257  025210  032765  020000  000014        BIT     #E.UNLD,P.PRST(R5)  ;CHECK IF DRIVE UNLOADING
6258  025216  001402                        BEQ     11$               ;NO, CONTINUE
6259  025220  000137  025702                JMP     I.UNLD            ;SERVICE DRIVE IN POSITION AFTER ERROR
6260
6261  025224  042765  000002  000014  11$:   BIC     #DRVPOS,P.PRST(R5)  ;RESET DRIVE POSITIONING
6262  025232  005062  000026                CLR     RKMR1(R2)         ;CLEAR MAINTENANCE REGISTER 1
6263  025236  112737  000001  023304        MOVB    #DR.SEL,T.CS1     ;LOAD COMMAND
6264  025244  004037  026102                JSR     R0,I.ISSU         ;SELECT DRIVE WITH ATTENTION HIGH
6265  025250  026002                        I.RTRN                    ;ERROR RETURN
6266  025252  013765  023332  000042        MOV     T.MR3,P.B00(R5)   ;STORE STATUS BYTE 00 MESS B
6267  025260  032765  000200  000042        BIT     #S.FLT,P.B00(R5)  ;CHECK IF DRIVE FAULT
6268  025266  001401                        BEQ     12$               ;NO, CHECK FOR DRIVE STATUS CHANGE
6269  025270  000461                        BR      I.AERR            ;PROCESS ERROR
6270
6271  025272  013765  023330  000040  12$:   MOV     T.MR2,P.A00(R5)   ;STORE MAINTENANCE REGISTER 2
6272  025300  032765  040000  000040        BIT     #S.DSC,P.A00(R5)  ;CHECK FOR DRIVE STATUS CHANGE
6273  025306  001004                        BNE     13$               ;YES, PROCESS DRIVE STATUS CHANGE
6274  025310  052765  004000  000014        BIS     #NODSC,P.PRST(R5)  ;SET NO DRIVE STATUS CHANGE
6275  025316  000446                        BR      I.AERR            ;PROCESS ERROR
6276
6277  025320  112737  000005  023304  13$:   MOVB    #DR.CLR,T.CS1     ;LOAD COMMAND
6278  025326  004037  026102                JSR     R0,I.ISSU         ;CLEAR DRIVE STATUS CHANGE
6279  025332  026002                        I.RTRN                    ;ERROR RETUN
6280  025334  013765  023320  000032        MOV     T.ASOF,P.ASOF(R5)  ;STORE ATTENTION SUMMARY
6281  025342  133765  023405  000033        BITB    INTMSK,P.ASOF+1(R5)  ;CHECK IF ATTENTION RESET
6282  025350  001407                        BEQ     15$               ;YES, CONTINUE INTERRUPT PROCESSING
6283  025352  052737  000020  023356        BIS     #E.CLAT,E.CONT    ;SET ATTENTION DID NOT RESET
6284                                                                  ;  WITH DRIVE CLEAR
6285  025360  004737  027030                JSR     PC,R.CONT         ;FLAG ERROR
6286  025364  000137  026002                JMP     I.RTRN            ;RESTORE REGISTERS
6287
6288  025370  013765  023330  000040  15$:   MOV     T.MR2,P.A00(R5)   ;STORE MAINTENANCE REGISTER 2
6289  025376  032765  040000  000040        BIT     #S.DSC,P.A00(R5)  ;CHECK IF DRIVE STATUS CHANGE
6290                                                                  ;  RESET
6291  025404  001404                        BEQ     16$               ;YES, CONTINUE INTERRUPT PROCESSING
6292  025406  052765  000040  000014        BIS     #DRVDSC,P.PRST(R5)  ;SET DRIVE STATUS CHANGE DID NOT CLEAR
6293  025414  000407                        BR      I.AERR            ;GO PROCESS ERROR
6294
6295  025416  105037  023404        16$:   CLRB    W.TIME            ;RESET TIMING ON THIS DRIVE
6296  025422  005037  023420                CLR     W.DRV             ;CLEAR DRIVE TIMING COUNT
6297  025426  004737  027016                JSR     PC,R.NORM         ;REPORT SUCCESSFUL COMMAND COMPLETETION
6298  025432  000563                        BR      I.RTRN            ;RESTORE REGISTERS
6299
```

```
                                        .SBTTL   *ATTENTION ERROR HANDLER

025434  042765  000004  000014  I.AERR: BIC     #DRVPDT.P.PRST(R5) ;RESET POSITIONING IN PROGRESS BECAUSE
                                                                   ;  OF DATA TRANSFER
025442  105037  023404                  CLRB    W.TIME             ;CLEAR TIMING FOR THIS DRIVE
025446  005037  023420                  CLR     W.DRV              ;RESET WATCH-DOG TIME
025452  042765  177741  000016          BIC     #177741,P.CS1(R5)  ;KEEP COMMAND ISSUED
025460  042737  000036  023304          BIC     #36,T.CS1          ;KEEP CURRENT CONTROLLER STATUS
025466  053765  023304  000016          BIS     T.CS1,P.CS1(R5)    ;MAKE GOOD MESSAGE
025474  013765  023306  000020          MOV     T.CS2,P.CS2(R5)    ;STORE CONTROLLER REGISTERS
025502  013765  023310  000022          MOV     T.WCR,P.WCR(R5)
025510  013765  023312  000024          MOV     T.BA,P.BAR(R5)
025516  013765  023314  000026          MOV     T.DA,P.DTS(R5)
025524  013765  023316  000030          MOV     T.DC,P.DCYL(R5)
025532  013765  023320  000032          MOV     T.ASOF,P.ASOF(R5)
025540  013765  023322  000034          MOV     T.ER,P.ER(R5)
025546  013765  023324  000036          MOV     T.DS,P.DS(R5)
025554  004037  026556                  JSR     R0,I.STAT          ;GATHER DRIVE STATUS
025560  026002                          I.RTRN                     ;ERROR RETURN
025562  112737  000005  023304          MOVB    #DR.CLR,T.CS1      ;LOAD COMMAND
025570  004037  026102                  JSR     R0,I.ISSU          ;CLEAR DRIVE ERRORS
025574  026002                          I.RTRN                     ;ERROR RETURN
025576  133737  023405  023321          BITB    INTMSK,T.ASOF+1    ;CHECK IF ATTENTION RESET
025604  001407                          BEQ     2$                 ;YES, FLAG DRIVE ERROR
025606  052737  000020  023356          BIS     #E.CLAT,E.CONT     ;SET ATTENTION DID NOT RESET
025614  004737  027030                  JSR     PC,R.CONT          ;REPORT ERROR
025620  000137  026002                  JMP     I.RTRN             ;RESTORE REGISTERS

025624  032765  000020  000014  2$:     BIT     #DRVHRD,P.PRST(R5) ;CHECK IF AHARD DRIVE ERROR
025632  001017                          BNE     10$                ;YES, REPORT ERROR
025634  032737  020000  023330          BIT     #S.PIP,T.MR2       ;CHECK IF DRIVE IS UNLOADING
025642  001413                          BEQ     10$                ;NO, REPORT ERROR
025644  052765  020000  000014          BIS     #E.UNLD,P.PRST(R5) ;SET DRIVE UNLOADING DUE TO ERROR
025652  113737  023405  023404          MOVB    INTMSK,W.TIME      ;SET TIMING ON THIS DRIVE
025660  013737  023370  023420          MOV     W.8SEC,W.DRV       ;LOAD 8 SECONDS FOR CYCLE UP TIME
025666  000137  026002                  JMP     I.RTRN             ;RESTORE REGISTERS

025672  004737  027004          10$:    JSR     PC,R.ABNL          ;REPORT ERROR
025676  000137  026002                  JMP     I.RTRN             ;RESTORE REGISTERS

                                        .SBTTL   *ERROR CAUSING DRIVE TO UNLOAD

025702  052765  020000  000014  I.UNLD: BIS     #E.UNLD,P.PRST(R5) ;CLEAR DRIVE UNLOADING BECAUSE OF ERROR
025710  112737  000005  023304          MOVB    #DR.CLR,T.CS1      ;LOAD IN DRIVE CLEAR
025716  004037  026102                  JSR     R0,I.ISSU          ;GO ISSUE DRIVE CLEAR
025722  026002                          I.RTRN                     ;ERROR RETURN
025724  136437  023405  023321          BITB    INTMSK(R4),T.ASOF+1 ;CHECK IF ATTENTION CLEARED
025732  001406                          BEQ     15$                ;YES, CONTINUE
025734  012737  000020  023356          MOV     #E.CLAT,E.CONT     ;SET ATTENTION DID NOT RESET
025742  004737  027030                  JSR     PC,R.CONT          ;REPORT ERROR
025746  000415                          BR      I.RTRN             ;RESTORE REGISTERS

025750  032737  040000  023330  15$:    BIT     #S.DSC,T.MR2       ;CHECK IF DRIVE STAUS CHANGE RESET
025756  001403                          BEQ     20$                ;YES, CONTINUE
025760  052765  000040  000014          BIS     #DRVDSC,P.PRST(R5) ;SET DRIVE STAUS CHANGE DID NOT CLEAR
025766  105037  023404          20$:    CLRB    W.TIME             ;RESET TIMING ON THIS DRIVE
```

F11


```
 6356  025772  005037  023420                  CLR     W.DRV           ;CLEAR TIME COUNT
 6357  025776  004737  027004                  JSR     PC,R.ABNL       ;REPORT ERROR
 6358
 6359  026002  012600              I.RTRN:     MOV     (SP)+,R0        ;RESTORE R0
 6360  026004  012601                          MOV     (SP)+,R1        ;RESTORE R1
 6361  026006  012602                          MOV     (SP)+,R2        ;RESTORE R2
 6362  026010  012603                          MOV     (SP)+,R3        ;RESTORE R3
 6363  026012  012604                          MOV     (SP)+,R4        ;RESTORE R4
 6364  026014  012605                          MOV     (SP)+,R5        ;RESTORE R5
 6365  026016  000002                          RTI                     ;RETURN
 6366
```

```
                                        .SBTTL   *CONTROLLER CLEAR ROUTINE

                                        ;;*****************************************************************
                                        ;*
                                        ;*      THIS ROUTINE WILL BE USED BY THE DRIVER TO CLEAR THE CONTROLLER
                                        ;*      AND CHECK IF THE CONTROLLER ERRORS ARE RESET.  IF THE ERROR IS NOT
                                        ;*      CLEARED, THE ROUTINE AS SPECIFIED IN A.CONT WILL BE CALLED WITH
                                        ;*      E.CCLR SET IN E.CONT.
                                        ;*
                                        ;*      REGISTER        USE
                                        ;*      --------        ---
                                        ;*
                                        ;*      R2              ADDRESS OF RK06 REGISTERS
                                        ;*      R5              ADDRESS OF PARAMETER BLOCK
                                        ;*
                                        ;*CALL  JSR     R0,I.CCLR
                                        ;*      <ADDRESS OF ERROR RETURN>
                                        ;*      RETURN
                                        ;*
                                        ;;*****************************************************************

026120  012762  100000  000000  I.CCLR: MOV     #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
026026  016237  000000  023304          MOV     RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REGISTER 1
026034  032737  100000  023304          BIT     #CERR,T.CS1     ;CHECK IF CONTROLLER CLEAR DID
                                                                ;  CLEAR ERROR
026042  001407                          BEQ     5$              ;YES, RETURN TO DRIVER PROCESSING
026044  052737  000001  023356          BIS     #E.CCLR,E.CONT  ;SET CLEAR CONTROLLER DID NOT CLEAR ERROR
026052  004737  027030                  JSR     PC,R.CONT       ;REPORT CONTROLLER ERROR
026056  011000                          MOV     (R0),R0         ;SET UP ERROR RETURN
026060  000200                          RTS     R0              ;RETURN

026062  012762  000100  000000  5$:     MOV     #IE,RKCS1(R2)   ;SET INTERRUPT ENABLE
026070  112737  177777  023400          MOVB    #-1,I.ISRL      ;SET INTERRUPT ENABLE ISSUED
026076  005720                          TST     (R0)+           ;ADJUST FOR NORMAL RETURN
026100  000200                          RTS     R0              ;RETURN
```

```
                                .SBTTL   *COMMAND ISSUED BY DRIVER SERVICE ROUTINE

                                ;;*****************************************************************
                                ;*
                                ;*      THIS ROUTINE WILL ISSUE THE COMMAND AS SPECIFIED IN T.CS1
                                ;*      AND CHECK IF A CONTROLLER ERROR OCCURRED.  IF A CONTROLLER
                                ;*      ERROR OCCURRED, E.CERR WILL BE SET IN E.CONT AND
                                ;*      CONTROL WILL BE TURN OVER TO THE ROUTINE SPECIFIED BY THE
                                ;*      ADDRESS IN A.CONT.
                                ;*
                                ;*      REGISTER        USE
                                ;*      --------        ---
                                ;*
                                ;*      R2              ADDDRESS OF RK06 REGISTERS
                                ;*      R5              ADDRESS OF PARAMETER BLOCK
                                ;*
                                ;*CALL  JSR     R0,I.ISSU
                                ;*      <ADDRESS OF ERROR RETURN>
                                ;*      RETURN
                                ;*
                                ;*      ROUTINES USED:
                                ;*      -------------
                                ;*
                                ;*              I.CCLR
                                ;*              I.STOR
                                ;*
                                ;;*****************************************************************

026102  013746  023304          I.ISSU: MOV     T.CS1,-(SP)     ;STORE COMMAND ISSUED
026106  005037  023306                  CLR     T.CS2           ;CLEAR TEMPORARY CS2
026112  116537  000000  023306          MOVB    P.DRVN(R5),T.CS2 ;LOAD IN DRIVE NUMBER
026120  013762  023306  000010          MOV     T.CS2,RKCS2(R2) ;LOAD DRIVE NUMBER FOR COMMAND
026126  116537  000007  023305          MOVB    P.CS1H(R5),T.CS1+1 ;STORE BITS 8-15 OF CS1
026134  142737  177753  023305          BICB    #^C<B.CDT!B.CFMT>,T.CS1+1 ;CLEAR ALL BITS EXCEPT
                                                                ;  FORMAT AND DRIVE TYPE
026142  013762  023304  000000          MOV     T.CS1,RKCS1(R2) ;ISSUE COMMAND
026150  105762  000000          1$:     TSTB    RKCS1(R2)       ;WAIT FOR READY
026154  100375                          BPL     1$
026156  004737  026324                  JSR     PC,I.STOR       ;GO STORE REGISTERS
026162  032737  100000  023304          BIT     #CERR,T.CS1     ;CHECK IF CONTROLLER ERROR OCCUERED
026170  001437                          BEQ     5$              ;NO, RETURN
026172  032737  001000  023306          BIT     #MDS,T.CS2      ;CHECK IF MULTIPLE DRIVE SELECT
026200  001406                          BEQ     2$              ;NO, CHECK FOR OTHER CONTROLLER ERRORS
026202  052737  100000  023356          BIS     #E.MDS,E.CONT   ;SET MULTIPLE DRIVE SELECT FLAG
026210  004737  027030                  JSR     PC,R.CONT       ;REPORT CONTROLLER ERROR
026214  000440                          BR      10$             ;RETURN

                                        ;CHECK IF ANY CONTROLLER ERROR IS SET
026216  032737  024000  023304  2$:     BIT     #CTO!SPAR,T.CS1
026224  001027                          BNE     7$
026226  032737  176400  023306          BIT     #UFE!PGE!NEM!NED!UPE!WCE!DLT,T.CS2
026234  001023                          BNE     7$
026236  032737  131761  023322          BIT     #ILC!DTYE!FMTE!ECH!BSE!HVRC!COE!DTE!OPI!DCK,T.ER
026244  001017                          BNE     7$

026246  122716  000005                  CMPB    #DR.CLR,(SP)    ;CHECK IF CLEAR DRIVE ,
```

```
6458  026252  001003                        BNE     3$                ;NO, DO NOT SET DRIVE HARD ERROR
6459  026254  052765  000020  000014        BIS     #DRVHRD,P.PRST(R5) ;SET HARD DRIVE ERROR
6460  026262  004037  026020          3$:   JSR     R0,I.CCLR         ;GO ISSUE A CONTROLLER CLEAR
6461  026266  026316                        10$                       ;ERROR RETURN
6462  026270  012762  000100  000000  5$:   MOV     #IE,RKCS1(R2)     ;SET INTERRUPT ENABLE
6463  026276  005726                        TST     (SP)+             ;ADJUST STACK
6464  026300  005720                        TST     (R0)+             ;ADJUST R0 FOR NORMAL RETURN
6465  026302  000200                        RTS     R0                ;RETURN
6466
6467  026304  052737  001000  023356  7$:   BIS     #E.CERR,E.CONT    ;SET CONTROLLER ERROR DURING
6468                                                                  ;  DRIVER SERVICING
6469  026312  004737  027030                JSR     PC,R.CONT         ;REPORT ERROR
6470  026316  005726                  10$:  TST     (SP)+             ;ADJUST STACK
6471  026320  011000                        MOV     (R0),R0           ;ADJUST R0 FOR ERROR RETURN
6472  026322  000200                        RTS     R0                ;RETURN
```

```
                                .SBTTL   *STORE RK611 UNIBUS REGISTERS

                                ;;*****************************************************************
                                ;*
                                ;*      THIS SUBROUTINE IS CALLED BY THE RK06 DRIVER TO STORE ALL
                                ;*      RK611 REGISTER IN TEMPORARY LOCATIONS.
                                ;*
                                ;*CALL  JSR     PC,I.STOR
                                ;*      RETURN
                                ;*
                                ;*      REGISTER        USE
                                ;*      --------        ---
                                ;*
                                ;*      R2              ADDRESS OF RK611 REGISTERS
                                ;*
                                ;;*****************************************************************

026324  016237  000000  023304  I.STOR: MOV     RKCS1(R2),T.CS1 ;STORE ALL CONTROLLER REGISTERS
026332  016237  000010  023306          MOV     RKCS2(R2),T.CS2 ;  EXCEPT DATA BUFFER
026340  016237  000002  023310          MOV     RKWC(R2),T.WCR
026346  016237  000004  023312          MOV     RKBA(R2),T.BA
026354  016237  000006  023314          MOV     RKDA(R2),T.DA
026362  016237  000012  023324          MOV     RKDS(R2),T.DS
026370  016237  000014  023322          MOV     RKER(R2),T.ER
026376  016237  000016  023320          MOV     RKASOF(R2),T.ASOF
026404  016237  000020  023316          MOV     RKDCYL(R2),T.DC
026412  016237  000026  023326          MOV     RKMR1(R2),T.MR1
026420  016237  000034  023330          MOV     RKMR2(R2),T.MR2
026426  016237  000036  023332          MOV     RKMR3(R2),T.MR3
026434  016237  000030  023334          MOV     RKECPS(R2),T.POS
026442  016237  000032  023336          MOV     RKECPT(R2),T.PAT
026450  000207                          RTS     PC              ;RETURN
```

```
                                .SBTTL   *STORE CONTROLLER STATUS

                                ;;*****************************************************************
                                ;*
                                ;*      THIS SUBROUTINE IS CALLED BY THE RK06 DRIVER AT PRIORITY 7.
                                ;*      THE FOLLOWING REGISTERS WILL BE STORED:
                                ;*
                                ;*              COMMAND AND STATUS REGISTER 2
                                ;*              WORD COUNT REGISTER
                                ;*              BUS ADDRESS REGISTER
                                ;*              DESIRED TRACK AND SECTOR
                                ;*              STATUS REGISTER
                                ;*              ERROR REGISTER
                                ;*              ATTENTION SUMMARY/OFFSET REGISTER
                                ;*              CYLINDER ADDRESS REGISTER
                                ;*
                                ;*CALL  JSR     PC,I.CSTS
                                ;*      RETURN
                                ;*
                                ;*      THIS ROUTINE ASSUMES THE FOLLOWING REGISTERS CONTAIN:
                                ;*
                                ;*
                                ;*              REGISTER                CONTENTS
                                ;*              --------                --------
                                ;*
                                ;*              R2                      RK06 BASE ADDRESS
                                ;*              R5                      ADDRESS OF PARAMETER BLOCK
                                ;*
                                ;;*****************************************************************

026452  042765  177741  000016  I.CSTS: BIC     #177741,P.CS1(R5)       ;CLEAR ALL BITS EXCEPT FUNCTION
                                                                        ;OF LAST COMMAND ISSUED
026460  042737  000036  023304          BIC     #36,T.CS1               ;CLEAR FUNCTION OF CS1 STATUS
026466  053765  023304  000016          BIS     T.CS1,P.CS1(R5)         ;GENERATE CS1 STATUS INFORMATION
026474  016265  000010  000020  I.CST1: MOV     RKCS2(R2),P.CS2(R5)     ;STORE COMMAND AND STATUS REGISTER 2
026502  016265  000002  000022          MOV     RKWC(R2),P.WCR(R5)      ;STORE WORD COUNT REGISTER
026510  016265  000004  000024          MOV     RKBA(R2),P.BAR(R5)      ;STORE BUS ADDRESS REGISTER
026516  016265  000006  000026          MOV     RKDA(R2),P.DTS(R5)      ;STORE DESIRED TRACK AND SECTOR
026524  016265  000012  000036          MOV     RKDS(R2),P.DS(R5)       ;STORE DRIVE STATUS REGISTER
026532  016265  000014  000034          MOV     RKER(R2),P.ER(R5)       ;STORE ERROR REGISTER
026540  016265  000016  000032          MOV     RKASOF(R2),P.ASOF(R5)   ;STORE ATTENTION SUMMARY AND
                                                                        ;  OFFSET
026546  016265  000020  000030          MOV     RKDCYL(R2),P.DCYL(R5)   ;STORE CYLINDER ADDRESS
026554  000207                  RTS     PC                              ;RETURN
```

```
                                .SBTTL   *GATHER DRIVE STATUS

                                ;;*****************************************************************
                                ;*
                                ;*      THIS SUBROUTINE WILL BE USED TO GATHER DRIVE STATUS
                                ;*      BYTE 01, 10, AND 11. IT IS ASSUMED THAT THE DRIVE
                                ;*      HAS PREVIOUSLY BEEN SEIZED.  IT RUNS AT PRIORITY 7.
                                ;*
                                ;*CALL  JSR     R0,I.STAT
                                ;*      <ADDRESS OF ERROR RETURN>
                                ;*      RETURN
                                ;*
                                ;*      THIS ROUTINE ASSUMES THE FOLLOWING REGISTERS CONTAIN:
                                ;*
                                ;*              REGISTER                CONTENTS
                                ;*              --------                --------
                                ;*
                                ;*              R2                      RK06 BASE ADDRESS
                                ;*              R5                      ADDRESS OF PARAMETER BLOCK
                                ;*
                                ;*      ROUTINES USED:
                                ;*              I.ISSU
                                ;*
                                ;;*****************************************************************

026556  012762  000001  000026  I.STAT: MOV     #1,RKMR1(R2)            ;LOAD MAINTENANCE REGISTER 1
                                                                        ;  FOR STATUS BYTE 01
026564  112737  000001  023304          MOVB    #DR.SEL,T.CS1           ;LOAD COMMAND
026572  004037  026102                  JSR     R0,I.ISSU               ;GET STATUS BYTES 01
026576  026766                          3$                              ;ERROR RETURN
026600  013765  023330  000044          MOV     T.MR2,P.A01(R5)         ;STORE STATUS BYTE 01 MESS A
026606  013765  023332  000046          MOV     T.MR3,P.B01(R5)         ;STORE STATUS BYTE 01 MESS B
026614  012762  000002  000026          MOV     #2,RKMR1(R2)            ;LOAD MAINTENANCE REGISTER 1
                                                                        ;  FOR STATUS BYTE 10
026622  112737  000001  023304          MOVB    #DR.SEL,T.CS1           ;LOAD COMMAND
026630  004037  026102                  JSR     R0,I.ISSU               ;GET STATUS BYTES 10
026634  026766                          3$                              ;ERROR RETURN
026636  013765  023330  000050          MOV     T.MR2,P.A10(R5)         ;STORE STATUS BYTE 10 MESS A
026644  013765  023332  000052          MOV     T.MR3,P.B10(R5)         ;STORE STATUS BYTE 10 MESS B
026652  012762  000003  000026          MOV     #3,RKMR1(R2)            ;LOAD MAINTENANCE REGISTER
                                                                        ;  FOR STATUS BYTE 11
026660  112737  000001  023304          MOVB    #DR.SEL,T.CS1           ;LOAD COMMAND
026666  004037  026102                  JSR     R0,I.ISSU               ;GET STATUS BYTES 11
026672  026766                          3$                              ;ERROR RETURN
026674  013765  023330  000054          MOV     T.MR2,P.A11(R5)         ;STORE STATUS BYTE 11 MESS A
026702  013765  023332  000056          MOV     T.MR3,P.B11(R5)         ;STORE STATUS BYTE 11 MESS B
026710  005062  000026                  CLR     RKMR1(R2)               ;LOAD MAINTENANCE REGISTER 1
                                                                        ;  FOR STATUS BYTE 00
026714  112737  000001  023304          MOVB    #DR.SEL,T.CS1           ;LOAD COMMAND
026722  004037  026102                  JSR     R0,I.ISSU               ;GET STATUS BYTES 00
026726  026766                          3$                              ;ERROR RETURN
026730  013765  023330  000040          MOV     T.MR2,P.A00(R5)         ;STORE STATUS BYTE 00 MESS A
026736  013765  023332  000042          MOV     T.MR3,P.B00(R5)         ;STORE STATUS BYTE 00 MESS B
026744  032737  001000  023332          BIT     #S.PAR,T.MR3            ;CHECK IF BAD PARITY DETECTED BY DRIVE
026752  001407                          BEQ     5$                      ;NO, RETURN NORMALLY
```

```
 6606  026754  052737  002000  023356          BIS     #E.DPAR,E.CONT          ;INDICATE BAD PARITY DETECTED BY DRIVE
 6607  026762  004737  027030                  JSR     PC,R.CONT               ;REPORT ERROR
 6608  026766  011000                  3$:     MOV     (R0),R0                 ;LOAD R0 FOR ERROR RETURN
 6609  026770  000200                          RTS     R0                      ;RETURN
 6610
 6611  026772  052765  001000  000014  5$:     BIS     #PBSVAL,P.PRST(R5)      ;SET PARAMETER BLOCK STATUS VALID
 6612  027000  005720                          TST     (R0)+                   ;ADJUST R0 FOR NORMAL RETURN
 6613  027002  000200                          RTS     R0                      ;RETURN
 6614
```

```
                                        .SBTTL   *COMMON DRIVER RETURNS

027004  105037  023405        R.ABNL: CLRB    INTMSK          ;INHIBIT FUTURE DRIVE INTERRUPT REPORTING
027010  004777  174336                JSR     PC,@A.ABNL      ;INDICATE ABNORMAL RETURN
027014  000207                        RTS     PC              ;RETURN

027016  105037  023405        R.NORM: CLRB    INTMSK          ;INHIBIT FUTURE DRIVE INTERRUPT REPORTING
027022  004777  174322                JSR     PC,@A.NORM      ;INDICATE NORMAL RETURN
027026  000207                        RTS     PC              ;RETURN

027030  105037  023405        R.CONT: CLRB    INTMSK          ;INHIBIT FUTURE DRIVE INTERRUPT REPORTING
027034  105037  023404                CLRB    W.TIME          ;RESET WATCH DOG TIMING ON THIS DRIVE
027040  005037  023420                CLR     W.DRV           ;CLEAR TIMING COUNT FOR THIS DRIVE
027044  004777  174304                JSR     PC,@A.CONT      ;INDICATE CONTROLLER ERROR RETURN
027050  000207                        RTS     PC              ;RETURN
```

```
                        .SBTTL   *COMMAND INITATOR

                        ;;*****************************************************************
                        ;*
                        ;*      THIS SUBROUTINE WILL INITIATE ALL COMMANDS AS SPECIFIED
                        ;*      BY THE COMMAND FIELD OF THE PARAMETER BLOCK. THE FOLLOWING
                        ;*      SPECIAL COMMAND ARE ALSO EXECUTED:
                        ;*
                        ;*              RELEASE
                        ;*              CONROLLER CLEAR
                        ;*              SUBSYSTEM CLEAR
                        ;*              READ ALL DRIVE STATUS
                        ;*              READ SPECIFIED HEADER
                        ;*
                        ;*      THE ABOVE COMMANDS ARE TRANSLATED INTO A SEQUENCE OF COMMANDS
                        ;*
                        ;*CALL  JSR     PC,C.INIT
                        ;*      (ADDRESS OF PARAMETER BLOCK)
                        ;*      RETURN
                        ;*
                        ;*      FOR THE SEQUENTIAL OPERATIONS, THE DRIVER WILL LOAD THE
                        ;*      LOCATIONS, PBLKT AND INTMSK.
                        ;*
                        ;*      ROUTINES USED:
                        ;*              W.WTCH
                        ;*              I.CSTS
                        ;*              I.STAT
                        ;*              I.CCLR
                        ;*
                        ;;*****************************************************************

027052  010546          C.INIT: MOV     R5,-(SP)        ;STORE R5 ON STACK
027054  010446                  MOV     R4,-(SP)        ;STORE R4 ON STACK
027056  010346                  MOV     R3,-(SP)        ;STORE R3 ON STACK
027060  010246                  MOV     R2,-(SP)        ;STORE R2 ON STACK
027062  010146                  MOV     R1,-(SP)        ;STORE R1 ON STACK
027064  010046                  MOV     R0,-(SP)        ;STORE R0 ON STACK
027066  013746  177776          MOV     PS,-(SP)        ;STORE PSW ON STACK
027072  013737  023346  177776  MOV     RKPRI,PS        ;LOCK OUT RK06 INTERRUPTS
027100  017605  000016          MOV     @16(SP),R5      ;STORE PARAMETER BLOCK ADDRESS
027104  062766  000002  000016  ADD     #2,16(SP)       ;ADJUST RETURN
027112  016504  000000          MOV     P.DRVN(R5),R4   ;STORE DRIVE NUMBER
027116  042704  177770          BIC     #^C<DRVMSK>,R4  ;MASK OUT JUNK
027122  010537  023416          MOV     R5,PBLKT        ;LOAD PARAMETER BLOCK TABLE
027126  116437  023406  023405  MOVB    I.DRV(R4),INTMSK ;LOAD INTERRUPT MASK
027134  116437  023406  023404  MOVB    I.DRV(R4),W.TIME ;SET WATCH-DOG TIMER FLAG
027142  013737  023366  023420  MOV     W.SEC,W.DRV     ;LOAD WATCH-DOG TIME

027150  013702  023342          MOV     RKBAS,R2        ;LOAD R2 WITH RK06 ADDRESS BASE

                        ;       RESET ALL BITS IN PROGRAM DEVICE STATUS REGISTER EXCEPT
                        ;               DRIVE IN USE
                        ;               WRITE FOR WRITE CHECK
                        ;               NO CHECK
                        ;               DROP DRIVE FROM TEST SEQUENCE
                        ;               INHIBIT BUS ADDRESS INCREMENT
```

```
6686  027154  042765  075176  000014          BIC     #1C<DRVUSE!W.WCK!NOCHK!DRPDRV!DTBAII>,P.PRST(R5)
6687
6688  027162  010500                          MOV     R5,R0           ;STORE PARAMETER BLOCK ADDRESS
6689  027164  062700  000016                  ADD     #P.CS1,R0       ;CALCULATE FIRST LOCATION TO BE CLEARED
6690  027170  010501                          MOV     R5,R1           ;STORE PARAMETER BLOCK ADDRESS
6691  027172  062701  000062                  ADD     #P.EPAT,R1      ;CALCULATE LAST LOCATION TO BE CLEARED
6692
6693  027176  005020                  1$:     CLR     (R0)+           ;CLEAR RETURN PARAMETER
6694  027200  020001                          CMP     R0,R1           ;CHECK IF FINISHED
6695  027202  101775                          BLOS    1$              ;NO, CLEAR NEXT RETURN PARAMETER
6696  027204  105037  023400                  CLRB    I.ISRL          ;CLEAR RELEASE OR INTERRUPT ISSUED
6697  027210  010465  000020                  MOV     R4,P.CS2(R5)    ;STORE DRIVE NUMBER
6698  027214  005062  000026                  CLR     RKMR1(R2)       ;CLEAR RK06 MAINTENANCE REGISTER 1
6699  027220  132765  000040  000001          BITB    #BIT5,P.CMND(R5) ;CHECK IF SPECIAL COMMAND
6700  027226  001402                          BEQ     3$              ;NO, PROCESS
6701  027230  000137  027744                  JMP     C.SPEC          ;JUMP TO SPECIAL COMMAND PROCESSOR
6702
6703  027234  122765  000107  000001  3$:     CMPB    #UNLOAD,P.CMND(R5) ;CHECK IF POSITIONING COMMAND
6704                                                                  ;  START SPINDLE
6705                                                                  ;  RECALIBRATE
6706                                                                  ;  OFFSET
6707                                                                  ;  SEEK
6708                                                                  ;  UNLOAD
6709
6710  027242  101174                          BHI     25$             ;NO, DRIVE COMMAND
6711                                                                  ;  SELECT DRIVE
6712                                                                  ;  PACK ACKNOWLEDGE
6713                                                                  ;  CLEAR
6714
6715  027244  122765  000117  000001          CMPB    #SEEK,P.CMND(R5) ;CHECK IF DATA TRANSFER
6716  027252  103540                          BLO     20$             ;YES, DATA TRANSFER COMMAND
6717                                                                  ;  READ DATA
6718                                                                  ;  WRITE DATA
6719                                                                  ;  READ HEADER
6720                                                                  ;  WRITE HEADER
6721                                                                  ;  WRITE CHECK
6722  027254  016562  000020  000010          MOV     P.CS2(R5),RKCS2(R2) ;LOAD DRIVE NUMBER
6723  027262  052765  000002  000014          BIS     #DRVPOS,P.PRST(R5) ;SET DRIVE POSITIONING
6724  027270  005037  023360                  CLR     O.WAIT          ;CLEAR WAIT FOR COMMAND
6725  027274  122765  000117  000001          CMPB    #SEEK,P.CMND(R5) ;CHECK IF SEEK
6726  027302  001007                          BNE     5$              ;NO, CHECK FOR OFFSET
6727  027304  016562  000002  000020          MOV     P.CYLN(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS
6728  027312  016562  000004  000006          MOV     P.SECT(R5),RKDA(R2) ;LOAD SECTOR AND TRACK
6729  027320  000431                          BR      8$              ;GO ISSUE COMMAND
6730
6731  027322  122765  000115  000001  5$:     CMPB    #OFFSET,P.CMND(R5) ;CHECK IF OFFSET
6732  027330  001007                          BNE     6$              ;NO, CHECK FOR UNLOAD
6733  027332  116565  000006  000032          MOVB    P.OFST(R5),P.ASOF(R5) ;STORE OFFSET
6734  027340  016562  000032  000016          MOV     P.ASOF(R5),RKASOF(R2) ;LOAD OFFSET REGISTER
6735  027346  000416                          BR      8$              ;GO ISSUE COMMAND
6736
6737  027350  122765  000111  000001  6$:     CMPB    #SRTSPL,P.CMND(R5) ;CHECK IF START SPINDLE
6738  027356  001003                          BNE     7$              ;NO, CHECK IF RECAL
6739  027360  013737  023372  023420          MOV     W.MIN,W.DRV     ;LOAD WATCH DOG TIME FOR 1 MINUTE
6740  027366  122765  000113  000001  7$:     CMPB    #RECAL,P.CMND(R5) ;CHECK IF RECAL
6741  027374  001003                          BNE     8$              ;NO, CONTINUE
```

```
 6742  027376  013737  023370  023420            MOV     W.BSEC,W.DRV    ;LOAD RECAL TIME FOR 8 SECONDS
 6743  027404  116565  000007  000017  8$:       MOVB    P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
 6744  027412  042765  165777  000016            BIC     #^C<CFMT!CDT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT FORMAT
 6745                                                      ;  AND DRIVE TYPE
 6746  027420  116565  000001  000016            MOVB    P.CMND(R5),P.CS1(R5) ;MOVE COMMAND INTO CS1
 6747  027426  042765  000200  000014            BIC     #W.WCK,P.PRST(R5) ;RESET WRITE FOR WRITE CHECK
 6748  027434  032765  000400  000014            BIT     #NOCHK,P.PRST(R5) ;CHECK IN NO CHECK MODE
 6749  027442  001533                            BEQ     30$             ;NO, SKIP CLEAR OF INTERRUPT ENABLE
 6750  027444  042765  000100  000016            BIC     #IE,P.CS1(R5)   ;CLEAR INTERRUPT ENABLE
 6751  027452  016562  000016  000000            MOV     P.CS1(R5),RKCS1(R2) ;ISSUE COMMAND
 6752  027460  004737  023422            10$:    JSR     PC,W.WTCH       ;CALL WATCH DOG TIMER
 6753  027464  016237  000000  023304            MOV     RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REGISTER 1
 6754  027472  032737  000200  023304            BIT     #RDY,T.CS1      ;WAIT FOR READY
 6755  027500  001767                            BEQ     10$
 6756  027502  032737  100000  023304            BIT     #CERR,T.CS1     ;CHECK FOR ERROR
 6757  027510  001011                            BNE     15$             ;YES, GIVE NORMAL RETURN
 6758  027512  004737  023422            11$:    JSR     PC,W.WTCH       ;CALL WATCH DOG TIMER
 6759  027516  016237  000016  023320            MOV     RKASOF(R2),T.ASOF ;STORE ATTENTION SUMMARY
 6760  027524  133737  023405  023321            BITB    INT.15K,T.ASOF+1 ;CHECK IF INTERRUPT HAS OCCURRED
 6761  027532  001767                            BEQ     11$             ;WAIT FOR DRIVE INTERRUPT
 6762  027534  105037  023404            15$:    CLRB    W.TIME          ;RESET TIMING ON THIS DRIVE
 6763  027540  005037  023420                    CLR     W.DRV           ;CLEAR DRIVE TIMING COUNT
 6764  027544  004737  027016                    JSR     PC,R.NORM       ;INDICATE COMMAND IS FINISHED
 6765  027550  000137  030724                    JMP     C.RTRN          ;RESTORE REGISTERS
 6766
 6767  027554  016562  000010  000004  20$:     MOV     P.BALO(R5),RKBA(R2) ;LOAD BUS ADDRESS REGISTER
 6768  027562  016562  000012  000002            MOV     P.WC(R5),RKWC(R2) ;LOAD WORD COUNT REGISTER
 6769  027570  016562  000002  000020            MOV     P.CYLN(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS REGISTER
 6770  027576  016562  000004  000006            MOV     P.SECT(R5),RKDA(R2) ;LOAD SECTOR AND TRACK NUMBER
 6771  027604  122765  000131  000001            CMPB    #WRTCHK,P.CMND(R5)  ;CHECK IF WRITE CHECK COMMAND
 6772  027612  001010                            BNE     25$             ;NO, GO ISSUE THE COMMAND
 6773  027614  032765  000200  000014            BIT     #W.WCK,P.PRST(R5)  ;CHECK IF WRITE COMMAND SHOULD BE ISSUED
 6774  027622  001404                            BEQ     25$             ;NO, GO ISSUE THE COMMAND
 6775  027624  012765  000123  000016            MOV     #WRDATA,P.CS1(R5)  ;ISSUE WRITE COMMAND
 6776  027632  000406                            BR      26$             ;GO ISSUE COMMAND
 6777
 6778  027634  116565  000001  000016  25$:     MOVB    P.CMND(R5),P.CS1(R5) ;MOVE COMMAND INTO CS1
 6779  027642  042765  000200  000014            BIC     #W.WCK,P.PRST(R5) ;RESET WRITE FOR WRITE CHECK
 6780  027650  116565  000007  000017  26$:     MOVB    P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
 6781  027656  142765  177750  000017            BICB    #^C<B.CFMT!B.CDT!B.BA16!B.BA17>,P.CS1+1(R5) ;CLEAR ALL BITS EXCEPT
 6782                                                      ;  FORMAT, DRIVE TYPE, AND BUS ADDRESS
 6783                                                      ;  BITS 16-17
 6784  027664  010537  023360                    MOV     R5,@.WAIT       ;LOAD WAITING FOR COMMAND
 6785  027670  032765  100000  000014            BIT     #DTBAII,P.PRST(R5) ;CHECK IF INHIBIT BUS ADDRESS INCREMENT
 6786  027676  001403                            BEQ     27$             ;NO, LOAD CS2
 6787  027700  052765  000020  000020            BIS     #BAI,P.CS2(R5)  ;SET INHIBIT BUS ADDRESS INCREMENT
 6788  027706  016562  000020  000010  27$:     MOV     P.CS2(R5),RKCS2(R2) ;LOAD CS2
 6789  027714  032765  000400  000014            BIT     #NOCHK,P.PRST(R5) ;CHECK IN NO CHECK MODE
 6790  027722  001403                            BEQ     30$             ;NO, SKIP CLEAR OF INTERRUPT ENABLE
 6791  027724  042765  000100  000016            BIC     #IE,P.CS1(R5)   ;CLEAR INTERRUPT ENABLE
 6792  027732  016562  000016  000000  30$:     MOV     P.CS1(R5),RKCS1(R2) ;ISSUE COMMAND
 6793  027740  000137  030724                    JMP     C.RTRN          ;RESTORE REGISTERS
 6794
 6795                                    .SBTTL  *SPECIAL COMMAND PROCESSING
 6796
 6797  027744  122765  000141  000001  C.SPEC: CMPB    #RDSTAT,P.CMND(R5) ;CHECK IF READ DRIVE STATUS
```

```
6798  027752  001132                          BNE     10$             ;NO, PROCESS OTHER COMMANDS
6799  027754  016562  000020  000010          MOV     P.CS2(R5),RKCS2(R2) ;LOAD CS2 FOR COMMAND
6800  027762  116565  000007  000017          MOVB    P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
6801  027770  042765  165777  000016          BIC     #^C<CFMT!CDT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT FORMAT
6802                                                                  ;  AND DRIVE TYPE
6803  027776  112765  000001  000016          MOVB    #DR.SEL,P.CS1(R5) ;STORE COMMAND
6804  030004  016562  000016  000000          MOV     P.CS1(R5),RKCS1(R2) ;ISSUE COMMAND
6805  030012  004737  023422          2$:     JSR     PC,W.WTCH       ;CALL WATCH-DOG TIMER
6806  030016  016265  000000  000016          MOV     RKCS1(R2),P.CS1(R5) ;STORE COMMAND AND STATUS REG. 1
6807  030024  032765  000200  000016          BIT     #RDY,P.CS1(R5)  ;WAIT FOR READY
6808  030032  001767                          BEQ     2$
6809  030034  004737  026474                  JSR     PC,I.CST1       ;STORE CONTROLLER REGISTERS
6810  030040  016265  000034  000040          MOV     RKMR2(R2),P.AOD(R5) ;STORE STATUS BYTE OD MESSAGE A
6811  030046  016265  000036  000042          MOV     RKMR3(R2),P.BOD(R5) ;STORE STATUS BYTE OD MESSAGE B
6812  030054  032765  100000  000016          BIT     #CERR,P.CS1(R5) ;CHECK IF CONTROLLER ERROR
6813  030062  001436                          BEQ     6$              ;NO, GATHER DRIVE STATUS
6814  030064  105037  023404                  CLRB    W.TIME          ;RESET WATCH DOG TIMING ON THIS DRIVE
6815  030070  005037  023420                  CLR     W.DRV           ;CLEAR WATCH DOG COUNT
6816  030074  032765  000400  000014          BIT     #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
6817  030102  001043                          BNE     8$              ;YES, INDICATE NORMAL RETURN
6818  030104  032765  001000  000020          BIT     #MDS,P.CS2(R5)  ;CHECK IF MULTIPLE DRIVE SELECT
6819  030112  001043                          BNE     9$              ;YES, INDICATE CONTROLLER ERROR
6820  030114  004037  026020                  JSR     R0,I.CCLR       ;CLEAR ERROR
6821  030120  030724                          C.RTRN                  ;ERROR RETURN
6822  030122  032765  010400  000020          BIT     #NED!UFE,P.CS2(R5) ;CHECK IF NON-EXISTENT DRIVE OR UNIT FIELD ERROR
6823  030130  001007                          BNE     5$              ;REPORT ERROR
6824  030132  032765  000001  000036          BIT     #DRA,P.DS(R5)   ;CHECK IF DRIVE AVAILIABLE
6825  030140  001003                          BNE     5$              ;YES, REPORT ERROR
6826  030142  052765  010000  000014          BIS     #DRVSZD,P.PRST(R5) ;INDICATE DRIVE IS SEIZED BY OTHER PORT
6827  030150  004737  027004          5$:     JSR     PC,R.ABNL       ;INDICATE ABNORMAL RETURN
6828  030154  000137  030724                  JMP     C.RTRN          ;RESTORE REGISTERS
6829
6830  030160  004037  026556          6$:     JSR     R0,I.STAT       ;GATHER DRIVE STATUS
6831  030164  030724                          C.RTRN                  ;ERROR RETURN
6832  030166  105037  023404                  CLRB    W.TIME          ;STOP WATCH-DOG TIMING ON DRIVE
6833  030172  005037  023420                  CLR     W.DRV           ;RESET WATCH-DOG TIME
6834  030176  032765  000400  000014          BIT     #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
6835  030204  001402                          BEQ     8$              ;NO, REPORT ERROR
6836  030206  005062  000000                  CLR     RKCS1(R2)       ;CLEAR INTERRUPT ENABLE
6837  030212  004737  027016          8$:     JSR     PC,R.NORM       ;REPORT COMMAND COMPLETE
6838  030216  000137  030724                  JMP     C.RTRN          ;RESTORE REGISTERS
6839
6840  030222  052737  100000  023356  9$:     BIS     #E.MDS,E.CONT   ;SET MULTIPLE DRIVE SELECT
6841  030230  004737  027030                  JSR     PC,R.CONT       ;INDICATE CONTROLLER ERROR
6842  030234  000137  030724                  JMP     C.RTRN
6843
6844  030240  122765  000140  000001  10$:    CMPB    #RELEAS,P.CMND(R5) ;CHECK IF RELEASE COMMAND
6845  030246  001040                          BNE     13$             ;NO, CHECK IF READ ALL HEADERS
6846  030250  010537  023360                  MOV     R5,O.WAIT       ;STORE PARAMETER BLOCK ADDRESS IN
6847                                                                  ;  WAIT FOR COMMAND
6848  030254  052765  000010  000020          BIS     #RLS,P.CS2(R5)  ;SET RELEASE BIT
6849  030262  016562  000020  000010          MOV     P.CS2(R5),RKCS2(R2) ;LOAD CS2 FOR DESELECT
6850  030270  112737  000001  023400          MOVB    #1,I.ISRL       ;SET FLAG FOR RELEASE COMMAND
6851  030276  116565  000007  000017          MOVB    P.CS1H(R5),P.CS1+1(R5);STORE BITS 8-15 OF CS1
6852  030304  042765  165777  000016          BIC     #^C<CFMT!CDT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT FORMAT
6853                                                                  ;  AND DRIVE TYPE
```

```
 6854  030312  112765  000101  000016          MOVB    #SELDRV,P.CS1(R5) ;STORE COMMAND
 6855  030320  032765  000400  000014          BIT     #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
 6856  030326  001403                          BEQ     11$               ;NO, DO NOT RESET INTERRUPT ENABLE
 6857  030330  042765  000100  000016          BIC     #IE,P.CS1(R5)     ;RESET INTERRUPT ENABLE
 6858  030336  016562  000016  000000  11$:    MOV     P.CS1(R5),RKCS1(R2) ;ISSUE COMMAND
 6859  030344  000137  030724                  JMP     C.RTRN            ;RESTORE REGISTERS
 6860
 6861  030350  122765  000164  000001  13$:    CMPB    #RDALHD,P.CMND(R5) ;CHECK IF READ ALL HEADERS
 6862  030356  001053                          BNE     30$               ;NO, CHECK IF CONTROLLER CLEAR
 6863  030360  010537  023360                  MOV     R5,O.WAIT         ;SET WAITING FOR COMMAND COMMPLETION
 6864  030364  016537  000010  023374          MOV     P.BAL0(R5),HDR.AD ;LOAD HEADER ADDRESS
 6865  030372  132765  000020  000007          BITB    #B.CFMT,P.CS1H(R5) ;CHECK IF 22 SECTOR FORMANT
 6866  030400  001404                          BEQ     14$               ;YES, LOAD 22 IN HEADER COUNT
 6867  030402  012737  000024  023376          MOV     #20.,HDR.CT       ;LOAD 20 IN SECTOR COUNT
 6868  030410  000403                          BR      22$               ;GO ISSUE READ HEADER COMMAND
 6869
 6870  030412  012737  000026  023376  14$:    MOV     #22.,HDR.CT       ;LOAD 22 IN SECTOR COUNT
 6871  030420  016562  000002  000020  22$:    MOV     P.CYLN(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS
 6872  030426  016562  000004  000006          MOV     P.SECT(R5),RKDA(R2) ;LOAD TRACK NUMBER
 6873  030434  016562  000020  000010          MOV     P.CS2(R5),RKCS2(R2) ;LOAD DRIVE NUMBER
 6874  030442  116565  000007  000017          MOVB    P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
 6875  030450  042765  165777  000016          BIC     #^C<CFMT!CDT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT DRIVE TYPE
 6876                                                                  ;  AND FORMAT
 6877  030456  112765  000125  000016          MOVB    #RDHEAD,P.CS1(R5) ;STORE READ HEADER COMMAND
 6878  030464  032765  000400  000014          BIT     #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
 6879  030472  001027                          BNE     34$               ;YES, INDICATE ILLEGAL DRIVER COMMAND
 6880  030474  016562  000016  000000          MOV     P.CS1(R5),RKCS1(R2) ;ISSUE READ HEADER
 6881  030502  000137  030724                  JMP     C.RTRN            ;RESTORE REGISTERS
 6882
 6883  030506  122765  000176  000001  30$:    CMPB    #CONCLR,P.CMND(R5) ;CHECK IF CONTROLLER CLEAR
 6884  030514  001012                          BNE     32$               ;NO, CHECK IF SUBSYSTEM CLEAR
 6885  030516  004037  026020                  JSR     R0,I.CCLR         ;CLEAR CONTROLLER
 6886  030522  030724                          C.RTRN                    ;ERROR RETURN
 6887  030524  032765  000400  000014          BIT     #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
 6888  030532  001472                          BEQ     40$               ;NO, INDICATE NORMAL RETURN
 6889  030534  005062  000000                  CLR     RKCS1(R2)         ;RESET INTERRUPT ENABLE
 6890  030540  000467                          BR      40$               ;INDICATE NORMAL RETURN
 6891
 6892  030542  122765  000177  000001  32$:    CMPB    #SUBCLR,P.CMND(R5) ;CHECK IF SUBSYSTEM CLEAR
 6893  030550  001406                          BEQ     36$               ;YES, CLEAR SUBSYSTEM
 6894  030552  052737  000100  023356  34$:    BIS     #E.ILLD,E.CONT    ;SET ILLEGAL DRIVER COMMAND
 6895  030560  004737  027030                  JSR     PC,R.CONT         ;REPORT ERROR
 6896  030564  000457                          BR      C.RTRN            ;RESTORE REGISTERS
 6897
 6898  030566  012762  000040  000010  36$:    MOV     #SCLR,RKCS2(R2)   ;ISSUE SUBSYSTEM CLEAR
 6899  030574  016265  000000  000016          MOV     RKCS1(R2),P.CS1(R5) ;STORE COMMAND AND STATUS REGISTER 1
 6900  030602  032765  100000  000016          BIT     #CERR,P.CS1(R5)   ;CLEAR IF CONTROLLER ERROR RESET
 6901  030610  001406                          BEQ     37$               ;NO, FINISH COMMAND
 6902  030612  052737  000001  023356          BIS     #BIT0,E.CONT      ;SET CLEAR SUBSYSTEM DID NOT CLEAR
 6903                                                                  ;  CONTROLLER ERROR
 6904  030620  004737  027030                  JSR     PC,R.CONT         ;REPORT ERROR
 6905  030624  000437                          BR      C.RTRN            ;RESTORE REGISTERS
 6906
 6907  030626  013746  023364          37$:    MOV     W.MILI,-(SP)      ;LOAD 16 MILI-SECOND COUNT FOR ATTENTION
 6908                                                                  ;  TO DISAPPEAR
 6909  030632  016265  000000  000016  38$:    MOV     RKCS1(R2),P.CS1(R5) ;STORE CS1
```

```
6910  030640  032765  040000  000016          BIT     #DI,P.CS1(R5)   ;CHECK IF ATTENTIONS CLEARED
6911  030646  001411                          BEQ     39$             ;YES, FINISH COMMAND
6912  030650  005316                          DEC     (SP)            ;DECREMENT 16 MILISECOND COUNT
6913  030652  001367                          BNE     38$             ;CHECK DRIVE INTERRUPT AGAIN
6914  030654  005726                          TST     (SP)+           ;ADJUST STACK
6915  030656  052737  000040  023356          BIS     #E.SCLR,E.CONT  ;SET SUBSYSTEM CLEAR DID NOT CLEAR
6916                                                                  ;  DRIVE ATTENTIONS
6917  030664  004737  027030                  JSR     PC,R.CONT       ;REPORT ERROR
6918  030670  000415                          BR      C.RTRN          ;RESTORE REGISTER
6919
6920  030672  005726                  39$:    TST     (SP)+           ;ADJUST STACK
6921  030674  032765  000400  000014          BIT     #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
6922  030702  001010                          BNE     C.RTRN          ;YES, RESTORE REGISTERS
6923  030704  112737  177777  023400          MOVB    #-1,I.ISRL      ;SET INTERRUPT ENABLE SET
6924  030712  012762  000100  000000          MOV     #IE,RKCS1(R2)   ;SET INTERRUPT ENABLE
6925  030720  004737  027016          40$:    JSR     PC,R.NORM       ;INDICATE NORMAL TERMINATION
6926
6927  030724  012637  177776          C.RTRN: MOV     (SP)+,PS        ;RESTORE PSW
6928  030730  012600                          MOV     (SP)+,R0        ;RESTORE R0
6929  030732  012601                          MOV     (SP)+,R1        ;RESTORE R1
6930  030734  012602                          MOV     (SP)+,R2        ;RESTORE R2
6931  030736  012603                          MOV     (SP)+,R3        ;RESTORE R3
6932  030740  012604                          MOV     (SP)+,R4        ;RESTORE R4
6933  030742  012605                          MOV     (SP)+,R5        ;RESTORE R5
6934  030744  000207                          RTS     PC              ;RETURN
6935                                  .SBTTL  OCTAL TO BINARY CONVERSION ROUTINE
6936
6937                                  ;;**************************************************************
6938                                  ;*
6939                                  ;*      THIS ROUTINE WILL CHECK A STRING OF ASCII CHARACTERS TERMINATED
6940                                  ;*      WITH A NULL <000> OR COMMA.  IF THE CHARACTERS ARE LEGAL
6941                                  ;*      IT  WILL GENERATE TWO BINARY WORDS PLACING THE LOW 16 BITS
6942                                  ;*      ON THE STACK AND THE HIGH 16 BITS IN LOCATION $nIOCT.
6943                                  ;*
6944                                  ;*CALL
6945                                  ;*      MOV     <ADDRESS OF ASCII STRING>,-(SP)
6946                                  ;*      JSR     PC,OCTBIN
6947                                  ;*      <ADDRESS OF ERROR RETURN>
6948                                  ;*      RETURN
6949                                  ;*
6950                                  ;;**************************************************************
6951
6952  030746  010046          OCTBIN: MOV     R0,-(SP)        ;SAVE R0
6953  030750  010146                  MOV     R1,-(SP)        ;SAVE R1
6954  030752  010246                  MOV     R2,-(SP)        ;SAVE R2
6955  030754  016600  000010          MOV     10(SP),R0       ;GET ADDRESS OF ASCII STRING
6956  030760  005001                  CLR     R1              ;CLEAR DATA WORDS
6957  030762  005002                  CLR     R2
6958  030764  112046          2$:     MOVB    (R0)+,-(SP)     ;PICK THIS CHARACTER
6959  030766  001423                  BEQ     3$              ;IF ZERO GET OUT
6960  030770  121627  000054          CMPB    (SP),#',        ;CHECK IF COMMA
6961  030774  001420                  BEQ     3$              ;IF COMMA GET OUT
6962  030776  122716  000060          CMPB    #'0,(SP)        ;MAKE SURE THIS CHARACTER IS
6963  031002  003030                  BGT     4$              ;  AN OCTAL DIGIT
6964  031004  122716  000067          CMPB    #'7,(SP)
6965  031010  002425                  BLT     4$
```

```
 6966  031012  006301                        ASL     R1              ; *2
 6967  031014  006102                        ROL     R2
 6968  031016  006301                        ASL     R1              ; *4
 6969  031020  006102                        ROL     R2
 6970  031022  006301                        ASL     R1              ; *8
 6971  031024  006102                        ROL     R2
 6972  031026  042716  177770                BIC     #^C7,(SP)       ;STRIP THE ASCII JUNK
 6973  031032  062601                        ADD     (SP)+,R1        ;ADD THIS DIGIT
 6974  031034  000753                        BR      2$              ;LOOP
 6975  031036  005726              3$:       TST     (SP)+           ;CLEAN PARTIAL FROM STACK
 6976  031040  010166  000010                MOV     R1,10(SP)       ;SAVE RESULT
 6977  031044  010237  031100                MOV     R2,$HIOCT
 6978  031050  012602                        MOV     (SP)+,R2        ;RESTORE R2
 6979  031052  012601                        MOV     (SP)+,R1        ;RESTORE R1
 6980  031054  012600                        MOV     (SP)+,R0        ;RESTORE R0
 6981  031056  062716  000002                ADD     #2,(SP)         ;ADJUST RETURN
 6982  031062  000207                        RTS     PC              ;RETURN
 6983
 6984  031064  005726              4$:       TST     (SP)+           ;CLEAN UP PARTIAL FROM STACK
 6985  031066  012602                        MOV     (SP)+,R2        ;RESTORE R2
 6986  031070  012601                        MOV     (SP)+,R1        ;RESTORE R1
 6987  031072  012600                        MOV     (SP)+,R0        ;RESTORE R0
 6988  031074  013616                        MOV     @(SP)+,(SP)     ;PUT ADDRESS OF ERROR ROUTINE ON STACK
 6989  031076  000207                        RTS     PC              ;GO PROCESS ERROR
 6990  031100  000000              $HIOCT: .WORD   0               ;HIGH ORDER BITS GO HERE
 6991                                .SBTTL  DECIMAL TO BINARY CONVERSION ROUTINE
 6992
 6993                                ;;******************************************************************
 6994                                ;*
 6995                                ;*      THIS ROUTINE WILL CHECK A STRING OF ASCII CHARACTERS TERMINATED
 6996                                ;*      WITH A NULL <000> OR COMMA.  IF THE CHARACTERS ARE LEGAL,
 6997                                ;*      IT WILL GENERATE A  BINARY WORD PLACING IT ON THE STACK.
 6998                                ;*
 6999                                ;*CALL
 7000                                ;*      MOV     <ADDRESS OF ASCII STRING>,-(SP)
 7001                                ;*      JSR     PC,DECBIN
 7002                                ;*      <ADDRESS OF ERROR RETURN>
 7003                                ;*      RETURN
 7004                                ;*
 7005                                ;;******************************************************************
 7006
 7007  031102  010046              DECBIN: MOV     R0,-(SP)        ;SAVE R0
 7008  031104  010146                      MOV     R1,-(SP)        ;SAVE R1
 7009  031106  010246                      MOV     R2,-(SP)        ;SAVE R2
 7010  031110  016600  000010              MOV     10(SP),R0       ;GET ADDRESS OF ASCII STRING
 7011  031114  005046                      CLR     -(SP)           ;CLEAR DATA WORD
 7012  031116  005002                      CLR     R2              ;SIGN SET POSITIVE
 7013  031120  122710  000055              CMPB    #'-,(R0)        ;SEE IF A MINUS SIGN
 7014  031124  001001                      BNE     2$              ;BRANCH IF NO MINUS SIGN
 7015  031126  112002                      MOVB    (R0)+,R2        ;SAVE FOR LATER USE
 7016  031130  112001              2$:     MOVB    (R0)+,R1        ;PICKUP THIS CHARACTER
 7017  031132  001427                      BEQ     3$              ;GET OUT IF ZERO
 7018  031134  120127  000054              CMPB    R1,#',          ;CHECK IF COMMA
 7019  031140  001424                      BEQ     3$              ;GET OUT IF COMMA
 7020  031142  122701  000060              CMPB    #'0,R1          ;MAKE SURE THIS CHARACTER IS
 7021  031146  003034                      BGT     5$              ;  A DIGIT BETWEEN 0 & 9
```

```
7022  031150  122701  000071          CMPB    #'9,R1
7023  031154  002431                  BLT     5$
7024  031156  032716  170000          BIT     #170000,(SP)    ;DON'T LET NUMBER GET TO BIG
7025  031162  001026                  BNE     5$              ;BRANCH IF NUMBER WOULD OVERFLOW
7026  031164  006316                  ASL     (SP)            ; *2
7027  031166  011646                  MOV     (SP),-(SP)      ;SAVE FOR LATER
7028  031170  006316                  ASL     (SP)            ; *4
7029  031172  006316                  ASL     (SP)            ; *8
7030  031174  062616                  ADD     (SP)+,(SP)      ; *10
7031  031176  102420                  BVS     5$              ;OVERFLOW ISN'T ALLOWED
7032  031200  162701  000060          SUB     #'0,R1          ;STRIP AWAY THE ASCII JUNK
7033  031204  060116                  ADD     R1,(SP)         ;ADD IN THIS DIGIT
7034  031206  102414                  BVS     5$              ;OVERFLOW ISN'T ALLOWED
7035  031210  000747                  BR      2$              ;LOOP
7036  031212  005702          3$:     TST     R2              ;CHECK IF NUMBER IS NEGATIVE
7037  031214  001401                  BEQ     4$              ;BRANCH IF NO
7038  031216  005416                  NEG     (SP)            ;YES--NEGATE THE NUMBER
7039  031220  012666  000010  4$:     MOV     (SP)+,10(SP)    ;SAVE RESULT
7040  031224  012602                  MOV     (SP)+,R2        ;RESTORE R2
7041  031226  012601                  MOV     (SP)+,R1        ;RESTORE R1
7042  031230  012600                  MOV     (SP)+,R0        ;RESTORE R0
7043  031232  062716  000002          ADD     #2,(SP)         ;ADJUST RETURN
7044  031236  000207                  RTS     PC              ;RETURN
7045
7046  031240  005726          5$:     TST     (SP)+           ;CLEAN PARTIAL NUMBER FROM STACK
7047  031242  012602                  MOV     (SP)+,R2        ;RESTORE R2
7048  031244  012601                  MOV     (SP)+,R1        ;RESTORE R1
7049  031246  012600                  MOV     (SP)+,R0        ;RESTORE R0
7050  031250  013616                  MOV     @(SP)+,(SP)     ;PUT ADDRESS OF ERROR ON STACK
7051  031252  000207                  RTS     PC              ;GO PROCESS ERROR
7052                          .SBTTL  TYPE ROUTINE
7053
7054                          ;;**********************************************************************
7055                          ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
7056                          ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
7057                          ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
7058                          ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
7059                          ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
7060                          ;*
7061                          ;*CALL:
7062                          ;*1) USING A TRAP INSTRUCTION
7063                          ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
7064                          ;*OR
7065                          ;*      TYPE
7066                          ;*      MESADR
7067                          ;*
7068
7069  031254  105737  001157  $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
7070  031260  100002                  BPL     1$              ;;BR IF YES
7071  031262  000000                  HALT                    ;;HALT HERE IF NO TERMINAL
7072  031264  000407                  BR      3$              ;;LEAVE
7073  031266  010046          1$:     MOV     R0,-(SP)        ;;SAVE R0
7074  031270  017600  000002          MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
7075  031274  112046          2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
7076  031276  001005                  BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
7077  031300  005726                  TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
```

```
7078  031302  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
7079  031304  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
7080  031310  000002                  RTI                             ;;RETURN
7081  031312  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
7082  031316  001430                  BEQ     8$
7083  031320  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
7084  031324  001006                  BNE     5$
7085  031326  005726                  TST     (SP)+                   ;;POP  <CR><LF> EQUIV
7086  031330  104400                  TYPE                            ;;TYPE A CR AND LF
7087  031332  001171                  $CRLF
7088  031334  105037  031470                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
7089  031340  000755                  BR      2$                      ;;GET NEXT CHARACTER
7090  031342  004737  031424          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
7091  031346  123726  001156          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
7092  031352  001350                  BNE     2$                      ;;IF NO GO GET NEXT CHAR.
7093  031354  013746  001154                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
7094                                                                  ;;AND THE NULL CHAR.
7095  031360  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
7096  031364  002770                  BLT     6$                      ;;BR IF NO--GO POP THE NULL OFF OF STACK
7097  031366  004737  031424                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
7098  031372  105337  031470                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
7099  031376  000770                  BR      7$                      ;;LOOP
7100
7101                                  ;HORIZONTAL TAB PROCESSOR
7102
7103  031400  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
7104  031404  004737  031424          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
7105  031410  132737  000007  031470          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
7106  031416  001372                  BNE     9$                      ;;TAB STOP
7107  031420  005726                  TST     (SP)+                   ;;POP SPACE OFF STACK
7108  031422  000724                  BR      2$                      ;;GET NEXT CHARACTER
7109  031424  105777  147520          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
7110  031430  100375                  BPL     $TYPEC
7111  031432  116677  000002  147512          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
7112  031440  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
7113  031446  001003                  BNE     1$                      ;;BRANCH IF NO
7114  031450  105037  031470                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
7115  031454  000406                  BR      $TYPEX                  ;;EXIT
7116  031456  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
7117  031464  001402                  BEQ     $TYPEX                  ;;BRANCH IF YES
7118  031466  105227                  INCB    (PC)+                   ;;COUNT THE CHARACTER
7119  031470  000000          $CHARCNT:.WORD  0                       ;;CHARACTER COUNT STORAGE
7120  031472  000207          $TYPEX: RTS     PC
7121
7122                                  .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
7123
7124                                  ;;**********************************************
7125                                  ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
7126                                  ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
7127                                  ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
7128                                  ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
7129                                  ;*REPLACED WITH SPACES.
7130                                  ;*CALL:
7131                                  ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
7132                                  ;*      TYPDS                   ;;GO TO THE ROUTINE
7133
```

```
7134  031474                                $TYPDS:
7135  031474  010046                              MOV     R0,-(SP)        ;;PUSH R0 ON STACK
7136  031476  010146                              MOV     R1,-(SP)        ;;PUSH R1 ON STACK
7137  031500  010246                              MOV     R2,-(SP)        ;;PUSH R2 ON STACK
7138  031502  010346                              MOV     R3,-(SP)        ;;PUSH R3 ON STACK
7139  031504  010546                              MOV     R5,-(SP)        ;;PUSH R5 ON STACK
7140  031506  012746  020200                      MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
7141  031512  016605  000020                      MOV     20(SP),R5       ;;GET THE INPUT NUMBER
7142  031516  100004                              BPL     1$              ;;BR IF INPUT IS POS.
7143  031520  005405                              NEG     R5              ;;MAKE THE BINARY NUMBER POS.
7144  031522  112766  000055  000001              MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
7145  031530  005000                      1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
7146  031532  012703  031710                      MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
7147  031536  112723  000040                      MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
7148  031542  005002                      2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
7149  031544  016001  031700                      MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
7150  031550  160105                      3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
7151  031552  002402                              BLT     4$              ;;BR IF DONE
7152  031554  005202                              INC     R2              ;;INCREASE THE BCD DIGIT BY 1
7153  031556  000774                              BR      3$
7154  031560  060105                      4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
7155  031562  005702                              TST     R2              ;;CHECK IF BCD DIGIT=0
7156  031564  001002                              BNE     5$              ;;FALL THROUGH IF 0
7157  031566  105716                              TSTB    (SP)            ;;STILL DOING LEADING 0'S?
7158  031570  100407                              BMI     7$              ;;BR IF YES
7159  031572  106316                      5$:     ASLB    (SP)            ;;MSD?
7160  031574  103003                              BCC     6$              ;;BR IF NO
7161  031576  116663  000001  177777              MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
7162  031604  052702  000060                6$:   BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
7163  031610  052702  000040                7$:   BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
7164  031614  110223                              MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
7165  031616  005720                              TST     (R0)+           ;;JUST INCREMENTING
7166  031620  020027  000010                      CMP     R0,#10          ;;CHECK THE TABLE INDEX
7167  031624  002746                              BLT     2$              ;;GO DO THE NEXT DIGIT
7168  031626  003002                              BGT     8$              ;;GO TO EXIT
7169  031630  010502                              MOV     R5,R2           ;;GET THE LSD
7170  031632  000764                              BR      6$              ;;GO CHANGE TO ASCII
7171  031634  105726                      8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
7172  031636  100003                              BPL     9$              ;;BR IF NO
7173  031640  116663  177777  177776              MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
7174  031646  105013                      9$:     CLRB    (R3)            ;;SET THE TERMINATOR
7175  031650  012605                              MOV     (SP)+,R5        ;;POP STACK INTO R5
7176  031652  012603                              MOV     (SP)+,R3        ;;POP STACK INTO R3
7177  031654  012602                              MOV     (SP)+,R2        ;;POP STACK INTO R2
7178  031656  012601                              MOV     (SP)+,R1        ;;POP STACK INTO R1
7179  031660  012600                              MOV     (SP)+,R0        ;;POP STACK INTO R0
7180  031662  104400  031710                      TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
7181  031666  016666  000002  000004              MOV     2(SP),4(SP)     ;;ADJUST THE STACK
7182  031674  012616                              MOV     (SP)+,(SP)
7183  031676  000002                              RTI                     ;;RETURN TO USER
7184  031700  023420                      $DTBL:  10000.
7185  031702  001750                              1000.
7186  031704  000144                              100.
7187  031706  000012                              10.
7188  031710  000004                      $DBLK:  .BLKW   4
7189                                      .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
```

```
7190
7191                                    ;;*****************************************************************
7192                                    ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
7193                                    ;*OCTAL (ASCII) NUMBER AND TYPE IT.
7194                                    ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
7195                                    ;*CALL:
7196                                    ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
7197                                    ;*      TYPOS                   ;;CALL FOR TYPEOUT
7198                                    ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
7199                                    ;*      .BYTE   M               ;;M=1 OR 0
7200                                    ;*                                      ;;1=TYPE LEADING ZEROS
7201                                    ;*                                      ;;0=SUPPRESS LEADING ZEROS
7202                                    ;*
7203                                    ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
7204                                    ;*$TYPOS OR $TYPOC
7205                                    ;*CALL:
7206                                    ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
7207                                    ;*      TYPON                   ;;CALL FOR TYPEOUT
7208                                    ;*
7209                                    ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
7210                                    ;*CALL:
7211                                    ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
7212                                    ;*      TYPOC                   ;;CALL FOR TYPEOUT
7213
7214  031720  017646  000000           $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
7215  031724  116637  000001  032143           MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
7216  031732  112637  032145                   MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
7217  031736  062716  000002                   ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
7218  031742  000406                           BR      $TYPON
7219  031744  112737  000001  032143  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
7220  031752  112737  000006  032145          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
7221  031760  112737  000005  032142  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
7222  031766  010346                           MOV     R3,-(SP)        ;;SAVE R3
7223  031770  010446                           MOV     R4,-(SP)        ;;SAVE R4
7224  031772  010546                           MOV     R5,-(SP)        ;;SAVE R5
7225  031774  113704  032145                   MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
7226  032000  005404                           NEG     R4
7227  032002  062704  000006                   ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
7228  032006  110437  032144                   MOVB    R4,$OMODE       ;;SAVE IT FOR USE
7229  032012  113704  032143                   MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
7230  032016  016605  000012                   MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
7231  032022  005003                           CLR     R3              ;;CLEAR THE OUTPUT WORD
7232  032024  006105                   1$:     ROL     R5              ;;ROTATE MSB INTO "C"
7233  032026  000404                           BR      3$              ;;GO DO MSB
7234  032030  006105                   2$:     ROL     R5              ;;FORM THIS DIGIT
7235  032032  006105                           ROL     R5
7236  032034  006105                           ROL     R5
7237  032036  010503                           MOV     R5,R3
7238  032040  006103                   3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
7239  032042  105337  032144                   DECB    $OMODE          ;;TYPE THIS DIGIT?
7240  032046  100016                           BPL     7$              ;;BR IF NO
7241  032050  042703  177770                   BIC     #177770,R3      ;;GET RID OF JUNK
7242  032054  001002                           BNE     4$              ;;TEST FOR 0
7243  032056  005704                           TST     R4              ;;SUPPRESS THIS 0?
7244  032060  001403                           BEQ     5$              ;;BR IF YES
7245  032062  005204                   4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
```

```
7246  032064  052703  000060            BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
7247  032070  052703  000040      5$:   BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
7248  032074  110337  032140            MOVB    R3,8$           ;;SAVE FOR TYPING
7249  032100  104400  032140            TYPE    ,8$             ;;GO TYPE THIS DIGIT
7250  032104  105337  032142      7$:   DECB    $OCNT           ;;COUNT BY 1
7251  032110  003347                    BGT     2$              ;;BR IF MORE TO DO
7252  032112  002402                    BLT     6$              ;;BR IF DONE
7253  032114  005204                    INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
7254  032116  000744                    BR      2$              ;;GO DO THE LAST DIGIT
7255  032120  012605            6$:     MOV     (SP)+,R5        ;;RESTORE R5
7256  032122  012604                    MOV     (SP)+,R4        ;;RESTORE R4
7257  032124  012603                    MOV     (SP)+,R3        ;;RESTORE R3
7258  032126  016666  000002  000004    MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
7259  032134  012616                    MOV     (SP)+,(SP)
7260  032136  000002                    RTI                     ;;RETURN
7261  032140     000      8$:   .BYTE   0               ;;STORAGE FOR ASCII DIGIT
7262  032141     000            .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
7263  032142     000      $OCNT:.BYTE   0               ;;OCTAL DIGIT COUNTER
7264  032143     000      $OFILL:.BYTE  0               ;;ZERO FILL SWITCH
7265  032144  000000      $OMODE:.WORD  0               ;;NUMBER OF DIGITS TO TYPE
7266                      .SBTTL  TTY INPUT ROUTINE
7267
7268                      ;;*****************************************************************
7269                      .ENABL  LSB
7270  032146  000000      $TKCNT: .WORD   0               ;;NUMBER OF ITEMS IN QUEUE
7271  032150  000000      $TKQIN: .WORD   0               ;;INPUT POINTER
7272  032152  000000      $TKQOUT: .WORD  0               ;;OUTPUT POINTER
7273  032154  000001      $TKQSRT: .BLKB  1               ;;TTY KEYBOARD QUEUE
7274          032155      $TKQEND=.
7275          032156      .EVEN
7276
7277                      ;*TK INITIALIZE ROUTINE
7278                      ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
7279                      ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
7280                      ;
7281                      ;*CALL:
7282                      ;*      JSR     PC,$TKINT
7283                      ;*      RETURN
7284                      ;
7285  032156  005037  032146  $TKINT: CLR     $TKCNT          ;;CLEAR COUNT OF ITEMS IN QUEUE
7286  032162  012737  032154  032150    MOV     #$TKQSRT,$TKQIN ;;MOVE THE STARTING ADDRESS OF THE
7287  032170  013737  032150  032152    MOV     $TKQIN,$TKQOUT  ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
7288  032176  012737  032226  000060    MOV     #$TKSRV,@#TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
7289  032204  012737  000200  000062    MOV     #200,@#TKVEC+2  ;;"BR" LEVEL 4
7290  032212  005777  146730            TST     @$TKB           ;;CLEAR DONE FLAG
7291  032216  012777  000100  146720    MOV     #100,@$TKS      ;;ENABLE TTY KEYBOARD INTERRUPT
7292  032224  000207                    RTS     PC              ;;RETURN TO CALLER
7293
7294                      ;*TK SERVICE ROUTINE
7295                      ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
7296                      ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
7297                      ;*IT IN THE QUEUE.
7298                      ;*IF THE CHARACTER IS A "CONTROL-C" (↑C) $TKINT IS CALLED AND
7299                      ;*UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (U.TSRT)
7300                      ;
7301  032226  117746  146714  $TKSRV: MOVB    @$TKB,-(SP)     ;;PICKUP THE CHARACTER
```

```
7302  032232  042716  177600                  BIC     #^C177,(SP)     ;;STRIP THE JUNK
7303  032236  021627  000003                  CMP     (SP),#3         ;;IS IT A CONTROL C?
7304  032242  001007                          BNE     1$              ;;BRANCH IF NO
7305  032244  104400  033424                  TYPE    ,$CNTLC         ;;TYPE A CONTROL-C (^C)
7306  032250  004737  032156                  JSR     PC,$TKINT       ;;INIT THE KEYBOARD
7307  032254  005726                          TST     (SP)+           ;;CLEAN UP STACK
7308  032256  000137  004116                  JMP     @DTSRT          ;;CONTROL C RESTART
7309  032262  021627  000007          1$:     CMP     (SP),#7         ;;IS IT A CONTROL G?
7310  032266  001004                          BNE     2$              ;;BRANCH IF NO
7311  032270  022737  000176  001140          CMP     #SWREG,SWR      ;;IS SOFT-SWR SELECTED?
7312  032276  001500                          BEQ     6$              ;;GO TO SWR CHANGE
7313
7314  03230C                          2$:
7315  032300  022737  000001  032146          CMP     #1,$TKCNT       ;;IS THE QUEUE FULL?
7316  032306  001004                          BNE     3$              ;;BRANCH IF NO
7317  032310  104400  001164                  TYPE    ,$BELL          ;;RING THE TTY BELL
7318  032314  005726                          TST     (SP)+           ;;CLEAN CHARACTER OFF OF STACK
7319  032316  000451                          BR      5$              ;;EXIT
7320  032320  021627  000023          3$:     CMP     (SP),#23        ;;IS IT A CONTROL-S?
7321  032324  001021                          BNE     32$             ;;BRANCH IF NO
7322  032326  005077  146612                  CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
7323  032332  005726                          TST     (SP)+           ;;CLEAN CHAR OFF STACK
7324  032334  105777  146604          31$:    TSTB    @$TKS           ;;WAIT FOR A CHAR
7325  032340  100375                          BPL     31$             ;;LOOP UNTIL ITS THERE
7326  032342  117746  146600                  MOVB    @$TKB,-(SP)     ;;GET THE CHARACTER
7327  032346  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
7328  032352  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
7329  032356  001366                          BNE     31$             ;;BRANCH IF NO
7330  032360  012777  000100  146556          MOV     #100,@$TKS      ;;REENABLE TTY KEYBOARD INTERRUPTS
7331  032366  000002                          RTI                     ;;RETURN
7332  032370  005237  032146          32$:    INC     $TKCNT          ;;COUNT THIS CHARACTER
7333  032374  021627  000140                  CMP     (SP),#140       ;;IS IT UPPER CASE?
7334  032400  002405                          BLT     4$              ;;BRANCH IF YES
7335  032402  021627  000175                  CMP     (SP),#175       ;;IS IT A SPECIAL CHAR?
7336  032406  003002                          BGT     4$              ;;BRANCH IF YES
7337  032410  042716  000040                  BIC     #40,(SP)        ;;MAKE IT UPPER CASE
7338  032414  112677  177530          4$:     MOVB    (SP)+,@$TKQIN   ;;AND PUT IT IN QUEUE
7339  032420  005237  032150                  INC     $TKQIN          ;;UPDATE THE POINTER
7340  032424  023727  032150  032155          CMP     $TKQIN,#$TKQEND ;;GO OFF THE END?
7341  032432  001003                          BNE     5$              ;;BRANCH IF NO
7342  032434  012737  032154  032150          MOV     #$TKQSRT,$TKQIN ;;RESET THE POINTER
7343  032442  000002                  5$:     RTI                     ;;RETURN
7344
7345                                          ;;**************************************************************
7346                                          ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
7347                                          ;*ROUTINE IS ENTERED  FROM THE TRAP HANDLER, AND WILL
7348                                          ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
7349                                          ;*CALL WHEN OPERATING IN TTY INTERRUPT MODE.
7350  032444  022737  000176  001140  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED
7351  032452  001124                          BNE     15$             ;;EXIT IF NOT
7352  032454  105777  146464                  TSTB    @$TKS           ;;IS A CHAR WAITING?
7353  032460  100121                          BPL     15$             ;;IF NOT, EXIT
7354  032462  117746  146460                  MOVB    @$TKB,-(SP)     ;;YES
7355  032466  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT  7-BIT ASCII
7356  032472  021627  000007                  CMP     (SP),#7         ;;IS IT A CONTROL-G?
7357  032476  001300                          BNE     2$              ;;IF NOT, PUT IT IN THE TTY QUEUE
```

```
7358                                                    ;;AND EXIT
7359
7360                                    ;;**************************************************************
7361                                    ;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
7362                                    ;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
7363                                    ;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
7364  032500  123727  001134  000001  6$:     CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
7365  032506  001674                          BEQ     2$              ;;BRANCH IF YES
7366  032510  005726                          TST     (SP)+           ;;CLEAR CONTROL-G OFF STACK
7367  032512  004737  032156                  JSR     PC,$TKINT       ;;FLUSH THE TTY INPUT QUEUE
7368  032516  005077  146422                  CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
7369  032522  112737  000001  001135          MOVB    #1,$INTAG       ;;SET INTERRUPT MODE INDICATOR
7370
7371  032530  104400  033436                  TYPE    .$CNTLG         ;;ECHO THE CONTROL-G (↑G)
7372  032534  104400  033443  $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
7373  032540  013746  000176                  MOV     $WREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
7374  032544  104401                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
7375  032546  104400  033454                  TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
7376  032552  005046                  19$:     CLR     -(SP)           ;;CLEAR COUNTER
7377  032554  005046                          CLR     -(SP)           ;;THE NEW SWR
7378  032556  105777  146362          7$:     TSTB    @$TKS           ;;CHAR THERE?
7379  032562  100375                          BPL     7$              ;;IF NOT TRY AGAIN
7380
7381  032564  117746  146356                  MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
7382  032570  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
7383
7384  032574  021627  000003                  CMP     (SP),#3         ;;IS IT A CONTROL-C?
7385  032600  001015                          BNE     9$              ;;BRANCH IF NOT
7386  032602  104400  033424                  TYPE    ,$CNTLC         ;;YES, ECHO CONTROL-C (↑C)
7387  032606  062706  000006                  ADD     #6,SP           ;;CLEAN UP STACK
7388  032612  123727  001135  000001          CMPB    $INTAG,#1       ;;REENABLE TTY KEYBOARD INTERRUPTS?
7389  032620  001003                          BNE     8$              ;;BRANCH IF NO
7390  032622  012777  000100  146314          MOV     #100,@$TKS      ;;ALLOW TTY KEYBOARD INTERRUPTS
7391  032630  000137  004116          8$:     JMP     UDTSRT          ;;CONTROL-C RESTART
7392
7393
7394  032634  021627  000025          9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
7395  032640  001005                          BNE     10$             ;;BRANCH IF NOT
7396  032642  104400  033431                  TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (↑U)
7397  032646  062706  000006          20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
7398  032652  000737                          BR      19$             ;;LET'S TRY IT AGAIN
7399
7400
7401  032654  021627  000015          10$:    CMP     (SP),#15        ;;IS IT A <CR>?
7402  032660  001022                          BNE     16$             ;;BRANCH IF NO
7403  032662  005766  000004                  TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
7404  032666  001403                          BEQ     11$             ;;BRANCH IF YES
7405  032670  016677  000002  146242          MOV     2(SP),@SWR      ;;SAVE NEW SWR
7406  032676  062706  000006          11$:    ADD     #6,SP           ;;CLEAR UP STACK
7407  032702  104400  001171          14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
7408  032706  123727  001135  000001          CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
7409  032714  001003                          BNE     15$             ;;BRANCH IF NOT
7410  032716  012777  000100  146220          MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
7411  032724  000002                  15$:    RTI                     ;;RETURN
7412  032726  004737  031424          16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
7413  032732  021627  000060                  CMP     (SP),#60        ;;CHAR < 0?
```

```
7414  032736  002420                        BLT     18$             ;;BRANCH IF YES
7415  032740  021627  000067                CMP     (SP),#67        ;;CHAR > 7?
7416  032744  003015                        BGT     18$             ;;BRANCH IF YES
7417  032746  042726  000060                BIC     #60,(SP)+       ;;STRIP-OFF ASCII
7418  032752  005766  000002                TST     2(SP)           ;;IS THIS THE FIRST CHAR
7419  032756  001403                        BEQ     17$             ;;BRANCH IF YES
7420  032760  006316                        ASL     (SP)            ;;NO, SHIFT PRESENT
7421  032762  006316                        ASL     (SP)            ;;   CHAR OVER TO MAKE
7422  032764  006316                        ASL     (SP)            ;;   ROOM FOR NEW ONE.
7423  032766  005266  000002        17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
7424  032772  056616  177776                BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
7425  032776  000667                        BR      7$              ;;GET THE NEXT ONE
7426  033000  104400  001170        18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
7427  033004  000720                        BR      20$             ;;SIMULATE CONTROL-U
7428                                .DSABL  LSB
7429
7430
7431                                ;;**********************************************************
7432                                ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
7433                                ;*CALL:
7434                                ;*      RDCHR                   ;;GET A CHARACTER FROM THE QUEUE
7435                                ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
7436                                ;*                              ;;WITH PARITY BIT STRIPPED OFF
7437                                ;
7438
7439  033006  011646                $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC AND
7440  033010  016666  000004  000002        MOV     4(SP),2(SP)     ;;THE PS
7441  033016  005066  000004                CLR     4(SP)           ;;GET READY FOR A CHARACTER
7442  033022  005046                        CLR     -(SP)           ;;PUT NEW PS ON STACK
7443  033024  012746  033032                MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
7444  033030  000002                        RTI                     ;;POP NEW PC AND PS
7445  033032                        64$:
7446  033032  005737  032146        1$:     TST     $TKCNT          ;;WAIT ON A CHARACTER
7447  033036  001775                        BEQ     1$
7448  033040  005337  032146                DEC     $TKCNT          ;;DECREMENT THE COUNTER
7449  033044  117766  177102  000004        MOVB    @$TKQOUT,4(SP)  ;;GET ONE CHARACTER
7450  033052  005237  032152                INC     $TKQOUT         ;;UPDATE THE POINTER
7451  033056  023727  032152  032155        CMP     $TKQOUT,#$TKQEND ;;DID IT GO OFF OF THE END?
7452  033064  001003                        BNE     2$              ;;BRANCH IF NO
7453  033066  012737  032154  032152        MOV     #$TKQSRT,$TKQOUT ;;RESET THE POINTER
7454  033074  000002                2$:     RTI                     ;;RETURN
7455                                ;;**********************************************************
7456                                ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
7457                                ;*CALL:
7458                                ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
7459                                ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
7460                                ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
7461
7462  033076  010346                $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
7463  033100  005046                        CLR     -(SP)           ;;CLEAR THE RUBOUT KEY
7464  033102  012703  033332        1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
7465  033106  022703  033424        2$:     CMP     #$TTYIN+72,R3   ;;BUFFER FULL?
7466  033112  101456                        BLOS    4$              ;;BR IF YES
7467  033114  104407                        RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
7468  033116  112613                        MOVB    (SP)+,(R3)      ;;GET CHARACTER
7469  033120  122713  000177        10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
```

```
 7470  033124  001022                          BNE     5$              ;;BR IF NO
 7471  033126  005716                          TST     (SP)            ;;IS THIS THE FIRST RUBOUT?
 7472  033130  001007                          BNE     6$              ;;BR IF NO
 7473  033132  112727  000134  033330          MOVB    #'\,9$          ;;TYPE A BACK SLASH
 7474  033140  104400  033330                  TYPE    ,9$
 7475  033144  012716  177777                  MOV     #-1,(SP)        ;;SET THE RUBOUT KEY
 7476  033150  005303                  6$:     DEC     R3              ;;BACKUP BY ONE
 7477  033152  020327  033332                  CMP     R3,#$TTYIN      ;;STACK EMPTY?
 7478  033156  103434                          BLO     4$              ;;BR IF YES
 7479  033160  111337  033330                  MOVB    (R3),9$         ;;SETUP TO TYPEOUT THE DELETED CHAR.
 7480  033164  104400  033330                  TYPE    ,9$             ;;GO TYPE
 7481  033170  000746                          BR      2$              ;;GO READ ANOTHER CHAR.
 7482  033172  005716                  5$:     TST     (SP)            ;;RUBOUT KEY SET?
 7483  033174  001406                          BEQ     7$              ;;BR IF NO
 7484  033176  112737  000134  033330          MOVB    #'\,9$          ;;TYPE A BACK SLASH
 7485  033204  104400  033330                  TYPE    ,9$
 7486  033210  005016                          CLR     (SP)            ;;CLEAR THE RUBOUT KEY
 7487  033212  122713  000025          7$:     CMPB    #25,(R3)        ;;IS CHARACTER A CTRL U?
 7488  033216  001003                          BNE     8$              ;;BR IF NO
 7489  033220  104400  033431                  TYPE    ,$CNTLU         ;;TYPE A CONTROL "U"
 7490  033224  000726                          BR      1$              ;;GO START OVER
 7491  033226  122713  000022          8$:     CMPB    #22,(R3)        ;;IS CHARACTER A "↑R"?
 7492  033232  001011                          BNE     3$              ;;BRANCH IF NO
 7493  033234  105013                          CLRB    (R3)            ;;CLEAR THE CHARACTER
 7494  033236  104400  001171                  TYPE    ,$CRLF          ;;TYPE A "CR" & "LF"
 7495  033242  104400  033332                  TYPE    ,$TTYIN         ;;TYPE THE INPUT STRING
 7496  033246  000717                          BR      2$              ;;GO PICKUP ANOTHER CHACTER
 7497  033250  104400  001170          4$:     TYPE    ,$QUES          ;;TYPE A '?'
 7498  033254  000712                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
 7499  033256  111337  033330          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
 7500  033262  104400  033330                  TYPE    ,9$
 7501  033266  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
 7502  033272  001305                          BNE     2$              ;;LOOP IF NOT RETURN
 7503  033274  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
 7504  033300  104400  001172                  TYPE    ,$LF            ;;TYPE A LINE FEED
 7505  033304  005726                          TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
 7506  033306  012603                          MOV     (SP)+,R3        ;;RESTORE R3
 7507  033310  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
 7508  033312  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
 7509  033320  012766  033332  000004          MOV     #$TTYIN,4(SP)
 7510  033326  000002                          RTI                     ;;RETURN
 7511  033330     000                  9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
 7512  033331     000                          .BYTE   0               ;;TERMINATOR
 7513  033332  000072                  $TTYIN: .BLKB   72              ;;RESERVE 72 BYTES FOR TTY INPUT
 7514  033424  041536  005015     000  $CNTLC: .ASCIZ  /↑C/<15><12>    ;;CONTROL "C"
 7515  033431     136  006525  000012  $CNTLU: .ASCIZ  /↑U/<15><12>    ;;CONTROL "U"
 7516  033436  043536  005015     000  $CNTLG: .ASCIZ  /↑G/<15><12>    ;;CONTROL "G"
 7517  033443     015  051412  051127  $MSWR:  .ASCIZ  <15><12>/SWR = /
 7518  033450  036440  000040
 7519  033454  020040  042516  020127  $MNEW:  .ASCIZ  /  NEW = /
 7520  033462  020075     000
 7521          033466                  .EVEN
 7522                                  .SBTTL  DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE
 7523
 7524                                  ;;**************************************************************
 7525                                  ;;THIS ROUTINE WILL CONVERT A 32-BIT BINARY NUMBER TO AN UNSIGNED
```

```
                                        ;*DECIMAL (ASCII) NUMBER. THE SIGN OF THE BINARY NUMBER MUST BE
                                        ;*POSITIVE.
                                        ;*CALL
                                        ;*      MOV     #PNTR,-(SP)     ;;POINTER TO LOW WORD OF BINARY NUMBER
                                        ;*      JSR     PC,@#$DB2D
                                        ;*      RETURN                  ;;THE FIRST ADDRESS OF ASCIZ
                                                                        ;;IS ON THE STACK       .

033466  104411                  $DB2D:  SAVREG                          ;;SAVE REGISTERS
033470  016602  000002                  MOV     2(SP),R2                ;;PICKUP THE DATA POINTER
033474  012700  033646                  MOV     #$DECVL,R0              ;;GET ADDRESS OF "$DECVL" STRING
033500  010066  000002                  MOV     R0,2(SP)                ;;PUT ADDRESS OF ASCIZ STRING ON STACK
033504  012201                          MOV     (R2)+,R1                ;;PICKUP THE BINARY NUMBER
033506  012202                          MOV     (R2)+,R2
033510  012737  000012  033564          MOV     #10.,4$                 ;;SET UP TO DO 10 CONVERSIONS
033516  012704  033576                  MOV     #$TNPWR,R4              ;;ADDRESS OF TEN POWER
033522  012705  033600                  MOV     #$TNPWR+2,R5
033526  005003                  1$:     CLR     R3                      ;;CLEAR PARTIAL
033530  161401                  2$:     SUB     (R4),R1                 ;;SUBTRACT TEN POWER
033532  005602                          SBC     R2
033534  161502                          SUB     (R5),R2
033536  002402                          BLT     3$                      ;;BR IF TEN POWER TO LARGE
033540  005203                          INC     R3                      ;;ADD 1 TO PARTIAL
033542  000772                          BR      2$                      ;;LOOP
033544  062401                  3$:     ADD     (R4)+,R1                ;;RESTORE SUBTRACTED VALUE
033546  005502                          ADC     R2
033550  062402                          ADD     (R4)+,R2
033552  022525                          CMP     (R5)+,(R5)+             ;;MOVE TO NEXT TEN POWER
033554  052703  000060                  BIS     #'0,R3                  ;;CHANGE PARTIAL TO ASCII
033560  110320                          MOVB    R3,(R0)+                ;;SAVE IT
033562  005327                          DEC     (PC)+                   ;;DONE?
033564  000000                  4$:     .WORD   0
033566  001357                          BNE     1$                      ;;BR IF NO
033570  105020                          CLRB    (R0)+                   ;;TERMINATOR
033572  104412                          RESREG                          ;;RESTORE REGISTERS
033574  000207                          RTS     PC                      ;;RETURN
033576  145000                  $TNPWR: 145000                          ;;1.0E09
033600  035632                          35632
033602  160400                          160400                          ;;1.0E08
033604  002765                          2765
033606  113200                          113200                          ;;1.0E07
033610  000230                          230
033612  041100                          041100                          ;;1.0E06
033614  000017                          17
033616  103240                          103240                          ;;1.0E05
033620  000001                          1
033622  023420                          23420                           ;;1.0E04
033624  000000                          0
033626  001750                          1750                            ;;1.0E03
033630  000000                          0
033632  000144                          144                             ;;1.0E02
033634  000000                          0
033636  000012                          12                              ;;1.0E01
033640  000000                          0
033642  000001                          1                               ;;1.0E00
```

```
7582  033644  000000                          0
7583  033646  000014          $DECVL: .BLKB   12.              ;;RESERVE STORAGE FOR ASCIZ STRING
7584                          .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES
7585
7586                          ;;*****************************************************************
7587                          ;*SAVE R0-R5
7588                          ;*CALL:
7589                          ;*      SAVREG
7590                          ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
7591                          ;*
7592                          ;*TOP---(+16)
7593                          ;* +2---(+18)
7594                          ;* +4---R5
7595                          ;* +6---R4
7596                          ;* +8---R3
7597                          ;*+10---R2
7598                          ;*+12---R1
7599                          ;*+14---R0
7600
7601  033662                  $SAVREG:
7602  033662  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
7603  033664  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
7604  033666  010246                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
7605  033670  010346                  MOV     R3,-(SP)        ;;PUSH R3 ON STACK
7606  033672  010446                  MOV     R4,-(SP)        ;;PUSH R4 ON STACK
7607  033674  010546                  MOV     R5,-(SP)        ;;PUSH R5 ON STACK
7608  033676  016646  000022          MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
7609  033702  016646  000022          MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
7610  033706  016646  000022          MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
7611  033712  016646  000022          MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
7612  033716  000002                  RTI
7613
7614                          ;*RESTORE R0-R5
7615                          ;*CALL:
7616                          ;*      RESREG
7617  033720                  $RESREG:
7618  033720  012666  000022          MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
7619  033724  012666  000022          MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
7620  033730  012666  000022          MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
7621  033734  012666  000022          MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
7622  033740  012605                  MOV     (SP)+,R5        ;;POP STACK INTO R5
7623  033742  012604                  MOV     (SP)+,R4        ;;POP STACK INTO R4
7624  033744  012603                  MOV     (SP)+,R3        ;;POP STACK INTO R3
7625  033746  012602                  MOV     (SP)+,R2        ;;POP STACK INTO R2
7626  033750  012601                  MOV     (SP)+,R1        ;;POP STACK INTO R1
7627  033752  012600                  MOV     (SP)+,R0        ;;POP STACK INTO R0
7628  033754  000002                  RTI
7629                          .SBTTL  RANDOM NUMBER GENERATOR ROUTINE
7630
7631                          ;;*****************************************************************
7632                          ;*THIS ROUTINE IS A DOUBLE PRECISION PSEUDO RANDOM NUMBER GENERATOR
7633                          ;*WITH A RANGE OF 0 TO 2(+33)-1.
7634                          ;*CALL:
7635                          ;*      JSR     PC,$RAND        ;;CALL THE ROUTINE
7636                          ;*      RETURN                  ;;RETURN HERE THE RANDOM
7637                          ;*                              ;;NUMBER WILL BE IN
```

```
7638                                    :*                              ;;$HINUM,$LONUM
7639
7640  033756                            $RAND:
7641  033756  010046                            MOV     R0,-(SP)        ;;PUSH R0 ON STACK
7642  033760  010146                            MOV     R1,-(SP)        ;;PUSH R1 ON STACK
7643  033762  010246                            MOV     R2,-(SP)        ;;PUSH R2 ON STACK
7644  033764  013700  034056                    MOV     $LONUM,R0       ;;SET R0 WITH LOW
7645  033770  013701  034054                    MOV     $HINUM,R1       ;;SET R1 WITH HIGH
7646  033774  012702  177771                    MOV     #-7,R2          ;;SET SHIFT COUNT
7647  034000  006300                    1$:     ASL     R0              ;;SHIFT R0 LEFT AND
7648  034002  006101                            ROL     R1              ;;ROTATE CARRY INTO R1 AND
7649  034004  005202                            INC     R2              ;;CHECK FOR DONE
7650  034006  001374                            BNE     1$              ;;CONTINUE SHIFT LOOP
7651  034010  063700  034056                    ADD     $LONUM,R0       ;;ADD NUMBER TO MAKE X 129
7652  034014  005501                            ADC     R1              ;;PROPOGATE CARRY
7653  034016  063701  034054                    ADD     $HINUM,R1       ;;ADD NUMBER TO MAKE X 129
7654  034022  062700  001057                    ADD     #1057,R0        ;;ADD LOW CONSTANT
7655  034026  005501                            ADC     R1              ;;PROPOGATE CARRY
7656  034030  062701  047401                    ADD     #47401,R1       ;;ADD HIGH CONSTANT
7657  034034  010037  034056                    MOV     R0,$LONUM       ;;SAVE R0
7658  034040  010137  034054                    MOV     R1,$HINUM       ;;SAVE R1
7659  034044  012602                            MOV     (SP)+,R2        ;;POP STACK INTO R2
7660  034046  012601                            MOV     (SP)+,R1        ;;POP STACK INTO R1
7661  034050  012600                            MOV     (SP)+,R0        ;;POP STACK INTO R0
7662  034052  000207                            RTS     PC              ;;RETURN
7663  034054  176543                    $HINUM: .WORD   176543
7664  034056  123456                    $LONUM: .WORD   123456
7665                                    .SBTTL  ROUTINE TO SIZE MEMORY
7666
7667                                    ;;**************************************************************
7668                                    ;*CALL:
7669                                    ;*      JSR     PC,$SIZE
7670                                    ;*      RETURN
7671                                    ;*$LSTAD WILL CONTAIN THE LAST AVAILABLE MEMORY LOCATION
7672
7673  034060  010046                    $SIZE:  MOV     R0,-(SP)        ;;SAVE R0 ON THE STACK
7674  034062  010146                            MOV     R1,-(SP)        ;;SAVE R1 ON THE STACK
7675  034064  013746  000004                    MOV     @#ERRVEC,-(SP)  ;;SAVE PRESENT ERROR VECTOR PS & PC
7676  034070  013746  000006                    MOV     @#ERRVEC+2,-(SP)
7677  034074  010600                            MOV     SP,R0           ;;SAVE THE STACK POINTER
7678                                    ;;SET THE ERRVEC PS TO THE PRESENT PS
7679  034076  013746  000034                    MOV     @#TRAPVEC,-(SP) ;;SAVE CURRENT TRAP VECTOR
7680  034102  012737  034112  000034            MOV     #64$,@#TRAPVEC  ;;SETUP NEW TRAP VECTOT
7681  034110  104400                            TRAP                    ;;PUSH OLD PSW AND PC ON STACK
7682  034112  016637  000002  000006  64$:     MOV     2(SP),@#ERRVEC+2        ;;SAVE PSW IN @#ERRVEC+2
7683  034120  012716  034126                    MOV     #65$,(SP)               ;;REPLACE OLD PC WITH NEW
7684  034124  000002                            RTI                     ;;RESTORE PSW
7685  034126  012637  000034            65$:    MOV     (SP)+,@#TRAPVEC ;;RESTORE OLD TRAP  VECTOR
7686  034132  012737  034152  000004            MOV     #2$,@#ERRVEC    ;;SET FOR TIMEOUT
7687  034140  012701  020000                    MOV     #20000,R1       ;;FIRST ADDRESS
7688  034144  005711                    1$:     TST     (R1)            ;;TEST THIS ADDRESS
7689  034146  005721                            TST     (R1)+           ;;STEP TO NEXT ADDRESS
7690  034150  000775                            BR      1$              ;;TRY ANOTHER
7691  034152  162701  000002            2$:     SUB     #2,R1           ;;DROP BACK
7692  034156  010006                            MOV     R0,SP           ;;RESTORE THE STACK
7693  034160  012637  000006                    MOV     (SP)+,@#ERRVEC+2 ;;RESTORE ERROR VECTOR
```

```
7694  034164  012637  000004                  MOV     (SP)+,@#ERRVEC
7695  034170  010137  034202                  MOV     R1,$LSTAD       ;;LAST ADDRESS
7696  034174  012601                          MOV     (SP)+,R1        ;;RESTORE R1
7697  034176  012600                          MOV     (SP)+,R0        ;;RESTORE R0
7698  034200  000207                          RTS     PC
7699  034202  000000              $LSTAD:     .WORD   0               ;;CONTAINS THE LAST ADDRESS
7700                              .SBTTL  POWER DOWN AND UP ROUTINES
7701
7702                              ;;**************************************************************
7703                              ;POWER DOWN ROUTINE
7704  034204  012737  034344  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
7705  034212  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
7706  034220  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
7707  034222  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
7708  034224  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
7709  034226  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
7710  034230  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
7711  034232  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
7712  034234  017746  144700                  MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
7713  034240  010637  034350                  MOV     SP,$SAVR6       ;;SAVE SP
7714  034244  012737  034256  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
7715  034252  000000                          HALT
7716  034254  000776                          BR      .-2             ;;HANG UP
7717
7718                              ;;**************************************************************
7719                              ;POWER UP ROUTINE
7720  034256  012737  034344  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
7721  034264  013706  034350                  MOV     $SAVR6,SP       ;;GET SP
7722  034270  005037  034350                  CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
7723  034274  005237  034350          1$:     INC     $SAVR6          ;;WAIT FOR THE INC
7724  034300  001375                          BNE     1$              ;;OF  WORD
7725  034302  012677  144632                  MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
7726  034306  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
7727  034310  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
7728  034312  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
7729  034314  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
7730  034316  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
7731  034320  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
7732  034322  012737  034204  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
7733  034330  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
7734  034336  104400                          TYPE                    ;REPORT THE POWER FAILURE
7735  034340  034352              $PWRMG:     .WORD   $POWER          ;;POWER FAIL MESSAGE POINTER
7736  034342  000002                          RTI
7737  034344  000000              $ILLUP:     HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
7738  034346  000776                          BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
7739  034350  000000              $SAVR6:     0                       ;;PUT THE SP HERE
7740  034352  005015  047520  042527  $POWER: .ASCIZ  <15><12>"POWER"
7741  034360  000122
7742                                  .EVEN
7743                              .SBTTL  TRAP DECODER
7744
7745                              ;;**************************************************************
7746                              ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
7747                              ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
7748                              ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
7749                              ;*GO TO THAT ROUTINE.
```

```
7750
7751  034362  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
7752  034364  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
7753  034370  005740                          TST     -(R0)           ;;BACKUP BY 2
7754  034372  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
7755  034374  006300                          ASL     R0              ;;POSITION FOR INDEXING
7756  034376  016000  034404                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
7757  034402  000200                          RTS     R0              ;;GO TO ROUTINE
7758
7759                                  .SBTTL  TRAP TABLE
7760
7761                                  ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
7762                                  ;*BY THE "TRAP" INSTRUCTION.
7763
7764                                  ;       ROUTINE
7765                                  ;       -------
7766  034404                          $TRPAD:
7767  034404  031254                          $TYPE   ;;CALL=TYPE     TRAP+0(104400)  TTY TYPEOUT ROUTINE
7768  034406  031744                          $TYPOC  ;;CALL=TYPOC    TRAP+1(104401)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
7769  034410  031720                          $TYPOS  ;;CALL=TYPOS    TRAP+2(104402)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
7770  034412  031760                          $TYPON  ;;CALL=TYPON    TRAP+3(104403)  TYPE OCTAL NUMBER (AS PER LAST CALL)
7771  034414  031474                          $TYPDS  ;;CALL=TYPDS    TRAP+4(104404)  TYPE DECIMAL NUMBER (WITH SIGN)
7772
7773  034416  032534                          $GTSWR  ;;CALL=GTSWR    TRAP+5(104405)  GET SOFT-SWR SETTING
7774
7775  034420  032444                          $CKSWR  ;;CALL=CKSWR    TRAP+6(104406)  TEST FOR CHANGE IN SOFT-SWR
7776  034422  033006                          $RDCHR  ;;CALL=RDCHR    TRAP+7(104407)  TTY TYPEIN CHARACTER ROUTINE
7777  034424  033076                          $RDLIN  ;;CALL=RDLIN    TRAP+10(104410) TTY TYPEIN STRING ROUTINE
7778  034426  033662                          $SAVREG ;;CALL=SAVREG   TRAP+11(104411) SAVE R0-R5 ROUTINE
7779  034430  033720                          $RESREG ;;CALL=RESREG   TRAP+12(104412) RESTORE R0-R5 ROUTINE
7780  034432  016732                          FPRINT  ;;CALL=FPRINT   TRAP+13(104413) FAILURE PRINT ROUTINE
7781  034434  000000                  OFILE:  .WORD   0
7782  034436  046511  042515  044504  HPDATA: .ASCII  /IMMEDIATE COMMAND SUMMARY:/<15><12><12>
7783  034444  052101  020105  047503
7784  034452  046515  047101  020104
7785  034460  052523  046515  051101
7786  034466  035131  005015     012
7787  034473     103  046517  040515          .ASCII  /COMMAND                    NMENONIC        PARAMETERS/<15><12><12>
7788  034500  042116  020040  020040
7789  034506  020040  020040  020040
7790  034514  020040  020040  020040
7791  034522  047040  042515  047516
7792  034530  044516  020103  020040
7793  034536  020040  050040  051101
7794  034544  046501  052105  051105
7795  034552  006523  005012
7796  034556  051104  053111  020105          .ASCII  /DRIVE SELECT                 DN          ,DRIVE NUMBER/<15><12>
7797  034564  042523  042514  052103
7798  034572  020040  020040  020040
7799  034600  020040  020040  020040
7800  034606  020040  042040  020116
7801  034614  020040  020040  020040
7802  034622  020040  042054  044522
7803  034630  042526  047040  046525
7804  034636  042502  006522     012
7805  034643     040  020040  020040          .ASCII  /                                       ,? (PRINT DRIVE SELECTED)/<15><12
```

```
 7806  034650  020040  020040  020040
 7807  034656  020040  020040  020040
 7808  034664  020040  020040  020040
 7809  034672  020040  020040  020040
 7810  034700  020040  020040  020040
 7811  034706  020040  026040  020077
 7812  034714  050050  044522  052116
 7813  034722  042040  044522  042526
 7814  034730  051440  046105  041505
 7815  034736  042524  024504  005015
 7816  034744     012
 7817  034745     106  051117  040515        .ASCII  /FORMAT SELECT              FT         ,FORMAT(24 OR 26)/<15><12>
 7818  034752  020124  042523  042514
 7819  034760  052103  020040  020040
 7820  034766  020040  020040  020040
 7821  034774  020040  020040  052106
 7822  035002  020040  020040  020040
 7823  035010  020040  026040  047506
 7824  035016  046522  052101  031050
 7825  035024  020064  051117  031040
 7826  035032  024466  005015
 7827  035036  020040  020040  020040        .ASCII  /                                ,? (PRINT FORMAT SELECTED)/<15><1
 7828  035044  020040  020040  020040
 7829  035052  020040  020040  020040
 7830  035060  020040  020040  020040
 7831  035066  020040  020040  020040
 7832  035074  020040  020040  020040
 7833  035102  020040  037454  024040
 7834  035110  051120  047111  020124
 7835  035116  047506  046522  052101
 7836  035124  051440  046105  041505
 7837  035132  042524  024504  005015
 7838  035140  052517  050124  052125        .ASCII  /OUTPUT TEST                OT         ,O (OBJECT)/<15><12>
 7839  035146  052040  051505  020124
 7840  035154  020040  020040  020040
 7841  035162  020040  020040  020040
 7842  035170  020040  047440  020124
 7843  035176  020040  020040  020040
 7844  035204  020040  047454  024040
 7845  035212  041117  042512  052103
 7846  035220  006451     012
 7847  035223     040  020040  020040        .ASCII  /                                ,S (SOURCE)/<15><12><12>
 7848  035230  020040  020040  020040
 7849  035236  020040  020040  020040
 7850  035244  020040  020040  020040
 7851  035252  020040  020040  020040
 7852  035260  020040  020040  020040
 7853  035266  020040  026040  020123
 7854  035274  051450  052517  041522
 7855  035302  024505  005015     012
 7856  035307     111  050116  052125        .ASCII  /INPUT TEST                 IT/<15><12><12>
 7857  035314  052040  051505  020124
 7858  035322  020040  020040  020040
 7859  035330  020040  020040  020040
 7860  035336  020040  020040  052111
 7861  035344  005015     012
```

```
 7862  035347     111  050116  052125          .ASCII  /INPUT STRING                IS/<15><12><12>
 7863  035354  051440  051124  047111
 7864  035362  020107  020040  020040
 7865  035370  020040  020040  020040
 7866  035376  020040  020040  051511
 7867  035404  005015     012
 7868  035407     103  050117  020131          .ASCII  /COPY TAPE                  CT/<15><12><12>
 7869  035414  040524  042520  020040
 7870  035422  020040  020040  020040
 7871  035430  020040  020040  020040
 7872  035436  020040  020040  052103
 7873  035444  005015     012
 7874  035447     111  042524  040522          .ASCII  /ITERATION COUNT            IC          ,NNNNN (DECIMAL COUNT)/<15><12>
 7875  035454  044524  047117  041440
 7876  035462  052517  052116  020040
 7877  035470  020040  020040  020040
 7878  035476  020040  020040  041511
 7879  035504  020040  020040  020040
 7880  035512  020040  026040  047116
 7881  035520  047116  020116  042050
 7882  035526  041505  046511  046101
 7883  035534  041440  052517  052116
 7884  035542  006451     012
 7885  035545     040  020040  020040          .ASCII  /                                   ,? (PRINT COUNT)/<15><12><12>
 7886  035552  020040  020040  020040
 7887  035560  020040  020040  020040
 7888  035566  020040  020040  020040
 7889  035574  020040  020040  020040
 7890  035602  020040  020040  020040
 7891  035610  020040  026040  020077
 7892  035616  050050  044522  052116
 7893  035624  041440  052517  052116
 7894  035632  006451  005012
 7895  035636  052502  043106  051105          .ASCII  /BUFFER DUMP                BD          ,BUFFER NAME(H,R,W,X,Y, OR Z)/<15
 7896  035644  042040  046525  020120
 7897  035652  020040  020040  020040
 7898  035660  020040  020040  020040
 7899  035666  020040  041040  020104
 7900  035674  020040  020040  020040
 7901  035702  020040  041054  043125
 7902  035710  042506  020122  040516
 7903  035716  042515  044050  051054
 7904  035724  053454  054054  054454
 7905  035732  020054  051117  055040
 7906  035740  006451     012
 7907  035743     040  020040  020040          .ASCII  /                                   ,NUMBER OF WORDS/<15><12>
 7908  035750  020040  020040  020040
 7909  035756  020040  020040  020040
 7910  035764  020040  020040  020040
 7911  035772  020040  020040  020040
 7912  036000  020040  020040  020040
 7913  036006  020040  026040  052516
 7914  036014  041115  051105  047440
 7915  036022  020106  047527  042122
 7916  036030  006523     012
 7917  036033     123  042520  044503          .ASCII  /SPECIAL DATA PATTERN       DP          ,PATTERN NAME (X,Y,Z)/<15><12>
```

```
 7918  036040  046101  042040  052101
 7919  036046  020101  040520  052124
 7920  036054  051105  020116  020040
 7921  036062  020040  020040  050104
 7922  036070  020040  020040  020040
 7923  036076  020040  026040  040520
 7924  036104  052124  051105  020116
 7925  036112  040516  042515  024040
 7926  036120  026130  026131  024532
 7927  036126  005015
 7928  036130  020040  020040  020040          .ASCII  /                                       ,DDDD...D(32 WORDS)/<15><12><12>
 7929  036136  020040  020040  020040
 7930  036144  020040  020040  020040
 7931  036152  020040  020040  020040
 7932  036160  020040  020040  020040
 7933  036166  020040  020040  020040
 7934  036174  020040  042054  042104
 7935  036202  027104  027056  024104
 7936  036210  031063  053440  051117
 7937  036216  051504  006451  005012
 7938  036224  042105  052111  041040          .ASCII  /EDIT BUFFER                    EB      ,PATTER NAME (X,Y,Z)/<15><12>
 7939  036232  043125  042506  020122
 7940  036240  020040  020040  020040
 7941  036246  020040  020040  020040
 7942  036254  020040  042440  020102
 7943  036262  020040  020040  020040
 7944  036270  020040  050054  052101
 7945  036276  042524  020122  040516
 7946  036304  042515  024040  026130
 7947  036312  026131  024532  005015
 7948  036320  020040  020040  020040          .ASCII  /                                       ,STARTING WORD NUMBER/<15><12>
 7949  036326  020040  020040  020040
 7950  036334  020040  020040  020040
 7951  036342  020040  020040  020040
 7952  036350  020040  020040  020040
 7953  036356  020040  020040  020040
 7954  036364  020040  051454  040524
 7955  036372  052122  047111  020107
 7956  036400  047527  042122  047040
 7957  036406  046525  042502  006522
 7958  036414     012
 7959  036415     040  020040  020040          .ASCII  /                                       ,DD..D(UP TO 32 WORDS)/<15><12>
 7960  036422  020040  020040  020040
 7961  036430  020040  020040  020040
 7962  036436  020040  020040  020040
 7963  036444  020040  020040  020040
 7964  036452  020040  020040  020040
 7965  036460  020040  026040  042104
 7966  036466  027056  024104  050125
 7967  036474  052040  020117  031063
 7968  036502  053440  051117  051504
 7969  036510  006451     012
 7970  036513     103  046517  044520          .ASCII  /COMPILE                        CO      ,NC (NO CHECK)/<15><12>
 7971  036520  042514  020040  020040
 7972  036526  020040  020040  020040
 7973  036534  020040  020040  020040
```

```
 7974  036542  020040  020040  047303
 7975  036550  020040  020040  020040
 7976  036556  020040  026040  041516
 7977  036564  024040  047516  041440
 7978  036572  042510  045503  006451
 7979  036600     012
 7980  036601     040  020040  020040          .ASCII  /                                ,BII (BUS INCREMENT INHIBIT)/<15>
 7981  036606  020040  020040  020040
 7982  036614  020040  020040  020040
 7983  036622  020040  020040  020040
 7984  036630  020040  020040  020040
 7985  036636  020040  020040  020040
 7986  036644  020040  026040  044502
 7987  036652  020111  041050  051525
 7988  036660  044440  041516  042522
 7989  036666  042515  052116  044440
 7990  036674  044116  041111  052111
 7991  036702  006451  005012
 7992  036706  042105  052111  040440          .ASCII  /EDIT ADD LINE           EA      ,LINE NUMBER/<15><12>
 7993  036714  042104  046040  047111
 7994  036722  020105  020040  020040
 7995  036730  020040  020040  020040
 7996  036736  020040  042440  020101
 7997  036744  020040  020040  020040
 7998  036752  020040  046054  047111
 7999  036760  020105  052516  041115
 8000  036766  051105  005015
 8001  036772  020040  020040  020040          .ASCII  /                                ,NEW COMMAND/<15><12><12>
 8002  037000  020040  020040  020040
 8003  037006  020040  020040  020040
 8004  037014  020040  020040  020040
 8005  037022  020040  020040  020040
 8006  037030  020040  020040  020040
 8007  037036  020040  047054  053505
 8008  037044  041440  046517  040515
 8009  037052  042116  005015     012
 8010  037057     105  044504  020124          .ASCII  /EDIT DELETE LINE        ED      ,LINE NUMBER/<15><12><12>
 8011  037064  042504  042514  042524
 8012  037072  046040  047111  020105
 8013  037100  020040  020040  020040
 8014  037106  020040  020040  042105
 8015  037114  020040  020040  020040
 8016  037122  020040  026040  044514
 8017  037130  042516  047040  046525
 8018  037136  042502  006522  005012
 8019  037144  051120  047111  020124          .ASCII  /PRINT TEST              PT/<15><12><12>
 8020  037152  042524  052123  020040
 8021  037160  020040  020040  020040
 8022  037166  020040  020040  020040
 8023  037174  020040  050040  006524
 8024  037202  005012
 8025  037204  051120  047111  020124          .ASCII  /PRINT LINE              PL      ,LINE NUMBER/<15><12><12>
 8026  037212  044514  042516  020040
 8027  037220  020040  020040  020040
 8028  037226  020040  020040  020040
 8029  037234  020040  050040  020114
```

```
8030  037242  020040  020040  020040
8031  037250  020040  046054  047111
8032  037256  020105  052516  041115
8033  037264  051105  005015     012
8034  037271     116  053505  052040          .ASCII  /NEW TEST                    NT/<15><12><12>
8035  037276  051505  020124  020040
8036  037304  020040  020040  020040
8037  037312  020040  020040  020040
8038  037320  020040  020040  052116
8039  037326  005015     012
8040  037331     122  047125  020040          .ASCII  /RUN                        RU/<15><12><12>
8041  037336  020040  020040  020040
8042  037344  020040  020040  020040
8043  037352  020040  020040  020040
8044  037360  020040  020040  052522
8045  037366  005015     012
8046  037371     120  044522  052116          .ASCII  /PRINT REGISTER             PR          ,REGISTER NAME OR NUMBER/<15><12>
8047  037376  051040  043505  051511
8048  037404  042524  020122  020040
8049  037412  020040  020040  020040
8050  037420  020040  020040  051120
8051  037426  020040  020040  020040
8052  037434  020040  026040  042522
8053  037442  044507  052123  051105
8054  037450  047040  046501  020105
8055  037456  051117  047040  046525
8056  037464  042502  006522  005012
8057  037472  042510  050114  020040          .ASCII  /HELP                       HP/<15><12><12>
8058  037500  020040  020040  020040
8059  037506  020040  020040  020040
8060  037514  020040  020040  020040
8061  037522  020040  044040  006520
8062  037530  005012
8063  037532  044524  042515  047440          .ASCII  /TIME OUT CHANGE            TO          ,NNNNN (DECIMAL)/<15><12>
8064  037540  052125  041440  040510
8065  037546  043516  020105  020040
8066  037554  020040  020040  020040
8067  037562  020040  052040  020117
8068  037570  020040  020040  020040
8069  037576  020040  047054  047116
8070  037604  047116  024040  042504
8071  037612  044503  040515  024514
8072  037620  005015
8073  037622  020040  020040  020040          .ASCII  /                                ,? (PRINT CONSTANT)/<15><12><12>
8074  037630  020040  020040  020040
8075  037636  020040  020040  020040
8076  037644  020040  020040  020040
8077  037652  020040  020040  020040
8078  037660  020040  020040  020040
8079  037666  020040  037454  024040
8080  037674  051120  047111  020124
8081  037702  047503  051516  040524
8082  037710  052116  006451  005012
8083  037716  046101  020114  042504          .ASCII  /ALL DECIMAL VALUES MUST BE 65535(10) OR LESS/<15><12><12><12>
8084  037724  044503  040515  020114
8085  037732  040526  052514  051505
```

```
 8096  037740  046440  051525  020124
 8097  037746  042502  033040  032465
 8098  037754  032463  030450  024460
 8099  037762  047440  020122  042514
 8100  037770  051523  005015  005012
 8101  037776  042504  042506  051122        .ASCII  /DEFERRED COMMAND SUMMARY:/<15><12><12>
 8102  040004  042105  041440  046517
 8103  040012  040515  042116  051440
 8104  040020  046525  040515  054522
 8105  040026  006472  005012
 8106  040032  047503  046515  047101        .ASCII  /COMMAND                    NMENONIC      PARAMETER/<15><12><12>
 8107  040040  020104  020040  020040
 8108  040046  020040  020040  020040
 8109  040054  020040  020040  020040
 8110  040062  020040  046516  047105
 8111  040070  047117  041511  020040
 8112  040076  020040  020040  040520
 8113  040104  040522  042515  042524
 8114  040112  006522  005012
 8115  040116  052523  051502  051531        .ASCII  /SUBSYSTEM FUNCTION          SF          ,SUBSYSTEM CMND/<15><12>
 8116  040124  042524  020115  052506
 8117  040132  041516  044524  047117
 8118  040140  020040  020040  020040
 8119  040146  020040  020040  051440
 8120  040154  020106  020040  020040
 8121  040162  020040  020040  051454
 8122  040170  041125  054523  052123
 8123  040176  046505  041440  047115
 8124  040204  006504     012
 8125  040207     040  020040  020040        .ASCII  /                                ,DRIVE NUM/<15><12>
 8126  040214  020040  020040  020040
 8127  040222  020040  020040  020040
 8128  040230  020040  020040  020040
 8129  040236  020040  020040  020040
 8130  040244  020040  020040  020040
 8131  040252  020040  020040  026040
 8132  040260  051104  053111  020105
 8133  040266  052516  006515     012
 8134  040273     040  020040  020040        .ASCII  /                                ,CYLINDER/<15><12>
 8135  040300  020040  020040  020040
 8136  040306  020040  020040  020040
 8137  040314  020040  020040  020040
 8138  040322  020040  020040  020040
 8139  040330  020040  020040  020040
 8140  040336  020040  020040  026040
 8141  040344  054503  044514  042116
 8142  040352  051105  005015
 8143  040356  020040  020040  020040        .ASCII  /                                ,TRACK/<15><12>
 8144  040364  020040  020040  020040
 8145  040372  020040  020040  020040
 8146  040400  020040  020040  020040
 8147  040406  020040  020040  020040
 8148  040414  020040  020040  020040
 8149  040422  020040  020040  052054
 8150  040430  040522  045503  005015
 8151  040436  020040  020040  020040        .ASCII  /                                ,SECTOR/<15><12>
```

```
8142  040444  020040  020040  020040
8143  040452  020040  020040  020040
8144  040460  020040  020040  020040
8145  040466  020040  020040  020040
8146  040474  020040  020040  020040
8147  040502  020040  020040  051454
8148  040510  041505  047524  006522
8149  040516     012
8150  040517     040  020040  020040          .ASCII  /                                   ,WORD COUNT/<15><12>
8151  040524  020040  020040  020040
8152  040532  020040  020040  020040
8153  040540  020040  020040  020040
8154  040546  020040  020040  020040
8155  040554  020040  020040  020040
8156  040562  020040  020040  026040
8157  040570  047527  042122  041440
8158  040576  052517  052116  005015
8159  040604  020040  020040  020040          .ASCII  /                                   ,DATA PATTERN/<15><12>
8160  040612  020040  020040  020040
8161  040620  020040  020040  020040
8162  040626  020040  020040  020040
8163  040634  020040  020040  020040
8164  040642  020040  020040  020040
8165  040650  020040  020040  042054
8166  040656  052101  020101  040520
8167  040664  052124  051105  006516
8168  040672     012
8169  040673     102  043125  042506          .ASCII  /BUFFER INITIALIZE          BI       ,PATTERN SELECT/<15><12><12>
8170  040700  020122  047111  052111
8171  040706  040511  044514  042532
8172  040714  020040  020040  020040
8173  040722  020040  020040  020040
8174  040730  044502  020040  020040
8175  040736  020040  020040  026040
8176  040744  040520  052124  051105
8177  040752  020116  042523  042514
8178  040760  052103  005015     012
8179  040763     104  052101  020101          .ASCII  /DATA COMPARE               DC       ,NNNNNN (OCTAL)/<15><12><12>
8180  040772  047503  050115  051101
8181  041000  020105  020040  020040
8182  041006  020040  020040  020040
8183  041014  020040  020040  020040
8184  041022  041504  020040  020040
8185  041030  020040  020040  026040
8186  041036  047116  047116  047116
8187  041044  024040  041517  040524
8188  041052  024514  005015     012
8189  041057     123  040524  052524          .ASCII  /STATUS COMPARE             SC       ,STATUS WD SELECT/<15><12>
8190  041064  020123  047503  050115
8191  041072  051101  020105  020040
8192  041100  020040  020040  020040
8193  041106  020040  020040  020040
8194  041114  041523  020040  020040
8195  041122  020040  020040  026040
8196  041130  052123  052101  051525
8197  041136  053440  020104  042523
```

```
 8198  041144  042514  052103  005015
 8199  041152  020040  020040  020040          .ASCII                                  ,EXPECTED VALUE/<15><12>
 8200  041160  020040  020040  020040
 8201  041166  020040  020040  020040
 8202  041174  020040  020040  020040
 8203  041202  020040  020040  020040
 8204  041210  020040  020040  020040
 8205  041216  020040  020040  042454
 8206  041224  050130  041505  042524
 8207  041232  020104  040526  052514
 8208  041240  006505     012
 8209  041243     040  020040  020040          .ASCII  /                               ,MASK/<15><12><12>
 8210  041250  020040  020040  020040
 8211  041256  020040  020040  020040
 8212  041264  020040  020040  020040
 8213  041272  020040  020040  020040
 8214  041300  020040  020040  020040
 8215  041306  020040  020040  026040
 8216  041314  040515  045523  005015
 8217  041322     012
 8218  041323     122  043505  051511          .ASCII  /REGISTER WRITE          RW     ,REG NAME OR NUM/<15><12>
 8219  041330  042524  020122  051127
 8220  041336  052111  020105  020040
 8221  041344  020040  020040  020040
 8222  041352  020040  020040  020040
 8223  041360  053522  020040  020040
 8224  041366  020040  020040  026040
 8225  041374  042522  020107  040516
 8226  041402  042515  047440  020122
 8227  041410  052516  006515     012
 8228          041416                          .EVEN
 8229  041416  020040  020040  020040  ENDLOC: .ASCII  /                               ,VALUE/<15><12><12>
 8230  041424  020040  020040  020040
 8231  041432  020040  020040  020040
 8232  041440  020040  020040  020040
 8233  041446  020040  020040  020040
 8234  041454  020040  020040  020040
 8235  041462  020040  020040  053054
 8236  041470  046101  042525  005015
 8237  041476     012
 8238  041477     122  043505  051511          .ASCII  /REGISTER COMPARE        RC     ,REG NAME OR NUM/<15><12>
 8239  041504  042524  020122  047503
 8240  041512  050115  051101  020105
 8241  041520  020040  020040  020040
 8242  041526  020040  020040  020040
 8243  041534  041522  020040  020040
 8244  041542  020040  020040  026040
 8245  041550  042522  020107  040516
 8246  041556  042515  047440  020122
 8247  041564  052516  006515     012
 8248  041571     040  020040  020040          .ASCII  /                               ,EXPECTED VALUE/<15><12>
 8249  041576  020040  020040  020040
 8250  041604  020040  020040  020040
 8251  041612  020040  020040  020040
 8252  041620  020040  020040  020040
 8253  041626  020040  020040  020040
```

```
8254  041634  020040  020040  026040
8255  041642  054105  042520  052103
8256  041650  042105  053040  046101
8257  041656  042525  005015
8258  041662  020040  020040  020040          .ASCII                                  ,MASK/<15><12><12>
8259  041670  020040  020040  020040
8260  041676  020040  020040  020040
8261  041704  020040  020040  020040
8262  041712  020040  020040  020040
8263  041720  020040  020040  020040
8264  041726  020040  020040  046454
8265  041734  051501  006513  005012
8266  041742  052123  046101  020114          .ASCII  /STALL                  ST      ,NNNNN (DECIMAL)/<15><12><12>
8267  041750  020040  020040  020040
8268  041756  020040  020040  020040
8269  041764  020040  020040  020040
8270  041772  020040  020040  051440
8271  042000  020124  020040  020040
8272  042006  020040  020040  047054
8273  042014  047116  047116  024040
8274  042022  042504  044503  040515
8275  042030  024514  005015     012
8276  042035     120  044522  052116          .ASCII  /PRINT MESSAGE           PM      ,MESSAGE/<15><12><12>
8277  042042  046440  051505  040523
8278  042050  042507  020040  020040
8279  042056  020040  020040  020040
8280  042064  020040  020040  020040
8281  042072  046520  020040  020040
8282  042100  020040  020040  026040
8283  042106  042515  051523  043501
8284  042114  006505  005012
8285  042120  047125  041111  051525          .ASCII  /UNIBUS INITIALIZE              UI/<15><12><12>
8286  042126  044440  044516  044524
8287  042134  046101  055111  020105
8288  042142  020040  020040  020040
8289  042150  020040  020040  052440
8290  042156  006511  005012
8291  042162  046101  020114  042504          .ASCII  /ALL DECIMAL VALUES MUST BE LESS THAN 65535(10)/<15><12><12><12>
8292  042170  044503  040515  020114
8293  042176  040526  052514  051505
8294  042204  046440  051525  020124
8295  042212  042502  046040  051505
8296  042220  020123  044124  047101
8297  042226  033040  032465  032463
8298  042234  030450  024460  005015
8299  042242  005012
8300  042244  052523  051502  051531          .ASCII  /SUBSYSTEM COMMANDS:/<15><12><12>
8301  042252  042524  020115  047503
8302  042260  046515  047101  051504
8303  042266  006472  005012
8304  042272  047503  046515  047101          .ASCII  /COMMAND              NMENONIC/<15><12><12>
8305  042300  020104  020040  020040
8306  042306  020040  020040  020040
8307  042314  020040  046516  047105
8308  042322  047117  041511  005015
8309  042330     012
```

```
8310  042331     122  040505  020104          .ASCII  /READ DATA              RD/<15><12>
8311  042336  040504  040524  020040
8312  042344  020040  020040  020040
8313  042352  020040  020040  020040
8314  042360  042122  005015
8315  042364  051127  052111  020105          .ASCII  /WRITE DATA             WD/<15><12>
8316  042372  040504  040524  020040
8317  042400  020040  020040  020040
8318  042406  020040  020040  053440
8319  042414  006504     012
8320  042417     127  044522  042524          .ASCII  /WRITE CHECK            WC/<15><12>
8321  042424  041440  042510  045503
8322  042432  020040  020040  020040
8323  042440  020040  020040  020040
8324  042446  041527  005015
8325  042452  051127  052111  020105          .ASCII  /WRITE HEADERS          WH/<15><12>
8326  042460  042510  042101  051105
8327  042466  020123  020040  020040
8328  042474  020040  020040  053440
8329  042502  006510     012
8330  042505     122  040505  020104          .ASCII  /READ HEADER            RH/<15><12>
8331  042512  042510  042101  051105
8332  042520  020040  020040  020040
8333  042526  020040  020040  020040
8334  042534  044122  005015
8335  042540  042523  045505  020040          .ASCII  /SEEK                   SK/<15><12>
8336  042546  020040  020040  020040
8337  042554  020040  020040  020040
8338  042562  020040  020040  051440
8339  042570  006513     012
8340  042573     103  042514  051101          .ASCII  /CLEAR SUBSYSTEM        CS/<15><12>
8341  042600  051440  041125  054523
8342  042606  052123  046505  020040
8343  042614  020040  020040  020040
8344  042622  051503  005015
8345  042626  047503  052116  047522          .ASCII  /CONTROLLER CLEAR       CC/<15><12>
8346  042634  046114  051105  041440
8347  042642  042514  051101  020040
8348  042650  020040  020040  041440
8349  042656  006503     012
8350  042661     104  044522  042526          .ASCII  /DRIVE CLEAR            DC/<15><12>
8351  042666  041440  042514  051101
8352  042674  020040  020040  020040
8353  042702  020040  020040  020040
8354  042710  041504  005015
8355  042714  042522  040503  044514          .ASCII  /RECALIBRATE            RC/<15><12>
8356  042722  051102  052101  020105
8357  042730  020040  020040  020040
8358  042736  020040  020040  051040
8359  042744  006503     012
8360  042747     104  044522  042526          .ASCII  /DRIVE SELECT           DS/<15><12>
8361  042754  051440  046105  041505
8362  042762  020124  020040  020040
8363  042770  020040  020040  020040
8364  042776  051504  005015
8365  043002  040520  045503  040440          .ASCII  /PACK ACK               PA/<15><12>
```

```
8366  043010  045503  020040  020040
8367  043016  020040  020040  020040
8368  043024  02004C  020040  050C40
8369  043032  006501     012
8370  043035     125  046116  040517          .ASCII  /UNLOAD                 UL/<15><12>
8371  043042  020104  020040  020C40
8372  043050  020043  020040  020C40
8373  043056  020040  020040  020040
8374  043064  046125  005015
8375  043070  052123  051101  020124          .ASCII  /START SPINDLE          SS/<15><12>
8376  043076  050123  047111  046104
8377  043104  020105  020040  020C40
8378  043112  020040  020040  051440
8379  043120  006523     012
8380  043123     117  043106  042523          .ASCII  /OFFSET                 OF/<15><12>
8381  043130  020124  020040  020C40
8382  043136  020040  020040  020040
8383  043144  020040  020040  020040
8384  043152  043117  005015
8385  043156  042522  042101  040440          .ASCII  /READ ALL HEADERS       AH/<15><12><12><12>
8386  043164  046114  044040  040505
8387  043172  042504  051522  020040
8388  043200  020040  020040  040440
8389  043206  006510  005012     012
8390  043213     104  052101  020101          .ASCII  /DATA PATTERNS:/<15><12><12>
8391  043220  040520  052124  051105
8392  043226  051516  006472  005012
8393  043234  040520  052124  051105          .ASCII  /PATTERNS "A" THROUGH "I" ARE PROVIDED./<15><12>
8394  043242  051516  021040  021101
8395  043250  052040  051110  052517
8396  043256  044107  021040  021111
8397  043264  040440  042522  050040
8398  043272  047522  044526  042504
8399  043300  027104  005015
8400  043304  042522  042506  020122          .ASCIZ  /REFER TO THE FUNCTIONAL SPEC FOR DETAILS./<15><12><12>
8401  043312  047524  052040  042510
8402  043320  043040  047125  052103
8403  043326  047511  040516  020114
8404  043334  050123  041505  043040
8405  043342  051117  042040  052105
8406  043350  044501  051514  006456
8407  043356  005012     000
8408          000001                  .END
```

```
ACLO  = 000010          1879#
APRES   013727          4628#   4838
A.ABNL  023352          4946*   4973*   5311*   5315*   5819#   6618
A.CONT  023354          5820#   6628
A.NORM  023350          5818#   6622
BADATT  015006          4732#   4757
BADCOM  003063          2217#   2450    3666
BADDEC  002520          2175#   2666    2765    2840    2954    3034    3985
BADDRV  003161          2229#   3000    4208
BADOCT  002646          2190#   2925    2998    3162    3230    3453    3849    3947    4200
BADSEL  003361          2255#   3232    3851
BAI   = 000020          1838#   6787
BAII  = 000004          1705#   3604    3613    4222    5485
BA16  = 000400          1814#
BA17  = 001000          1815#
BDDAT   014504          4696#   5143    5202    5262
BDRTE   007522          2333    3387#
BIT0  = 000001          1606#   1811    1828    1854    1876    2024    2394    4801    4808    5827    6902
BIT00 = 000001          1596#   1606
BIT01 = 000002          1595#   1605
BIT02 = 000004          1594#   1604
BIT03 = 000010          1593#   1603
BIT04 = 000020          1592#   1602
BIT05 = 000040          1591#   1601
BIT06 = 000100          1590#   1600
BIT07 = 000200          1589#   1599
BIT08 = 000400          1588#   1598
BIT09 = 001000          1587#   1597
BIT1  = 000002          1605#   1704    1829    1856    2025    4804    4818    5828
BIT10 = 002000          1586#   1816    1845    1867    1902    1916    1929    1944    1958    2037    5837
BIT11 = 004000          1585#   1817    1846    1868    1887    1903    1917    1930    1945    1959    2038
BIT12 = 010000          1584#   1819    1847    1869    1904    1918    1931    1932    1946    1960    2039
BIT13 = 020000          1583#   1820    1848    1870    1888    1905    1919    1933    1947    1961    2040
BIT14 = 040000          1582#   1821    1849    1871    1889    1906    1920    1934    1948    1962    2041    5838
BIT15 = 100000          1581#   1822    1823    1850    1872    1890    1907    1967    2042    3062    3891    5839
BIT2  = 000004          1604#   1705    1830    1857    1858    1878    2026    4840    4853    5829
BIT3  = 000010          1603#   1706    1836    1837    1859    1879    2027    4843    4859    5830
BIT4  = 000020          1602#   1831    1838    1860    1880    1881    1896    1938    1952    2028    4846    4865    5831
BIT5  = 000040          1601#   1839    1840    1861    1882    1897    1911    1924    1939    1953    2029    4849    4871
                        5832    6699
BIT6  = 000100          1600#   1812    1841    1862    1883    1898    1912    1925    1940    1954    2030    4878    5834
BIT7  = 000200          1599#   1813    1842    1863    1884    1885    1899    1913    1926    1941    1955    2032
BIT8  = 000400          1598#   1814    1843    1864    1865    1886    1900    1914    1927    1942    1956    2033    5835
BIT9  = 001000          1597#   1815    1844    1866    1901    1915    1928    1943    1957    2034    5657    5836
BLDHDR  022242          5565    5651#
BPTVEC= 000014          1613#
BSE   = 000200          1863#   6052    6218    6454
B.BA16= 000001          1828#   6781
B.BA17= 000002          1829#   6781
B.CDT = 000004          1830#   6435    6781
B.CFMT= 000020          1831#   5491    5494    5654    6435    6781    6865
CCLR  = 100000          1823#   4751    4948    6388
CDT   = 002000          1816#   6194    6744    6801    6852    6875
CEFLG   023240          4753    5016*   5018*   5772#
CERR  = 100000          1822#   3504    6090    6149    6208    6390    6441    6756    6812    6900
CERRD   022410          5693#   5789
```

```
CERR1   022457          5700*   5788
CERR10  027121          5752*   5779
CERR11  023217          5767*   5778
CERR12  023217          5768*   5777
CERR13  023217          5769*   5776
CERR14  023217          5770*   5775
CERR15  023167          5759*   5774
CERR16  023217          5771*   5790
CERR2   022534          5708*   5787
CERR3   022564          5712*   5786
CERR4   022621          5717*   5785
CERR5   022674          5725*   5784
CERR6   022754          5733*   5783
CERR7   023217          5764*   5767    5768    5769    5770    5771    5782
CERR8   023005          5738*   5781
CERR9   023047          5744*   5780
CFMT  = 010000          1819*   6194    6744    6801    6852    6875
CHNFLG  001663          1727*   3534    4410*   4457*   4464    4470*
CKSWR = 104406          7775*
CLEAR = 000105          1782*   5542
CLR   = 000040          1839*
CMDTO = 000100          2030*   4771    4788    4959    5951
CNTSTR  001714          1745*   3507*   3529*   4520
COE   = 001000          1366*   6052    6218    6454
COMLEV  004416          2411    2432*   2449    2454    3377    3538
COMMA   004665          2526*   2530
COMPOK  003000          2206*   3648
COMSZE  001254          1712*   3940*   3944    4273*
CONCLR= 000176          1800*   5536    6883
CONERR  016444          4992*   5820
CORTE   010302          2315    3593*
CR    = 000015          1521*   7112    7122
CRLF  = 000200          1522*   7083    7122
CTO   = 004000          1817*   6450
CTRTE   022114          2303    5617*
CUROP   013652          4620*   4795
CVTBUF  006336          3060*   3125    3144
CYLNUM  001200          1681*   4236*   4261
CSBI    011154          2337    3885*
CSCBI   011200          3893*   4189
CSCBII  011202          3892    3895*
CSCBIS  004016          2337*   4187
CSDC    011264          2339    3932*
CSERR   001664          1728*   3602*   3643    3675*
CSPM    010666          2349    3730*
CSRC    010744          2343    3806*
CSRCWS  010776          3805    3808    3811*
CSRW    010726          2345    3803*
CSSC    010762          2341    3809*
CSSF    011570          2335    4130*
CSST    011342          2347    3972*
CSUI    011414          2351    4001*
C.INIT  027052          4553    6661*
C.RTRN  030724          6765    6793    6821    6828    6831    6838    6842    6859    6881    6886    6896    6905    6918
                        6922    6927*
C.SPEC  027744          6701    6797*
```

```
DCK    = 100000          1872*   6052    6218    6454
DCLO   = 000020          1881*
DCMDPR  004510           2446    2477*
DDISP  = 177570          1528*   1662    2373
DDT    = 000400          1886*
DECBIN  031102           2618    2721    2809    2948    3028    3976    7007*
DESL   = 000010          1836*
DI     = 040000          1821*   3504    6238    6910
DISPLA  001142           1662*   2373*   2381*
DISPRE  000174           1628*   2381
DLT    = 100000          1850*   6046    6176    6212    6452
DMC    = 000040          1897*
DMPHDR  003475           2269*   3436
DNRTE   006144           2297    2977*
DONE    001667           1731*   4551*   4556    4571*   4744*   4995*
DPRTE   006356           2313    3064*
DPRTE1  006362           3063    3065*
DRA    = 000001          1876*   4964    6041    6824
DRDY   = 000200          1885*
DRIVE   001176           1680*   2984    2993*   4231*   4252    5480*   5482
DROT   = 000040          1882*
DRPAR  = 000010          1859*
DRPDRV= 002000           2037*   6686
DRVCAL  013570           4551*   5112    5610
DRVDSC= 000040           2029*   4762    6126    6292    6354
DRVERR  015036           4737*   4973    5017    5315    5819
DRVHRD= 000020           2028*   4759    6133    6328    6459
DRVMSK= 000007           1835*   6035    6076    6227    6672
DRVPDT= 000004           2026*   6234    6302
DRVPOS= 000002           2025*   6234    6261    6723
DRVSZD= 010000           2039*   4768    4771    6064    6826
DRVUSE= 000001           2024*   6686
DRY    = 000200          1884*
DR.CLR= 000005           1807*   6114    6277    6319    6343    6457
DR.SEL= 000001           1806*   6184    6263    6578    6585    6592    6599    6803
DSC    = 040000          1889*
DSCNOT  014731           4723*   4767
DSWR   = 177570          1527*   1661    2372
DTBAII= 100000           2042*   5487    6686    6785
DTE    = 010000          1869*   6454
DTYE   = 000040          1861*   6454
EARTE   0C..54           2317    2600*
EBRTE   006346           2311    3062*
ECCW   = 020000          1905*
ECH    = 000100          1862*   6052    6218    6454
EDRTE   005266           2319    2711*
EMTVEC= 000030           1616*
ENDLOC  041416           2389    2391    2395    8229*
EQSGN   002516           2174*   2833    2877    3670
ERRFRE  013644           4571*   5818
ERRVEC= 000004           1609*   2370    2371*   2382*   7675    7676    7682*   7686*   7693*   7694*
ETABL   023242           5011    5774*
E$AH    021574           4047    5554*
E$BI    020536           2337    4462    5331*
E$CC    021500           4029    5536*
E$CS    021464           4031    5533*
```

```
E$DATC  020204          2339    5224*
E$DC    021524          4033    5542*
E$DS    021512          4037    5539*
E$OF    021562          4045    5551*
E$PA    021450          4039    5530*
E$PM    013516          2349    4519*
E$RC    021434          4035    5527*
E$RD    021652          4017    5567*
E$REGC  020026          2343    5171*
E$RH    021606          4025    5557*
E$RW    020426          2345    5289*
E$SC    017610          2341    5104*
E$SK    021620          4027    5560*
E$SS    021550          4043    5548*
E$ST    020446          2347    5305*
E$UI    021424          2351    5504*
E$UL    021536          4041    5545*
E$WC    021736          4021    5583*
E$WD    021722          4019    5580*
E$WH    021632          4023    5563*
E.CCLR= 000001          5827*   6393
E.CERR= 001000          5836*   6467
E.CLAT= 000020          5831*   6119    6283    6324    6348
E.CMTO= 040000          5838*
E.CONT  023356          5001    5015*   5821*   6018*   6056*   6119*   6129*   6200*   6222*   6245*   6251*   6283*   6324*
                        6348*   6393*   6445*   6467*   6606*   6840*   6894*   6902*   6915*
E.DLT = 000400          5835*   6200
E.DPAR= 002000          5837*   6129    6606
E.ILLD= 000100          5834*   6894
E.MDS = 100000          5839*   6018    6445    6840
E.NOAT= 000002          5828*   6245
E.SCLR= 000040          5832*   6915
E.UATT= 000004          5829*   6251
E.UDAT= 000010          5830*   6056    6222
E.UNLD= 020000          2040*   6137    6257    6332    6342
FCMND   014112          4651*   4785
FDATC   014434          4688*   5252
FDRIVE  014103          4649*   4781
FFPRIN  016732          5074*   7780
FMTE  = 000020          1860*   6454
FORMAT  001212          1686*   2914    2921*   4244
FOURP   021756          5579    5586*
FPARM1  014120          4652*   4803
FPARM2  014161          4658*   4806
FPARM3  014205          4662*   4842
FPARM4  014226          4665*   4845
FPARM5  014247          4668*   4848
FPARM6  014270          4671*   4851
FPARM7  014275          4672*   4881
FPARM8  014332          4677*   4889
FPRINT= 104413          4783    4786    4811    4813    4814    4815    4821    4823    4824    4857    4858    4863    4864
                        4869    4870    4875    4884    4890    7780*
FTRTE   006000          2309    2909*
GDDAT   014467          4693*   5140    5199    5259
GNS   = ******  U       1627    7767    7768    7769    7770    7771    7773    7775    7776    7777    7778    7779    7780
GO    = 000001          1811*
```

```
GODRV   022102          5592    5601    5606    5609*
GTSWR = 104405          7773*
HARDER  014646          4714*   4761
HCRC  = 000400          1964*
HDBUFF  001736          1754*   3419    5596
HDR.AD  023374          5861*   6170    6174*   6864*
HDR.CT  023376          5862*   6178*   6867*   6870*
HELPQ   002406          2160*   2412
HPDATA  034436          3335    7782*
HPFILE  002664          2193*   3332
HPRTE   007364          2331    3330*
HPVLD   001672          1734*   2410    2448*   3330
HT    = 000011          1519*   7081    7122
HVRC  = 000400          1865*   6052    6218    6454
H.HEAD  023401          5865*
IBUFPT  001710          1743*   2395*   2396*   2483    2484    2494    2607    2728    2816    2872    3360    3398    3615
                        4318    4352    4451    5227    5569    5578
ICDEC   004704          2435    2558*   2628    3627
ICRTE   006254          2307    3018*
ICTBL   003544          2294*   2563
IDAE  = 002000          1867*
IE    = 000100          1812*   5022    63      6462    6750    6791    6857    6924
ILC   = 000001          1854*   6454
ILF   = 000004          1857*
INTERR  003031          2212*   3664
INTMSV  023405          5870*   6093    6117    6138    6163    6249    6281    6322    6333    6346    6617*   6621*   6625*
                        6674*   6760
INTR    000300          1802*
IOBFSZ  003315          2248*   2405
IOTVEC= 000020          1614*
IR    = 000100          1841*
ISRTE   013012          2305    4410*
ITCNT   001232          1694*   3023    3030*   3507    4520
ITRTE   013020          2301    3536    4411*
IVDADD  002562          2181*   2670
IVDEDT  002540          2178*   2668    2763
IVDLN   002603          2184*   2672    2767    2842
IVDNC   003136          2225*   4205
IVDPAR  002624          2187*   3160    3455    4197    4358
IVDRUN  002764          2204*   3539    4360
IVDWCT  003111          2221*   4203
I.AERR  025434          6269    6275    6293    6302*
I.ATTN  024764          6029    6073    6206*
I.CCLR  026020          6107    6157    6388*   6460    6820    6885
I.CSTS  026452          6100    6535*
I.CSTI  026474          6104    6156    6539*   6809
I.DRV   023406          5874*   6674    6675
I.ERR   024206          6101    6105*
I.ERRA  024166          6100*   6168
I.ERRC  024174          6038    6071    6103*   6236
I.HDAL  024432          6084    6147*
I.INTR  023552          2385    6008*
I.ISRL  023400          5864*   6022    6025*   6028*   6399*   6696*   6850*   6923*
I.ISSU  026102          6115    6185    6264    6278    6320    6344    6430*   6579    6586    6593    6600
I.IOO   024032          6026    6043    6070*
I.RTRN  026002          6020    6058    6068    6096    6108    6113    6116    6122    6131    6140    6143    6158    6160
```

```
                    6186    6189    6198    6202    6224    6240    6247    6253    6265    6279    6286    6298    6318
                    6321    6326    6335    6338    6345    6350    6359#
I.STAT  026556      6112    6317    6576#   6830
I.STOR  026324      6440    6490#
I.UNLD  025702      6259    6342#
JSRR4   001720      1747#   3623
KIPAR0= ****** U    7671
LCNTOF  003441      2264#   3525
LF    = 000012      1520#   7116    7122
LINNUM  001116      1725#   3361*   3510*   3533*   4519*   4777    4895    5105*   5131    5171*   5190    5226*   5245
                    5289*   5305*   5333*   5471*   5505*
LNCNT   001243      1702#   2481*   2502*   2503    2621    2636*   2724    2754*   2812    3362*   3634
LOC2$   010144      3508    3512    3514#
LOC3$   010240      3528    3531#   3540
LOFFST  001256      1713#   4238*   4258
LPCNT1  001722      1748#   3496*   3522*   4522    5052    5056
LPCNT2  001724      1749#   3497*   3523*   5054
LPLABL  003415      2260#   5061
LPRET   010124      3509#   3526    3530
MAXWDS  001700      1739#   2400*   2482    2492    2608    3365    4220    5335    5336    5570
MCLK  = 000400      1900#
MDS   = 001000      1844#   6016    6443    6818
MERD  = 001000      1901#
MESMSK= 000017      1894#
MEWD  = 002000      1902#
MIND  = 000200      1899#
MSKLAB  014571      4705#   5146    5205
MSP   = 000100      1898#
M.CADD= 017760      1970#
M.CDIF= 017760      1969#
M.DRV = 000007      1966#
M.HEAD= 007000      1973#
M.ID  = 000003      1968#
M.PAR = 100000      1967#
M.SECT= 000760      1972#
M.SER = 077770      1971#
NED   = 010000      1847#   6031    6109    6452    6822
NEM   = 004000      1846#   6046    6212    6452
NOCHK = 000400      2033#   4558    5490    6686    6748    6789    6816    6834    6855    6878    6887    6921
NOCK  = 000002      1704#   3604    3609    5488
NODSC = 004000      2038#   4765    6274
NOROOM  002473      2170#   2497    2612
NOSRC   003015      2209#   2844    2888    3662    4362
NTRTE   007410      2325    3354#
NULL    004664      2525#   2528    2615    2626
NXF   = 000004      1858#
OBJSZE  001674      1736#   3369    3616    3636
OBUFPT  001706      1742#   2389*   3402    5228    5334    5337    5451    5585    5658
OCTBIN  030746      2919    2991    3095    3145    3217    3431    3831    3938    4216    6952#
OFFLAG  001665      1729#   4144*   4151*   4164*   4234    4256    4259*
OFFSET= 000115      1786#   5551    6731
OFILE   034434      3359    3511    3614    3638    4312    4349    4445    7781#
OFPTR   001712      1744#   3359*   3370    3647*   4311    4448*   4449*
OFST  = 000004      1878#
ONEP    022072      5529    5532    5535    5538    5541    5544    5547    5550    5607#
OPFLGS  001244      1703#   3604*   3609*   3613*   4222    4243*   4246*   4253
```

```
OPI    = 020000          1870*  6052   6218   6454
OR     = 000200          1842*
OTRTE    012420          2299   4299*
O.WAIT   023360          5841*  5952   5956*  6066*  6070   6086*  6152*  6165*  6180*  6724*  6784*  6846*  6863*
PACK   = 000103          1781*  5530
PARMGV   013753          4632*  4800
PARM0    002142          2072*  4900   4908   4929   5104   5472   5885
PARM1    002230          2105*  4903   4911   4932   5108   5475
PAT    = 000020          1896*
PATPTR   001676          1737*
PATR     001562          1722*  3902   3906   4346   4441   5341
PATRDF   001242          1701*  3908*
PATSEL   001236          1697*  3890   3895*  4461
PATX     001262          1716*  3078   3407   4328   4423   5345   5682
PATXDF   001237          1698*  3079*  4324   4419
PATY     001362          1718*  3083   3411   4333   4429   5349
PATYDF   001240          1699*  3084*
PATZ     001462          1720*  3088   3415   4339   4435   5353
PATZDF   001241          1700*  3089*
PBLKT    023416          5885*  5949   6036   6062   6230   6255   6673*
PBSEL    021270          5471*  5563   5576   5580   5583   5593   5602   5607
PBSVAL=  001000          2034*  5109   6611
PBSW     001670          1732*  4901   4909   4930   5106   5473   5476*
PC     =%000007          1540*  2387*  2401*  2532*  2614*  2618*  2625*  2717*  2721*  2805*  2809*  2909*  2919*
                         2938*  2948*  2979*  2991*  3018*  3028*  3072*  3093*  3095*  3124*  3145*  3208*  3213*
                         3217*  3315*  3317*  3393*  3421*  3431*  3511*  3531*  3603*  3610*  3730*  3817*  3825*
                         3831*  3887*  3903*  3934*  3938*  3972*  3976*  4133*  4167*  4216*  4303*  4553*  4555*
                         4563*  4572*  4790*  4829*  4836*  4837*  4958*  4977*  5014*  5017*  5023*  5024*  5048*
                         5057*  5062*  5112*  5147*  5148*  5206*  5207*  5272*  5273*  5312*  5495*  5563*  5565*
                         5576*  5580*  5583*  5593*  5602*  5607*  5610*  5690*  5957*  5963*  6019*  6057*  6067*
                         6095*  6100*  6104*  6121*  6130*  6142*  6156*  6159*  6188*  6201*  6223*  6246*  6252*
                         6285*  6297*  6325*  6337*  6349*  6357*  6394*  6440*  6446*  6469*  6504*  6548*  6607*
                         6618*  6619*  6622*  6623*  6628*  6629*  6752*  6758*  6764*  6805*  6809*  6827*  6837*
                         6841*  6895*  6904*  6917*  6925*  6934*  6982*  6989*  7044*  7051*  7090*  7097*  7104*
                         7118*  7120*  7292*  7306*  7367*  7412*  7557*  7562*  7662*  7698*
PCA    = 004300          1903*
PCD    = 010000          1904*
PGE    = 002000          1845*  6452
PIP    = 020000          1888*
PIRQ   = 177772          1526*
PIRQVE=  000240          1620*
PLINE    014061          4645*  4779   5133   5192   5247
PLRTE    005500          2323   2798*
PRERR    003217          2236*  4469   5636
PREVOP   013700          4624*  4904
PRINH    001662          1726*
PRLPCT   016670          3531   4829   5052*  5147   5206   5272
PROGID   002316          2150*  2402
PRRTE    006740          2329   3206*
PRTCON=  000064          2068*  4758*  4796   4972*  4979*  5609*
PR0    = 000000          1543*  5316
PR1    = 000040          1544*
PR2    = 000100          1545*
PR3    = 000140          1546*
PR4    = 000200          1547*
PR5    = 000240          1548*  5817
```

```
PR6     = 000300          1549#
PR7     = 000340          1550#  2386   5306
PS      = 177776          1523#  1524   5306*  5316*  5938   5944*  5958*  6667   6668*  6827*
PSW     = 177776          1524#
PTPDB   001734            1753#  4384*  5631*
PTPSR   001732            1752#  4380*  4381   5619*  5624   5632
PTRDB   001730            1751#  4493   4501   5631
PTRSR   001726            1750#  4488*  4489*  4490   4497*  4498   5618*  5620*  5621   5627*  5628
PTRTE   005700            2321   2867#
PUCODE  001174            1679#  4310*  4316*  4320   4350   4415   4446
PUSERR  003203            2233#  4364   5638
PUF.SZ  001250            1709#  4319*  4353   4448   4455   4458
PUNCH   012752            4322   4329   4335   4341   4347   4354   4380#
PWR.EC= 000024            1615#  2366*  2367*  7704*  7705*  7714*  7720*  7732*  7733*
P.BSOF= 000032            2055#  4874   6280*  6281   6314*  6545*  6733*  6734
P.B00   = 000040          2058#  5117   6271*  6272   6288*  6289   6602*  6810*
P.B01   = 000044          2060#  6581*
P.B10   = 000050          2062#  6588*
P.B11   = 000054          2064#  6595*
P.BAHI= 000007            2017#  4817
P.BALO= 000010            2018#  5578*  5585*  5596*  6767   6864
P.BCR   = 000024          2052#  6311*  6541*
P.BDO   = 000042          2059#  6266*  6267   6603*  6811*
P.BD1   = 000046          2061#  6582*
P.BI0   = 000052          2063#  6589*
P.BI1   = 000056          2065#  6187*  6596*
P.CMD= 000001             2011#  5111*  5564*  5577*  5581*  5584*  5597*  5603*  5608*  6082   6699   6703   6715
                          6725   6731   6737   6740   6746   6771   6778   6797   6844   6861   6883   6892
P.CS1   = 000016          2049#  4856   4935   6103*  6155*  6193*  6194*  6196*  6197   6306*  6308*  6535*  6538*
                          6689   6743*  6744*  6746*  6750*  6751   6775*  6778*  6780*  6781*  6791*  6792   6800*
                          6801*  6803*  6804   6806*  6807   6812   6851*  6852*  6854*  6857*  6858   6874*  6875*
                          6877*  6880   6899*  6900   6909*  6910
P.CS1H= 000007            2016#  5491*  5494*  5654   6193   6434   6743   6780   6900   6851   6865   6874
P.CS2   = 000020          2050#  6109   6309*  6539*  6697*  6722   6787*  6788   6799   6818   6822   6848*  6849
                          6873
P.CYLN= 000002            2012#  5587*  5598*  6191   6727   6769   6871
P.DCYL= 000030            2054#  6313*  6547*
P.DRVN= 000000            2010#  5477*  5482*  6079   6256   6432   6671
P.DS    = 000036          2057#  6316*  6543*  6824
P.DTS   = 000026          2053#  4868   6312*  6542*
P.EPAT= 000062            2067#  6105*  6691
P.EPOS= 000060            2066#  6106*
P.ER    = 000034          2056#  4883   6315*  6544*
P.OFST= 000006            2015#  5604*  6733
P.PRST= 000014            2020#  4558   4755   4759   4762   4765   4768   4771   4788   4945*  4959   5109   5484*
                          5487*  5490*  5951*  6064*  6098*  6126*  6133   6137*  6234*  6257   6261*  6274*  6292*
                          6302*  6328   6332*  6342*  6354*  6459*  6611*  6686*  6723*  6747*  6748   6773   6779*
                          6785   6789   6816   6826*  6834   6855   6878   6887   6921
P.SECT= 000004            2013#  5589*  5600*  6192   6728   6770   6872
P.TRCK= 000005            2014#  5588*  5599*
P.WC    = 000012          2019#  5590*  5591*  6768
P.WCR   = 000022          2051#  4862   6310*  6540*
QNEWSW= 000000            1770   3195   3196   3293   3294   3784   3785   5605
Q.INIT= 040000            2041#
RCERR   014601            4707#  5193
RCALHO= 000164            1799#  5554   5594   6082   6861
```

```
RDCHR = 104407          7467    7776#
RDCONT  013374          4424    4430    4436    4442    4459    4497#   4503
RDDATA= 000121          1798#   5577
RDGATE= 100000          1907#
RDHEAD= 000125          1790#   5557    6196    6877
RDLIN = 104410          2433    3154    7777#
RDLOOP  013400          4498#   4499
RDSTAT= 000141          1798#   5111    6797
RDY   = 000200          1813#   4956    6754    6807
READ    013336          4417    4488#
READRG  016120          4790    4925#   5014
RECAL = 000113          1785#   5527    6740
REGLAB  014633          4712#   5196
REGNUM  001224          1691#   3220*   3221    3803    3806
REGVAL  001226          1692#
RELEAS= 000140          1797#   6844
RESREG= 104412          5639    5689    7561    7779#
RESVEC= 000010          1610#
RGDCDE  007054          3213    3250#   3825
RKASOF= 000016          1766#   4942    6092    6162    6242    6497    6545    6734*   6759
RKBA  = 000004          1761#   4939    6493    6541    6767*
RKBAS   023342          3224    3502    4560    4750    4934    5176    5292    5815#   5945    6014    6678
RKCS1 = 000000          1759#   3504    4561    4751*   4936    4948*   4953*   4956    4962    5022*   6037    6089    6147
                        6197*   6206    6388*   6389    6398*   6437*   6438    6462*   6490    6751*   6753    6792*   6804*
                        6806    6836*   6858*   6880*   6889*   6899    6909    6924*
RKCS2 = 000010          1763#   3506*   4937    4947    4950*   5021*   6015    6175    6433*   6491    6539    6722*   6788*
                        6799*   6849*   6873*   6898*
RKDA  = 000006          1762#   4940    6192*   6494    6542    6728*   6770*   6872*
RKDB  = 000024          1769#   6171    6172    6173
RKDC  = 000020          1767#   4941
RKDCYL= 000020          1768#   6191*   6498    6547    6727*   6769*   6871*
RKDS  = 000012          1764#   4944    4964    6040    6495    6543
RKECPS= 000030          1774#   6106    6502
RKECPT= 000032          1776#   6105    6503
RKER  = 000014          1765#   4943    6049    6215    6496    6544
RKMR1 = 000026          1770#   4952*   6183*   6262*   6499    6576*   6583*   6590*   6597*   6698*
RKMR2 = 000034          1771#   4966    6500    6810
RKMR3 = 000036          1772#   4967    6501    6811
RKPAT = 000032          1775#
RKPOS = 000030          1773#
RKPRI   023346          5817#   5944    6668
RKVEC   023344          2384    5816#
RKWC  = 000002          1760#   4938    6492    6540    6768*
RLS   = 000010          1837#   6848
RMASK   001230          1693#
RPSWIT  001671          1733#   4743*   4791    4827    4897    4899*   4907*
RTSPC   001716          1746#   3645
RUEXIT  010274          3500    3513    3537    3539#
RURETN  010270          3535    3538#
RUPTE   010030          2327    3496#   4466
R.ABNL  027004          5957    6067    6142    6159    6337    6357    6617#   6827
R.CONT  027030          6019    6057    6121    6130    6201    6223    6246    6252    6285    6325    6349    6394    6446
                        6469    6607    6625#   6841    6895    6904    6917
R.NORM  027016          6095    6188    6297    6621#   6764    6837    6925
R0    =%000000          1531#   2388*   2390*   2391*   2393*   2394*   2396    2398*   2399*   2400    2403    2601    2607*
                        2608*   2609*   2610    2621*   2622*   2623    2635*   2636    2637*   2648    2651    2654*   2655
```

```
                    2656* 2681* 2712  2728* 2729  2731  2733  2742  2743  2746  2748* 2749  2758
                    2774* 2799  2816* 2817  2819  2821  2824* 2825  2826  2828  2831  2834  2850*
                    2868  2872* 2875  2878  2882  2884  2892* 3066  3090* 3099* 3147* 3170* 3382
                    3435* 3437* 3446* 3461* 3594  3624* 3656* 3668  3671  3812  3815* 3827* 3833*
                    3934  3837* 3841  3842  3845  3855* 3886  3888* 3890* 3891* 3894  3895  3899
                    3900  3902* 3904* 3905* 3906  3910* 3933  3950* 4131  4156* 4170  4176  4180
                    4182  4185  4188* 4195* 4213* 4218  4225  4227  4231* 4232* 4242* 4254* 4262*
                    4265* 4268* 4300  4320* 4328* 4333* 4339* 4346* 4349* 4352* 4369* 4384  4412
                    4415* 4423* 4429* 4435* 4441* 4445* 4449  4451* 4454  4475* 4493* 4495  4501*
                    4738  4780* 4816* 4817* 4835* 4855* 4856* 4861* 4862* 4867* 4868* 4873* 4874*
                    4882* 4883* 4926  4933* 4935* 4936* 4937* 4938* 4939* 4940* 4941* 4942* 4943*
                    4944* 4966* 4967* 4976* 4993  5001* 5005* 5020* 5077  5079  5113* 5116* 5117*
                    5118* 5119  5141  5172* 5175* 5176* 5177  5178* 5181* 5182  5227* 5232  5256
                    5260  5290* 5291* 5292* 5293* 5359* 5362* 5384* 5386* 5392* 5394* 5401* 5403*
                    5408* 5415* 5417* 5421* 5424* 5430* 5432* 5438* 5445* 5449* 5528* 5531* 5534*
                    5537* 5540* 5543* 5546* 5549* 5552* 5555* 5558* 5561* 5567  5569* 5571* 5575*
                    5586* 5609  5652* 5656* 5679* 5683  5684* 5685* 5687* 6013  6107* 6112* 6115*
                    6157* 6185* 6264* 6278* 6317* 6320* 6344* 6359* 6395* 6396* 6400  6401* 646C*
                    6464  6465* 6471* 6472* 6579* 6586* 6593* 6600* 6608* 6609* 6612  6613* 6666
                    6688* 6689* 6693* 6694  6820* 6830* 6885* 6928* 6952  6955* 6958  6980* 6987*
                    7007  701C* 7013  7015  7016  7042* 7049* 7073  7074* 7075  7078* 7135  7145*
                    7149  7165  7166  7179* 7537* 7538  7556* 7560* 7602  7627* 7641  7644* 7647*
                    7651* 7654* 7657  7661* 7673  7677* 7692  7697* 7706  7731* 7751  7752* 7753
                    7754* 7755* 7756* 7757*
R1      =%000001    1532# 2384* 2385* 2386* 2434* 2504  2528  2530  2559  2560  2562* 2602  2615
                    2617  2626  2634  2643  2647* 2656  2657  2680* 2713  2718  2720  2773* 2800
                    2806  2808  2849* 2910  2912  2918  2939  2941  2947  2978  2980  2982  2988
                    2990  3003* 3019  3021  3027  3067  3073  3076  3081  3086  3094  3127  3155*
                    3169* 3209  3211  3216  3250  3254  3258  3260* 3261  3267  3269* 3270  3274
                    3278  3284  3288  3290* 3291  3295  3301  3303* 3304  3310  3389  3394  3396
                    3400  3405  3409  3413  3417  3422  3426* 3428* 3430  3433* 3444* 3460* 3595
                    3605  3607  3611  3615* 3624  3625  3631  3655* 3731  3736  3742  3818  3820
                    3822  3830  3888  3935  3937  3973  3975  4134  4136  4138* 4147  4168  4174
                    4188  4215  4304  4306  4739  4834* 4994  5004* 5007  5009  5011  5019* 5114*
                    5121* 5123  5173* 5180* 5182  5228* 5232  5256  5257  5338* 5339  5341* 5343
                    5345* 5347  5349* 5351  5353* 5355  5357* 5360  5364  5366  5368* 5371  5373*
                    5374  5376  5378* 5379  5381  5383* 5385  5388* 5389* 5391* 5393  5396* 5398
                    5400* 5402  5405* 5407* 5411  5413* 5414* 5416  5419* 5422* 5423  5427  5429*
                    5431  5434* 5436  5437* 5440  5442* 5443  5446  5451* 5568  5570* 5572* 5574*
                    5659* 5670  5672  5675  5682* 5686  6012  6170* 6171* 6172* 6173* 6174  6360*
                    6665  6690* 6691* 6694  6929* 6953  6956* 6966* 6968* 6970* 6973* 6976  6979*
                    6986* 7008  7016* 7018  7020  7022  7032* 7033  7041* 7048* 7136  7149* 7150
                    7154  7178* 7539* 7545* 7551* 7603  7626* 7642  7645* 7648* 7652* 7653* 7655*
                    7656* 7658  7660* 7674  7687* 7688  7689  7691* 7695  7696* 7707  7730*
R2      =%000002    1533# 2437  2438  2440  2441  2563* 2564  2566  2568* 2573  2603  2630  2631
                    2633  2679* 2714  2724* 2725* 2726  2745* 2746  2772* 3068  3097* 3098* 3099
                    3100* 3101* 3109* 3135* 3148* 3149  3168* 3390  3392* 3404* 3420* 3424  3426
                    3434* 3438  3443* 3459* 3502* 3504  3506* 3596  3629  3654* 3734  3735  3839
                    3840  3897  3898  3942  3943  3980  3981  4002  4003  4132  4140* 4142  4145
                    4152* 4153  4155  4157* 4159  4161* 4162  4165  4186  4187* 4194* 4212* 4241
                    4242  4560* 4561  4740  4750* 4751* 4776* 4782* 4810* 4812* 4820* 4822* 4833*
                    4839* 4905* 4927  4934* 4936  4937  4938  4939  4943  4941  4942  4943  4944
                    4947  4948* 4950* 4952* 4953* 4956  4962  4964  4966  4967  4975* 5021* 5022*
                    5074  5115* 5120* 5121  5122  5174* 5179* 5180  5215  5229* 5235* 5337* 5360*
                    5361* 5364* 5385* 5393* 5402* 5416* 5423* 5431* 5436* 5446* 5447  5660* 5663*
                    5664* 5665* 5666* 5667* 5668* 5669* 5671  5673  5676  5678* 5937  5945* 5959*
```

```
                        6011   6014*  6015   6037   6040   6049   6089   6092   6105   6106   6147   6171   6172
                        6173   6175   6183*  6191*  6192*  6197*  6206   6215   6242   6262*  6361*  6388*  6389
                        6398*  6422*  6437*  6438   6462*  6490   6491   6492   6493   6494   6495   6496   6497
                        6498   6499   6500   6501   6502   6503   6539   6540   6541   6542   6543   6544   6545
                        6547   6576*  6583*  6590*  6597*  6664   6678*  6698*  6722*  6727*  6728*  6734*  6751*
                        6753   6759   6767*  6768*  6769*  6770*  6788*  6792*  6799*  6804*  6806   6810   6811
                        6836*  6849*  6858*  6871*  6872*  6873*  6880*  6889*  6898*  6899   6909   6924*  6930*
                        6954   6957*  6967*  6969*  6971*  6977   6978*  6985*  7009   7012*  7015*  7036   7040*
                        7047*  7137   7148*  7152*  7155   7162*  7163*  7164   7169*  7177*  7536*  7539   7540*
                        7546*  7547*  7552*  7553*  7604   7625*  7643   7646*  7649*  7659*  7708   7729*
R3      =%000003        1534#  2477   2482*  2487*  2499*  2509*  2558   2561*  2564   2575*  2604   2620*  2623
                        2651   2659*  2664   2678*  2715   2723*  2726   2729   2771*  2801   2811*  2814   2817
                        2848*  2869   2873*  2884   2891*  3069   3078*  3083*  3088*  3090   3105*  3123*  3129*
                        3133   3137*  3151   3167*  3207   3219*  3220   3221*  3223*  3224*  3225   3235*  3252*
                        3256*  3263*  3265*  3272*  3276*  3280*  3282*  3286*  3293*  3297*  3299*  3306*  3308*
                        3312*  3355   3363*  3366*  3370*  3371*  3375*  3391   3398*  3402*  3407*  3411*  3415*
                        3419*  3441   3458*  3597   3616*  3619*  3623*  3653*  3733   3824   3827   3828*  3838
                        3896   3941   3979   4001   4240   4301   4311*  4312*  4317*  4318*  4319   4321*  4327*
                        4334*  4340*  4345*  4353*  4368*  4385*  4413   4416*  4422*  4428*  4434*  4440*  4454*
                        4455*  4456   4458*  4474*  4496*  4502*  4741   4796*  4797   4801   4804   4808   4818
                        4832*  4840   4843   4846   4849   4853   4859   4865   4871   4879   4928   4947*  4949*
                        4950   4951*  4952   4968   4970*  4974*  5119*  5122*  5123   5177*  5178   5200   5230*
                        5243   5248   5263*  5265   5269   5358*  5361   5369*  5374*  5379*  5443*  5527*  5530*
                        5533*  5536*  5539*  5542*  5545*  5548*  5551*  5554*  5557*  5560*  5594   5597   5603
                        5608   5661*  5670*  5673*  5674*  5677   5683*  5685   5936   5960*  6010   6362*  6663
                        6931*  7138   7146*  7147*  7161*  7164*  7173*  7174*  7176*  7222   7231*  7237*  7238*
                        7241*  7246*  7247*  7248   7257*  7462   7464*  7465   7468*  7469   7476*  7477   7479
                        7487   7491   7493*  7499   7501   7503*  7506*  7544*  7549*  7555*  7556   7605   7624*
                        7709   7728*
R4      =%000004        1535#  2435*  2441*  2446*  2500*  2510   2511*  2570*  2572   2576*  2628*  2673*  2675
                        2682*  2761   2768*  2775*  2838   2845*  2851*  2886   2889*  2893*  2922   2924*  2926*
                        2951   2953*  2955*  2996   3001*  3004*  3031   3033*  3035*  3071   3104*  3106*  3126*
                        3130*  3138   3142*  3157*  3158   3163*  3164*  3171*  3228   3233*  3236*  3333   3334*
                        3451   3456*  3462*  3536*  3598   3617*  3618*  3622*  3625   3627*  3629*  3633*  3634
                        3652*  3657   3658*  3743   3744*  3813   3814*  3835*  3846*  3847   3852*  3853*  3856*
                        3911   3912*  3945   3948*  3951*  3983   3986*  3987*  4004   4005*  4189*  4210*  4214*
                        4274   4322*  4329*  4335*  4341*  4347*  4354*  4356   4365*  4370*  4387   4388*  4389*
                        4390*  4417*  4424*  4430*  4436*  4442*  4459*  4462*  4466*  4467   4471*  4476*  4504
                        4505*  4506*  4507*  4524   4528   4530   4532   4535*  4742   4831*  4880*  4885*  4888*
                        4891*  5113   5114   5115   5130*  5135   5138   5144   5149*  5172   5173   5174   5189*
                        5194   5197   5203   5208*  5229   5276*  5290   5293   5294*  5307   5317*  5331   5338
                        5457*  5477   5478   5480   5483   5485   5488   5492   5506*  5587   5588   5589   5590
                        5598   5599   5600   5604   5605   5611*  5640   5641*  5659   5660   5661   5671*  5672*
                        5674   5935   5961*  6009   6034*  6035*  6075*  6076*  6079   6226*  6227*  6256*  6346
                        6363*  6662   6671*  6672*  6674   6675   6697   6932*  7223   7225*  7226*  7227*  7228
                        7229*  7243   7245*  7253*  7256*  7542*  7545   7551   7553   7606   7623*  7710   7727*
R5      =%000005        1536#  2478   2484*  2485*  2486*  2491*  2495   2498*  2503*  2504*  2505   2507   2508*
                        2605   2606*  2610   2637   2642*  2645*  2646   2648*  2655*  2660   2662   2664*  2677*
                        2716   2742*  2749*  2753   2757*  2758   2770*  2802   2812*  2813*  2814   2819   2847*
                        3070   3125*  3127*  3140*  3141*  3166*  3356   3365*  3367*  3369*  3372*  3374*  3599
                        3614*  3617   3640   3645*  3647   3651*  3733*  3735*  3736*  3739   3741*  3838*  3840*
                        3841*  3842*  3845*  3896*  3898*  3899*  3941*  3943*  3944*  3979*  3981*  3982*  4001*
                        4003*  4240*  4241*  4249*  4252*  4253*  4258*  4261*  4264*  4267*  4270*  4302   4324*
                        4325   4331   4337   4343   4367*  4414   4419*  4420   4426   4432   4438   4473*  4552
                        4558   4755   4758*  4759   4762   4765   4768   4771   4780   4788   4796   4816   4855
                        4861   4867   4873   4882   4900*  4903*  4908*  4911*  4929*  4932*  4933   4945*  4959
```

```
SW09   = 001000            1559*   1569
SW1    = 000002            1577*   5250
SW10   = 002000            1558*   4745    4996    5125    5184    5238
SW11   = 004000            1557*   3527
SW12   = 010000            1556*
SW13   = 020000            1555*   4748    4999    5128    5187    5267
SW14   = 040000            1554*   3514
SW15   = 100000            1553*   5037
SW2    = 000004            1576*   5241
SW3    = 000010            1575*
SW4    = 000020            1574*
SW5    = 000040            1573*
SW6    = 000100            1572*
SW7    = 000200            1571*
SW8    = 000400            1570*
SW9    = 001000            1569*   3518    5040
S.ACLO= 000100             1925*
S.BRHM= 000100             1940*
S.BRKE= 040000             1962*
S.CART= 000400             1942*
S.DCLO= 010000             1932*
S.DIB  = 002000            1958*
S.DOOR= 000200             1941*
S.DRA  = 000040            1911*
S.DROT= 020000             1933*
S.DRY  = 000200            1913*
S.DSC  = 040000            1920*   6124    6272    6289    6352
S.FLT  = 000200            1926*   6267
S.FORM= 001000             1915*
S.FWD  = 002000            1944*
S.HDFL= 000200             1955*
S.HDHM= 000040             1939*
S.ICYL= 000040             1924*
S.ILF  = 000400            1927*
S.LIMD= 020000             1961*
S.LOAD= 010000             1946*
S.MHD  = 000400            1956*
S.NMCY= 010000             1960*
S.OFF  = 002000            1916*
S.PAR  = 001000            1928*   6127    6604
S.PIP  = 020000            1919*   6135    6330
S.PLO  = 004000            1959*
S.REV  = 004000            1945*
S.RTZ  = 020000            1947*
S.SECT= 000020             1952*
S.SKI  = 002000            1929*
S.SPIN= 010000             1918*
S.SPLS= 010000             1931*
S.SPOK= 001000             1943*
S.TYPE= 000400             1914*
S.UNLD= 040000             1948*
S.UNS  = 040000            1934*
S.VV   = 000100            1912*
S.WCLK= 000040             1953*
S.WGAT= 000100             1954*
S.WLE  = 004000            1930*
```

```
S.WRL = 004000           1917#
S.XDOK= 000020           1938#
S.XERR= 001000           1957#
TBITVE= 000014           1611#
TEMP1   001702           1740#  2492*  2493*  2494*  2495   2559*  2560*  2561   3062*  3064*  3091   3636*  3637*
                         3638*  3639*  3640   3803*  3806*  3809*  3815   3837   4134*  4136*  4149   4153   4271
                         5653*  5657*  5669
TEMP2   001704           1741#  3804*  3807*  3810*  3814   3843   5310*  5314   5334*  5335*  5336*  5447
THREEP  022010           5556   5559   5562   5593#
TKVEC = 000060           1618#  7288*  7289*
TORTE   006062           2295   2938#
TOTMSC  003267           2244#  5271
TOVAL   001260           1715#  2943   2950*  3498
TPVEC = 000064           1619#
TRAPVE= 000034           1617#  2364*  2365*  7679   7680*  7685*
TRKNUM  001202           1682#  4264
TRTVEC= 000014           1612#
TWOP    022052           5553   5602#
TYPDS = 104404           2832   2876   2944   3024   3669   4778   5132   5191   5246   7771#
TYPE  = 104400           2402   2405   2412   2432   2450   2452   2453   2497   2612   2666   2668   2670   2672
                         2763   2765   2767   2833   2835   2837   2840   2842   2844   2877   2879   2881   2888
                         2916   2925   2945   2954   2986   2998   3000   3025   3034   3153   3160   3162   3227
                         3230   3232   3332   3335   3436   3440   3448   3450   3453   3455   3525   3539   3648
                         3660   3662   3664   3666   3670   3672   3674   3849   3851   3947   3985   4197   4200
                         4203   4205   4208   4358   4360   4362   4364   4469   4525   4527   4747   4757   4761
                         4764   4767   4770   4772   4775   4779   4781   4784   4785   4787   4795   4800   4803
                         4806   4807   4825   4826   4838   4842   4845   4848   4851   4852   4876   4877   4881
                         4887   4889   4893   4894   4904   4978   4998   5003   5012   5059   5061   5081   5127
                         5133   5134   5137   5140   5143   5146   5186   5192   5193   5196   5199   5202   5205
                         5240   5247   5252   5255   5259   5262   5271   5636   5638   7086   7180   7249   7305
                         7317   7371   7372   7375   7386   7396   7407   7426   7474   7480   7485   7489   7494
                         7495   7497   7500   7504   7734   7767#
TYPOC = 104401           2404   2915   2985   3226   3439   3442   5080   5136   5139   5142   5145   5195   5198
                         5201   5204   5254   5258   5261   5270   7374   7768#
TYPON = 104403           7770#
TYPOS = 104402           7769#
T.ASOF  023320           5802#  6092*  6093   6117   6162*  6163   6242*  6243   6249   6280   6314   6322   6346
                         6497*  6759*  6760
T.BA    023312           5799#  6311   6493*
T.CS1   023304           5794#  6037*  6089*  6090   6103   6114*  6147*  6149   6155   6184*  6206*  6208   6238
                         6263*  6277*  6307*  6308   6319*  6343*  6389*  6390   6430   6434*  6435*  6437   6441
                         6450   6490*  6537*  6538   6578*  6585*  6592*  6599*  6753*  6754   6756
T.CS2   023306           5796#  6015*  6016   6031   6034   6046   6075   6175*  6176   6212   6226   6309   6431*
                         6432*  6433   6443   6452   6491*
T.DA    023314           5800#  6312   6494*
T.DB    023340           5811#
T.DC    023316           5801#  6313   6498*
T.DS    023324           5805#  6040*  6041   6316   6495*
T.ER    023322           5804#  6049*  6052   6215*  6218   6315   6454   6496*
T.MR1   023326           5806#  6499*
T.MR2   023330           5807#  6124   6135   6271   6288   6330   6352   6500*  6581   6588   6595   6602
T.MR3   023332           5808#  6127   6187   6266   6501*  6582   6589   6596   6603   6604
T.PAT   023336           5810#  6503*
T.POS   023334           5809#  6502*
T.WCR   023310           5798#  6310   6492*
UDTSRT  004116           1631   2355#  7308   7391
```

```
UEXATT= 000010           2027*  4755   6098
UFE   = 000400           1843*  6031   6109   6152   6822
UNLOAD= 000107           1783*  5545   6703
UNS   = 010000           1871*
UPE   = 020000           1848*  6046   6212   6452
VLDOBJ  001235           1696*  3358*  3499   3621*  3646*  4308   4453*
VV    = 000100           1883*
WCE   = 040000           1849*  6046   6212   6452
WDCNT   001206           1684*  3428   4270   4273
WLE   = 004000           1868*  6052   6218
WLOOP   013350           4490*  4491
WRDATA= 000123           1789*  5581   6775
WRHEAD= 000127           1791*  5564
WRL   = 004000           1887*
WRTCHK= 000131           1792*  5584   6771
WRTGAT= 040000           1906*
WSTART  013344           4489*  4494
W.DRV   023420           4954*  5307*  5891*  5946*  6061*  6087*  6139*  6154*  6167*  6181*  6229*  6296*  6305*
                         6334*  6356*  6627*  6676*  6739*  6742*  6763*  6815*  6833*
W.MILI  023364           5844*  5941   6907
W.MIN   023372           5860*  6739
W.MTIM  023362           5843*  5939*  5941*
W.SEC   023366           3498*  4954   5857*  6676
W.TIME  023404           4955*  5308*  5866*  5942   5948*  6060*  6088*  6138*  6153*  6166*  6182*  6228*  6295*
                         6304*  6333*  6355*  6626*  6675*  6762*  6814*  6832*
W.WCK = 000200           2032*  6686   6747   6773   6779
W.WTCH  023422           4555   4958   5312   5934*  6752   6758   6805
W.BSEC  023370           5859*  6139   6334   6742
XREAD   013426           4492   4500   4506*
$AUTOB  001134           1658*  7364   7521
$BDADR  001122           1653*
$BDDAT  001126           1655*
$BELL   001164           1673*  4747   4998   5127   5186   5240   7317   7514
$CHARC  031470           7088*  7098*  7105   7114*  7119*
$CKSWR  032444           7350*  7775
$CMTAG  001100           1641*  2357   2358
$CM3  = 000000           1671*
$CNTLC  033424           7305   7386   7514*
$CNTLG  033436           7371   7516*
$CNTLU  033431           7396   7489   7515*
$CRLF   001171           1675*  2837   2881   2916   2945   2986   3025   3227   3448   3450   3674   4527   4784
                         4787   4807   4825   4852   4876   4887   4893   5255   7087   7122   7407   7494   7514
$DBLK   031710           7146   7180   7188*
$DB2D   033466           5057   7535*
$DECVL  033646           7537   7583*
$DTBL   031700           7149   7184*
$ERFLG  001103           1644*  3503*  3516   3520*  5042*
$ERMAX  001115           1650*
$ERRPC  001116           1651*  1725
$ERRTB  002316           2149*
$ERTTL  001112           1648*
$ESCAP  001162           1672*
$FILLC  001156           1669*  7091   7122
$FILLS  001155           1668*  7122
$GDADR  001120           1652*
$GDDAT  001124           1654*
```

```
$GTSWR  032534            7372*   7773
$HD   = 000001            1496    1497
$HINUM  034054            3904    7645    7653    7658*   7663*
$HIOCT  031100            6977*   6990*
$ICNT   001104            1645*
$ILLUP  034344            7704    7720    7737*
$INTAG  001135            1659*   7369*   7388    7408    7521
$ITEMB  001114            1649*
$LF     001172            1676*   4826    4877    4894    7122    7504    7514
$LONUM  034056            3905    7644    7651    7657*   7664*
$LPADR  001106            1646*
$LPERR  001110            1647*   3508*   5043
$LSTAD  034202            2388    7695*   7699*
$MAIL = ****** U          2384    7075
$MNEW   033454            7375    7519*
$MSWR   033443            7372    7517*
$NULL   001154            1667*   7093    7122
$OCNT   032142            7221*   7250*   7263*
$OMODE  032144            7216*   7220*   7225    7228*   7239*   7265*
$PASS   001100            1642*
$POWER  034352            7735    7740*
$PWRDN  034204            2366    7704*   7732
$PWRMG  034340            7735*
$PWRUP  034256            7714    7720*
$QUES   001170            1674*   2453    7122    7426    7497    7514
$RAND   033756            3903    7640*
$RDCHR  033006            7439*   7776
$RDDEC= ****** U          7778
$RDLIN  033076            7462*   7777
$RDOCT= ****** U          7778
$RDSZ = 000072            7455*
$RESRE  033720            7617*   7779
$R2A  = ****** U          7780
$SAVRE  033662            7601*   7778
$SAVR6  034350            7713*   7721    7722*   7723*   7739*
$SETUP= 000114            2355*   2363    2364    2366    2368    7309    7314    7315    7345    7521
$SIZE   034060            2387    7673*
$STUP = 177777            2355*
$SWR  = 167000            1485*   1496    1501    1502    1503    1504    1505    1506    1507    1671    1672    1673    7736
$TIMES  001160            1671*
$TKB    001146            1664*   7269    7290    7301    7326    7354    7381
$TKCNT  032146            7270*   7285*   7315    7332*   7446    7448*
$TKINT  032156            2401    7285*   7306    7367
$TKQEN= 032155            7274*   7340    7451
$TKQIN  032150            7271*   7286*   7287    7338*   7339*   7340    7342*
$TKQOU  032152            7272*   7287*   7449    7450*   7451    7453*
$TKQSR  032154            7273*   7286    7342    7453
$TKS    001144            1663*   7269    7291*   7322*   7324    7330*   7352    7368*   7378    7390*   7410*
$TKSRV  032226            7288    7301*
$TN   = 000001            1496*
$TNPWR  033576            7542    7543    7563*
$TPB    001152            1666*   7111*   7122
$TPFLG  001157            1670*   7069    7122
$TPS    001150            1665*   7109    7122
$TRAP   034362            2364    7751*
$TRP  = 000014            7759*   7768*   7769*   7770*   7771*   7772*   7773    7774*   7775    7776*   7777*   7778*   7779*
```

```
                        7780*   7781*
$TRPAD  034404          7756    7766*
$TSTNM  001102          1643*
$TTYIN  033332          2452    7464    7465    7477    7495    7509    7513*
$TYPBN= ****** U        7772
$TYPDS  031474          7134*   7771
$TYPE   031254          7069*   7759    7767
$TYPEC  031424          7090    7097    7104    7109*   7110    7412
$TYPEX  031472          7115    7117    7120*
$TYPOC  031744          7219*   7768
$TYPON  031760          7218    7221*   7770
$TYPOS  031720          7214*   7769
$$.SM= ****** U         2406    3223    3507    5175    5291    5843    5963    6198    6389    6400    6438    6752    6793
                        6805    6859    6881    6899    6926    6935
$DFILL  032143          7215*   7219*   7229    7264*
.       = 043361        1623*   1627*   1630*   1640*   1677    1707*   1716*   1718*   1720*   1722*   1735*   1754*   2293*
                        2361    4518*   4736*   5103*   5330*   5773*   7122    7188*   7269    7273*   7274    7275*   7513*
                        7514    7521*   7583*   7716    773R    8228*
```

```
COMMEN   1621#
ENDCOM   1621#
ERROR    1515#
ESCAPE   1621#
GETPRI   1484#  1621#  7679
GETSWR   1621#
MULT     1621#
MYTAGS   1633#  1678
NEWTST   1621#
PARMBK      1#  2069   2102
PARMDF      1#  1974
POP      1621#  2676   2769   2846   2890   3002   3165   3234   3374   3457   3650   3854   3910   3949   4211
         4366   4472   4830   4974   5019   5274   7175   7622   7659   7725   7726
PUSH     1621#  2600   2711   2798   2867   2977   3065   3206   3354   3387   3593   3811   3885   3893   3932
         4130   4299   4411   4737   4925   4992   5224   7134   7602   7640   7706   7712
P.DRVR      1#  5791
REGDEF      1#  1756
REPORT   1621#
R.DRIV      1#  5892
R.QDRV      1#
SCOPE    1516#
SETPRI   1484#  1621#  2406   4980   5044   7442
SETTRA   1484#  7759#  7768   7769   7770   7771   7773   7775   7776   7777   7778   7779   7780
SETUP    1621#  2355
SKIP     1621#
SLASH    1621#
SPACE    1621#
STARS    1621#  1636   1677   2278   2292   2353   2413   2429   2456   2475   2513   2524   2534   2556   2578
         2685   2709   2779   2796   2854   2865   2897   2908   2928   2937   2956   3006   3016   3037   3059
         3173   3204   3238   3249   3318   3328   3338   3352   3378   3386   3465   3542   3591   3679   3713
         3715   3727   3746   3801   3858   3883   3914   3930   3953   3970   3989   3999   4007   4014   4050
         4128   4277   4297   4371   4378   4391   4408   4477   4486   4517   4536   4550   4564   4570   4573
         4619   4913   4914   4924   4991   5025   5036   5049   5051   5063   5073   5083   5102   5150   5170
         5209   5223   5277   5288   5295   5304   5318   5329   5458   5470   5496   5503   5507   5526   5612
         5642   5650   5691   5901   5932   5966   6006   6369   6386   6404   6428   6475   6488   6507   6533
         6552   6574   6632   6659   6937   6950   6993   7005   7054   7124   7191   7268   7345   7360   7431
         7455   7524   7586   7631   7667   7702   7718   7745
SWRSU    1621#  2368#
S.DRVE      1#  1908
TRMTRP   7759#
TYPBIN   1621#
TYPDEC   1621#
TYPNAM   1621#
TYPNUM   1621#
TYPOCS   1621#
TYPOCT   1621#  7373
TYPTXT   1621#
$DECBN      1#  6991
$OCTBN      1#  6935
$$CMRE   1634#
$$CMTM   1634#
$$ESCA   1621#
$$NEWT   1621#
$$SET    7759#  7768   7769   7770   7771   7773   7775   7776   7777   7778   7779   7780
$$SKIP   1621#
.EQUAT   1481#  1511
```

```
.HEADE   1481#   1486
.SETUP   1481#   2355
.SWRHI   1481#   1497
.SWRLO   1481#   1507#   1508    1509
.$CATC   1481#   1621
.$CMTA   1482#   1634
.$DB2D   1483#   7522
.$POWE   1482#   7700
.$RAND   1484#   7629
.$READ   1482#   7266
.$SAVE   1483#   7584
.$SIZE   1484#   7665
.$TRAP   1482#   7743
.$TYPD   1483#   7122
.$TYPE   1482#   7052
.$TYPO   1482#   7189
```

```
ADC     3523  7552  7652  7655
ADD     2396  2494  2568  2608  2654  3099  3224  3260  3290  3316  3522  3628  4157  4232  4449
        4455  4817  4856  4862  4868  4874  4883  4935  5117  5118  5176  5298  5335  5336  5685
        6610  6699  6691  6973  6981  7030  7033  7043  7079  7154  7217  7227  7387  7397  7406
        7551  7553  7651  7653  7654  7656
ASL     3098  3223  3637  5025  5116  5175  5291  5388  5405  5434  5663  5654  5665  5665  5667
        5684  6966  6969  6    7026  7028  7029  7420  7421  7422  7647  7755
ASLB    7159
ASR     5396
ASRB    5524  7160
BEQ     5006
BGE     2411  2480  2529  2565  2567  2616  2627  2663  2719  2730  2747  2750  2807  2818  2911
        2940  2981  2989  3020  3074  3128  3134  3139  3150  3152  3210  3335  3425  3445  3500
        3505  3517  3535  3606  3608  3612  3732  3740  3819  3821  3844  3936  3974  4135  4148
        4154  4156  4183  4226  4245  4248  4255  4257  4263  4266  4269  4305  4309  4326  4332
        4338  4344  4351  4382  4421  4427  4433  4439  4465  4491  4494  4439  4559  4746  4749
        4756  4760  4763  4756  4769  4772  4789  4802  4805  4809  4819  4828  4841  4841  4847
        4850  4854  4860  4866  4872  4879  4910  4931  4965  4969  4997  5008  5038  5041  5055
        5075  5107  5124  5126  5183  5185  5236  5239  5251  5266  5387  5395  5404  5418  5425
        5433  5448  5486  5489  5493  5622  5625  5629  5633  5655  5943  6017  6023  6033  6054
        6080  6118  6125  6128  6136  6150  6164  6209  6220  6258  6268  6282  6291  6323  6331
        6347  6353  6392  6442  6444  6625  6700  6749  6755  6761  6774  6786  6790  6808  6813
        6835  6856  6866  6888  6893  6901  6911  6959  6961  7017  7019  7037  7082  7117  7244
        7312  7365  7404  7419  7447  7483
BGT     4181
BHI     6963  7021  7168  7251  7336  7416
BHIS    2624  2732  2815  2820  2995  3635  4230  6710
BIC     2394  2622  2725  2813  4380  4488  4949  5121  5122  5180  5181  5391  5618  5619  5672
        5673  6035  6076  6194  6227  6234  6261  6302  6306  6307  6535  6537  6672  6686  6744
        6747  6750  6779  6791  6801  6852  6857  6875  6972  7241  7302  7327  7337  7355  7382
        7417
BICB    3604  4243  5491  6435  6781
BIS     3062  3506  3891  4820  4950  5407  5487  5490  5657  5668  5669  5674  5951  6018  6056
        6064  6098  6119  6126  6129  6137  6200  6222  6245  6251  6274  6283  6292  6308  6324
        6332  6342  6354  6393  6445  6459  6467  6538  6606  6611  6723  6787  6826  6840  6648
        6894  6902  6915  7162  7163  7246  7247  7424  7555
BISB    2756  3079  3084  3089  3357  3501  3609  3613  3675  4246  4410  4444  4744  4955  4995
        5016  5042  5494
BIT     3504  3514  3518  3527  3739  4222  4381  4490  4498  4558  4745  4748  4755  4759  4762
        4765  4768  4771  4788  4793  4797  4801  4804  4808  4818  4840  4843  4846  4849  4853
        4859  4865  4871  4878  4956  4959  4964  4996  4999  5037  5040  5109  5125  5128  5184
        5187  5238  5241  5250  5267  5621  5624  5628  5632  6016  6031  6041  6046  6052  6090
        6109  6124  6127  6133  6135  6149  6176  6208  6212  6218  6238  6257  6267  6272  6289
        6328  6330  6352  6390  6441  6443  6450  6452  6454  6604  6748  6754  6756  6773  6785
        6789  6807  6812  6816  6818  6822  6824  6834  6855  6878  6887  6900  6910  6921  7024
BITB    5485  5488  5492  5654  6093  6117  6163  6249  6281  6322  6346  6699  6760  6365  7105
BLO     2727  6716  7478
BLOS    2496  2611  3212  3641  3823  4221  4896  5249  6695  7466
BLT     6965  7023  7096  7151  7167  7252  7334  7414  7548
BMI     2632  3102  4383  4492  4500  4963  5479  5623  5626  5630  5634  6024  7158
BNE     2361  2375  2488  2506  2531  2644  2650  2653  2658  2661  2734  2744  2755  2759  2804
        2822  2827  2871  2883  2895  2913  2942  2983  3022  3077  3082  3087  3107  3131  3143
        3251  3255  3259  3262  3268  3271  3275  3279  3285  3289  3292  3296  3302  3305  3311
        3331  3368  3373  3397  3401  3406  3410  3414  3418  3423  3447  3515  3519  3528  3530
        3601  3620  3626  3632  3644  3737  3836  3889  3901  3907  4137  4143  4150  4163  4169
        4171  4175  4177  4219  4223  4228  4235  4272  4307  4315  4386  4447  4503  4521  4523
```

```
        4529  4531  4557  4754  4792  4794  4798  4886  4892  4898  4902  4957  4960  5000  5053
        5110  5129  5188  5233  5242  5244  5268  5340  5344  5348  5352  5356  5363  5367  5372
        5377  5382  5399  5412  5428  5441  5450  5474  5573  5595  5662  5680  5688  5340  5947
        5954  6043  6048  6071  6083  6091  6094  6111  6134  6177  6179  6214  6239  6244  6250
        6273  6329  6451  6453  6455  6458  6726  6732  6738  6741  6757  6772  6798  6817  6819
        6823  6825  6845  6862  6879  6884  6913  6922  7014  7025  7076  7084  7092  7106  7113
        7156  7242  7304  7310  7316  7321  7329  7341  7351  7357  7385  7389  7395  7402  7409
        7452  7470  7472  7488  7492  7502  7559  7650  7724
BPL     2439  3092  4562  5002  5332  6439  7070  7110  7142  7172  7240  7325  7353  7379
BR      2377  2445  2449  2451  2454  2501  2569  2571  2613  2665  2667  2669  2671  2674  2735
        2752  2762  2764  2766  2823  2839  2841  2843  2887  2917  2923  2946  2952  2987  2997
        2999  3026  3032  3063  3080  3085  3103  3132  3136  3156  3159  3161  3215  3229  3231
        3253  3257  3264  3266  3272  3277  3281  3283  3287  3294  3298  3300  3307  3309  3313
        3336  3399  3403  3408  3412  3416  3427  3429  3449  3452  3454  3512  3513  3521  3526
        3540  3642  3661  3663  3665  3676  3805  3808  3829  3848  3850  3892  3946  3984  4158
        4160  4173  4179  4184  4196  4198  4201  4204  4206  4224  4233  4237  4239  4251  4260
        4275  4313  4357  4359  4361  4363  4450  4468  4774  4799  4961  4971  4984  5010  5078
        5237  5264  5313  5350  5354  5365  5370  5375  5380  5390  5397  5406  5409  5420  542E
        5435  5439  5444  5456  5481  5538  5541  5544  5547  5550  5553  5556  5559  5562  5566
        5579  5582  5592  5601  5606  5635  5637  5681  6026  6101  6269  6275  6293  6298  6350
        6447  6729  6735  6776  6866  6890  6896  6905  6918  6974  7035  7072  7089  7099  7108
        7115  7153  7170  7218  7233  7254  7319  7398  7425  7427  7481  7490  7496  7498  7550
        7690  7716  7738
BVS     7031  7034
CLC     2392  2397
CLR     2359  2406  2486  2830  2874  3064  3105  3123  3129  3135  3263  3366  3371  3392  3434
        3435  3496  3497  3510  3533  3618  3667  4166  4776  4810  4822  4839  4905  4945  4951
        4980  5004  5015  5044  5076  5230  5231  5358  5368  5484  5571  5653  5956  6061  6066
        6086  6087  6152  6154  6165  6167  6180  6181  6229  6262  6296  6305  6356  6431  6597
        6627  6693  6698  6724  6763  6815  6833  6836  6889  6956  6957  7011  7012  7145  7148
        7231  7285  7322  7368  7376  7377  7441  7442  7463  7486  7544  7722
CLRB    2448  2481  2489  2757  3358  3361  3362  3503  3520  3602  3621  3741  4250  4259  4452
        4453  4457  4470  4551  4743  4907  5018  5948  6025  6028  6060  6088  6153  6166  6182
        6228  6295  6304  6355  6617  6621  6625  6626  6696  6762  6814  6832  7088  7114  7174
        7493  7503  7560
CMP     2360  2374  2495  2564  2610  2623  2662  2731  2746  2758  2814  2884  2994  3138  3640
        3734  3839  3843  3897  3906  3942  3980  4002  4142  4149  4153  4162  4180  4182  4218
        4220  4225  4229  4244  4271  4350  4446  4520  4895  4968  5007  5123  5182  5232  5248
        5256  5447  5594  5952  6694  7166  7303  7309  7311  7315  7320  7328  7333  7335  7340
        7350  7356  7384  7394  7401  7413  7415  7451  7465  7477  7554
CMPB    2528  2530  2566  2615  2626  2651  2726  2729  2817  2819  2825  2912  2941  2982  2988
        3021  3076  3081  3086  3211  3250  3254  3258  3261  3267  3270  3274  3278  3284  3298
        3291  3295  3301  3304  3310  3396  3400  3405  3409  3413  3417  3607  3625  3634  3820
        3822  3900  4147  4168  4306  5339  5343  5347  5351  5355  5366  5371  5376  5381  5398
        5411  5427  5440  6079  6082  6457  6703  6715  6725  6731  6737  6740  6771  6797  6844
        6861  6883  6892  6960  6962  6964  7013  7018  7020  7022  7081  7083  7091  7112  7116
        7364  7388  7408  7469  7487  7491  7501
COM     3137  4782  4812  5120  5179
COMB    3646  3908  4141  4144  4151  4164  4172  4178  4571  4899  5476
DEC     2487  2562  3106  3130  3140  3142  3148  3367  3372  3444  3446  3529  3619  3835  4138
        4254  4262  4265  4268  4385  4496  4502  4885  4891  5235  5362  5386  5394  5403  5417
        5424  5432  5449  5572  5679  5687  5939  5946  6178  6912  7448  7476  7557
DECB    2748  2754  7095  7098  7239  7250
EMT     1515
HALT    1627  5039  6081  7071  7715  7737
INC     2635  2642  2645  2659  3269  3303  3443  3633  4489  4497  4519  4970  5105  5171  5226
```

```
        5234  5263  5289  5305  5333  5389  5471  5505  5620  5627  5678  7152  7245  7253  7332
        7339  7423  7450  7549  7649  7723
INCB    2502  7118
IOT     1516
JMP     1631  3377  3538  4146  4752  4906  4912  5342  5346  5529  5532  5535  6020  6029  6039
        6058  6068  6073  6084  6096  6122  6131  6140  6143  6160  6168  6199  6198  6202  6224
        6236  6240  6247  6253  6259  6286  6326  6335  6338  6701  6765  6793  6828  6638  6842
        6859  6881  7308  7391
JSR     2387  2401  2435  2441  2446  2614  2618  2625  2628  2717  2721  2805  2809  2909  2919
        2938  2948  2979  2991  3018  3028  3072  3093  3095  3124  3145  3208  3213  3217  3293
        3421  3431  3511  3531  3536  3603  3610  3627  3629  3730  3817  3825  3831  3887  3903
        3934  3938  3972  3976  4133  4167  4189  4216  4303  4322  4329  4335  4341  4347  4354
        4417  4424  4430  4436  4442  4459  4462  4466  4553  4555  4790  4829  4836  4958  5014
        5017  5023  5057  5112  5147  5148  5206  5207  5272  5273  5312  5563  5565  5576  5580
        5583  5593  5602  5607  5610  5957  6019  6057  6067  6095  6100  6104  6107  6112  6115
        6121  6130  6142  6156  6157  6159  6185  6188  6201  6223  6246  6252  6264  6278  6285
        6297  6317  6320  6325  6337  6344  6349  6357  6394  6440  6446  6460  6469  6579  6586
        6593  6600  6607  6618  6622  6628  6752  6758  6764  6805  6809  6820  6827  6830  6837
        6841  6885  6895  6904  6917  6925  7090  7097  7104  7306  7367  7412
MOV     2358  2362  2364  2365  2366  2367  2370  2371  2372  2373  2378  2380  2381  2382  2384
        2385  2386  2388  2389  2395  2400  2403  2407  2434  2477  2478  2482  2483  2484  2491
        2492  2498  2499  2500  2507  2508  2509  2558  2561  2563  2570  2575  2601  2602  2603
        2604  2605  2606  2607  2617  2620  2634  2637  2646  2647  2655  2673  2677  2678  2679
        2680  2681  2712  2713  2714  2715  2716  2720  2723  2728  2742  2745  2753  2768  2770
        2771  2772  2773  2774  2799  2800  2801  2802  2808  2811  2816  2824  2834  2845  2847
        2848  2849  2850  2868  2869  2872  2873  2878  2889  2891  2892  2914  2918  2921  2924
        2943  2947  2950  2953  2978  2984  2990  2993  3001  3003  3023  3027  3030  3033  3066
        3067  3068  3069  3070  3071  3078  3083  3088  3090  3094  3097  3100  3104  3109  3125
        3126  3144  3147  3155  3157  3163  3164  3166  3167  3168  3169  3170  3207  3216  3219
        3220  3221  3225  3233  3235  3252  3256  3265  3272  3276  3280  3282  3286  3293  3297
        3299  3306  3308  3312  3314  3355  3356  3359  3360  3363  3365  3369  3370  3374  3375
        3376  3388  3389  3390  3391  3398  3402  3404  3407  3411  3415  3419  3420  3426  3428
        3430  3433  3437  3438  3441  3456  3458  3459  3460  3461  3498  3502  3507  3508  3509
        3532  3594  3595  3596  3597  3598  3599  3614  3615  3616  3617  3622  3623  3624  3636
        3645  3647  3651  3652  3653  3654  3655  3656  3671  3733  3735  3803  3804  3806  3807
        3809  3810  3812  3813  3814  3815  3824  3827  3828  3830  3833  3837  3838  3840  3841
        3842  3845  3846  3852  3853  3855  3886  3894  3896  3898  3899  3902  3904  3905  3910
        3933  3937  3940  3941  3943  3944  3948  3950  3975  3978  3979  3981  3982  3986  4001
        4003  4131  4132  4140  4152  4159  4161  4185  4186  4187  4194  4195  4210  4212  4213
        4215  4231  4236  4238  4240  4241  4242  4258  4261  4270  4273  4300  4301  4302  4310
        4311  4316  4317  4319  4320  4321  4324  4327  4328  4333  4334  4339  4340  4345  4346
        4349  4352  4353  4365  4367  4368  4369  4389  4412  4413  4414  4415  4416  4419  4422
        4423  4428  4429  4434  4435  4440  4441  4445  4448  4451  4454  4456  4458  4471  4473
        4474  4475  4506  4524  4552  4560  4738  4739  4740  4741  4742  4750  4751  4758  4777
        4780  4796  4816  4831  4832  4833  4834  4835  4855  4861  4867  4873  4880  4882  4888
        4900  4903  4908  4911  4926  4927  4928  4929  4932  4933  4934  4936  4937  4938  4939
        4940  4941  4942  4943  4944  4946  4947  4948  4953  4954  4966  4967  4972  4973  4974
        4975  4976  4979  4981  4993  4994  5001  5011  5019  5020  5021  5022  5043  5045  5056
        5058  5079  5104  5108  5113  5114  5115  5119  5131  5135  5138  5141  5144  5172  5173
        5174  5177  5178  5190  5194  5197  5200  5203  5225  5227  5228  5229  5245  5253  5257
        5260  5269  5275  5290  5293  5306  5307  5310  5311  5314  5315  5316  5334  5337  5338
        5341  5345  5349  5353  5357  5359  5360  5361  5364  5369  5373  5374  5378  5379  5383
        5384  5385  5392  5393  5400  5401  5402  5408  5413  5415  5416  5421  5423  5429  5430
        5431  5436  5437  5438  5442  5443  5445  5446  5451  5472  5475  5528  5531  5534  5537
        5540  5543  5546  5549  5552  5555  5558  5561  5567  5568  5569  5570  5574  5575  5578
        5585  5586  5587  5590  5596  5598  5609  5652  5656  5658  5659  5670  5671  5675  5676
```

```
            5677  5682  5683  5686  5934  5935  5936  5937  5938  5941  5944  5945  5949  5958  5959
            5960  5961  5962  6008  6009  6010  6011  6012  6013  6014  6015  6034  6036  6037  6040
            6049  6062  6070  6075  6089  6092  6103  6105  6106  6139  6147  6155  6162  6170  6171
            6172  6173  6174  6175  6183  6187  6191  6192  6197  6206  6215  6226  6230  6242  6255
            6266  6271  6280  6288  6309  6310  6311  6312  6313  6314  6315  6316  6334  6348  6359
            6360  6361  6362  6363  6364  6388  6389  6395  6398  6430  6433  6437  6462  6471  6490
            6491  6492  6493  6494  6495  6496  6497  6498  6499  6500  6501  6502  6503  6539  6540
            6541  6542  6543  6544  6545  6547  6576  6581  6582  6583  6588  6589  6590  6595  6596
            6602  6603  6608  6661  6662  6663  6664  6665  6666  6667  6668  6669  6671  6673  6676
            6678  6688  6690  6697  6722  6727  6728  6734  6739  6742  6751  6753  6759  6767  6768
            6769  6770  6775  6784  6788  6792  6799  6804  6806  6810  6811  6846  6849  6858  6863
            6864  6867  6870  6871  6872  6873  6880  6898  6899  6907  6909  6924  6927  6928  6929
            6930  6931  6932  6933  6952  6953  6954  6955  6976  6977  6978  6979  6980  6985  6986
            6987  6988  7007  7008  7009  7010  7027  7039  7040  7041  7042  7047  7048  7049  7050
            7073  7074  7078  7093  7135  7136  7137  7138  7139  7140  7141  7146  7149  7169  7175
            7176  7177  7178  7179  7181  7182  7214  7222  7223  7224  7230  7237  7255  7256  7257
            7258  7259  7286  7287  7288  7289  7291  7330  7342  7373  7390  7405  7410  7439  7440
            7443  7453  7462  7464  7475  7506  7507  7508  7509  7536  7537  7538  7539  7540  7541
            7542  7543  7602  7603  7604  7605  7606  7607  7608  7609  7610  7611  7618  7619  7620
            7621  7622  7623  7624  7625  7626  7627  7641  7642  7643  7644  7645  7646  7657  7658
            7659  7660  7661  7673  7674  7675  7676  7677  7679  7680  7682  7683  7685  7686  7687
            7692  7693  7694  7695  7696  7697  7704  7705  7706  7707  7708  7709  7710  7711  7712
            7713  7714  7720  7721  7725  7726  7727  7728  7729  7730  7731  7732  7733  7751  7752
            7756
MOVB        2503  2504  2559  2560  2621  2636  2648  2656  2664  2724  2749  2812  2831  2875  3127
            3141  3668  3736  3888  3890  3895  4134  4136  4188  4249  4252  4253  4264  4267  4384
            4461  4493  4501  4952  5077  5111  5308  5477  5480  5482  5527  5530  5533  5536  5539
            5542  5545  5548  5551  5554  5557  5560  5564  5577  5581  5584  5588  5589  5597  5599
            5600  5603  5604  5608  5631  5660  5661  6114  6138  6184  6193  6196  6256  6263  6277
            6319  6333  6343  6399  6432  6434  6578  6585  6592  6599  6674  6675  6733  6743  6746
            6778  6780  6800  6803  6850  6851  6854  6874  6877  6923  6958  7015  7016  7075  7103
            7111  7144  7147  7161  7164  7173  7215  7216  7219  7220  7221  7225  7228  7229  7248
            7301  7326  7338  7354  7369  7381  7449  7468  7473  7479  7484  7499  7556  7754
NEG         5591  7038  7143  7226
RESET       5504
ROL         5414  5419  6967  6969  6971  7232  7234  7235  7236  7238  7648
ROR         2393  2398  5422
RTI         2379  2408  4982  5046  5082  6365  7080  7183  7260  7331  7343  7411  7444  7454  7510
            7612  7628  7684  7736
RTS         2511  2532  2576  2682  2775  2851  2893  2926  2955  3004  3035  3171  3236  3315  3317
            3334  3462  3658  3744  3856  3912  3951  3987  4005  4214  4370  4388  4390  4476  4505
            4507  4535  4563  4572  4837  4977  5024  5048  5062  5149  5208  5276  5294  5317  5457
            5495  5506  5611  5641  5690  5963  6396  6401  6465  6472  6504  6548  6609  6613  6619
            6623  6629  6934  6982  6989  7044  7051  7120  7292  7562  7662  7698  7757
SBC         7546
SUB         2390  2391  2399  2485  2493  2609  3101  3639  4312  4318  5130  5189  7032  7150  7545
            7547  7691
TRAP        7681  7759  7768  7769  7770  7771  7773  7775  7776  7777  7778  7779  7780
TST         2437  2438  2440  2510  2572  2573  2630  2631  2633  2675  2761  2838  2886  2922  2951
            2996  3031  3091  3133  3149  3151  3158  3228  3333  3424  3451  3657  3743  3834  3847
            3911  3945  3983  4004  4145  4155  4165  4170  4176  4202  4207  4227  4274  4356  4387
            4467  4504  4522  4962  5009  5052  5054  5074  5243  5265  5331  5640  6400  6463  6464
            6470  6612  6914  6920  6975  6984  7036  7046  7077  7085  7107  7155  7165  7243  7290
            7307  7318  7323  7366  7403  7418  7446  7471  7482  7505  7688  7689  7753
TSTB        2410  2479  2505  2643  2657  2660  2718  2733  2743  2803  2806  2821  2826  2828  2[illegible]
            2882  2910  2939  2980  3019  3073  3209  3330  3394  3422  3499  3516  3534  3600  [illegible]5
```

```
            3611  3631  3643  3731  3742  3818  3935  3973  4174  4234  4247  4256  4304  4308  4314
            4325  4331  4337  4343  4420  4426  4432  4438  4464  4495  4528  4530  4532  4556  4561
            4753  4791  4827  4897  4901  4909  4930  5106  5473  5478  5483  5605  5942  6022  6243
            6438  7069  7109  7157  7171  7324  7352  7378
.ASCII      1674  1675  2150  2294  2296  2298  2300  2302  2304  2306  2308  2310  2312  2314  2316
            2318  2320  2322  2324  2326  2328  2330  2332  2334  2336  2338  2340  2342  2344  2346
            2348  2350  2352  2526  4016  4018  4020  4022  4024  4026  4028  4030  4032  4034  4036
            4038  4040  4042  4044  4046  4677  7782  7787  7796  7805  7817  7827  7838  7847  7856
            7862  7868  7874  7885  7895  7907  7917  7928  7938  7948  7959  7970  7980  7992  8001
            8010  8019  8025  8034  8040  8046  8057  8063  8073  8083  8091  8096  8105  8115  8124
            8133  8141  8150  8159  8169  8179  8189  8199  8209  8218  8229  8238  8248  8258  8266
            8276  8285  8291  8300  8304  8310  8315  8320  8325  8330  8335  8340  8345  8350  8355
            8360  8365  8370  8375  8380  8385  8390  8393
.ASCIZ      1673  1676  2156  2160  2165  2166  2170  2174  2175  2178  2181  2184  2187  2190  2193
            2204  2206  2209  2212  2217  2221  2225  2229  2233  2236  2239  2244  2248  2255  2260
            2264  2269  2274  2276  4620  4624  4628  4632  4636  4640  4645  4649  4651  4652  4658
            4662  4665  4668  4671  4672  4685  4688  4693  4696  4698  4702  4705  4707  4712  4714
            4718  4723  4728  4732  5693  5700  5708  5712  5717  5725  5733  5738  5744  5752  5759
            5764  7514  7515  7516  7517  7519  7740  8400
.BLKB       7273  7513  7583
.BLKW       1716  1718  1720  1722  1754  7188
.BYTE       1643  1644  1649  1650  1658  1659  1667  1668  1669  1670  1695  1696  1697  1698  1699
            1700  1701  1702  1703  1726  1727  1728  1729  1730  1731  1732  1733  1734  2072  2073
            2075  2076  2077  2078  2105  2106  2108  2109  2110  2111  2525  5772  5864  5865  5866
            5970  5874  5875  5876  5877  5878  5879  5880  5881  7261  7262  7263  7264  7511  7512
.DSABL      7428
.ENABL      1479  7269
.END        8408
.ENDC       1491  1504  1506  1507  1515  1607  1621  1630  1637  1641  1643  1671  1672  1673  1674
            1678  1771  2073  2106  2135  2279  2293  2354  2355  2362  2363  2364  2366  2368  2384
            2407  2414  2430  2457  2476  2514  2525  2535  2557  2579  2686  2710  2780  2797  2855
            2866  2898  2909  2929  2938  2957  3007  3017  3038  3060  3174  3195  3197  3205  3223
            3239  3250  3294  3319  3329  3339  3353  3379  3387  3466  3543  3592  3680  3714  3716
            3728  3747  3784  3786  3802  3859  3884  3915  3931  3954  3971  3990  4000  4008  4015
            4051  4129  4278  4298  4372  4379  4392  4409  4478  4487  4518  4537  4551  4565  4571
            4574  4620  4914  4915  4925  4981  4992  5026  5037  5045  5050  5052  5064  5074  5084
            5103  5151  5171  5175  5178  5210  5224  5278  5289  5291  5294  5296  5305  5319  5330
            5459  5471  5497  5504  5508  5527  5605  5613  5643  5651  5692  5861  5892  5893  5902
            5933  5958  5967  6007  6370  6387  6405  6429  6476  6489  6508  6534  6553  6575  6633
            6660  6935  6938  6951  6994  7006  7055  7075  7125  7192  7269  7300  7309  7313  7344
            7346  7361  7392  7428  7432  7443  7455  7456  7464  7466  7469  7497  7514  7515  7521
            7525  7587  7632  7668  7679  7683  7686  7700  7703  7712  7713  7719  7725  7726  7736
            7743  7746  7752  7755  7767  7768  7769  7770  7771  7772  7773  7774  7775  7776  7777
            7778  7779  7780
.EQUIV      1515  1516  1524  1539  1540  1569  1570  1571  1572  1573  1574  1575  1576  1577  1578
            1597  1598  1599  1600  1601  1602  1603  1604  1605  1606  1725  5767  5768  5769  5770
            5771
.EVEN       1707  1735  2293  4015  4736  5773  7275  7521  7742  8228
.IF         1487  1504  1505  1506  1507  1513  1579  1607  1630  1636  1640  1642  1671  1672  1673
            1677  1678  1770  2069  2102  2278  2292  2353  2355  2357  2362  2364  2366  2368  2384
            2406  2413  2429  2456  2475  2513  2524  2534  2556  2578  2685  2709  2779  2796  2854
            2865  2897  2908  2928  2937  2956  3006  3016  3037  3059  3173  3195  3196  3204  3223
            3238  3249  3293  3294  3318  3328  3338  3352  3378  3386  3465  3542  3591  3679  3713
            3715  3727  3746  3784  3785  3801  3858  3883  3914  3930  3953  3970  3989  3999  4007
            4014  4050  4128  4277  4297  4371  4378  4391  4408  4477  4486  4517  4536  4550  4564
            4570  4573  4619  4913  4914  4924  4980  4991  5025  5036  5044  5049  5051  5063  5073
```

```
        5083  5102  5150  5170  5175  5209  5223  5277  5288  5291  5295  5304  5318  5329  5458
        5470  5496  5503  5507  5526  5605  5612  5642  5650  5691  5843  5867  5892  5901  5932
        5942  5966  6006  6019  6369  6386  6404  6428  6475  6488  6507  6533  6552  6574  6632
        6659  6937  6950  6993  7005  7054  7075  7124  7191  7268  7270  7298  7303  7309  7345
        7346  7360  7384  7431  7432  7442  7455  7463  7465  7469  7470  7513  7514  7521  7524
        7586  7631  7667  7671  7679  7682  7686  7702  7712  7713  7718  7725  7726  7734  7736
        7740  7745  7751  7755  7759  7768  7769  7770  7771  7772  7773  7775  7776  7777  7778
        7779  7780
.IFF     1504  1506  1507  1513  1637  1640  1642  1671  1678  1770  2073  2106  2279  2293  2354
        2362  2407  2414  2430  2457  2476  2514  2525  2535  2557  2579  2685  2710  2780  2797
        2855  2866  2898  2909  2929  2938  2957  3007  3017  3038  3060  3174  3205  3239  3259
        3319  3329  3339  3353  3379  3387  3466  3543  3592  3680  3714  3716  3728  3747  3802
        3859  3884  3915  3931  3954  3971  3990  4000  4008  4015  4051  4129  4278  4298  4372
        4379  4392  4409  4478  4487  4518  4537  4551  4565  4571  4574  4620  4914  4915  4925
        4981  4992  5026  5037  5045  5050  5052  5064  5074  5084  5103  5151  5171  5210  5224
        5278  5289  5296  5305  5319  5330  5459  5471  5497  5504  5508  5527  5613  5643  5651
        5692  5843  5892  5893  5902  5933  5951  5958  5967  6007  6020  6037  6058  6064  6068
        6079  6089  6094  6096  6118  6122  6131  6140  6143  6155  6160  6164  6168  6183  6189
        6202  6224  6231  6240  6247  6257  6282  6286  6299  6306  6323  6326  6335  6338  6350
        6358  6370  6387  6395  6405  6429  6447  6470  6476  6489  6508  6534  6553  6575  6608
        6633  6660  6677  6740  6743  6766  6818  6828  6838  6842  6896  6905  6918  6926  6935
        6938  6951  6994  7006  7055  7125  7192  7269  7346  7361  7371  7432  7435  7443  7455
        7456  7465  7497  7513  7525  7587  7632  7668  7671  7682  7686  7703  7719  7736  7746
        7752
.IFT     5957  6036  6057  6060  6067  6087  6093  6095  6117  6121  6130  6138  6142  6153  6159
        6163  6166  6181  6188  6201  6223  6228  6246  6249  6281  6285  6295  6304  6322  6325
        6333  6337  6349  6355  6394  6446  6469  6607  6615  6673  6739  6742  6748  6789  6814
        6826  6832  6841  6855  6878  6887  6895  6904  6917  6925  7434  7439  7675  7686  7696
        7700
.IFTF    5951  6020  6037  6058  6064  6068  6079  6089  6094  6096  6118  6122  6131  6140  6143
        6155  6160  6164  6168  6183  6189  6202  6224  6231  6240  6247  6257  6282  6286  6299
        6306  6323  6326  6335  6338  6350  6358  6395  6447  6470  6608  6630  6677  6740  6743
        6766  6792  6818  6828  6838  6842  6858  6880  6890  6896  6905  6918  6926  7371  7432
        7435  7672  7675  7692  7696
.IIF     1486  1491  1496  1497  1501  1502  1503  1504  1507  1508  1509  1627  1677  2363  2406
        3507  5963  6198  6389  6400  6438  6752  6793  6805  6859  6881  6899  6926  6935  7122
        7269  7275  7314  7315  7374  7505  7514  7521  7767  7768  7769  7770  7771  7773  7775
        7776  7777  7778  7779  7780
.IRP     1678  2355  2601  2677  2712  2770  2799  2847  2868  2891  2978  3003  3066  3166  3207
        3235  3355  3374  3388  3458  3594  3651  3812  3855  3886  3894  3910  3933  3950  4131
        4212  4300  4367  4412  4473  4738  4831  4926  4974  4993  5019  5225  5275  7135  7175
        7602  7622  7641  7659  7705  7712  7725  7726
.LIST       1  1478  1621  1627  1671  2355  2368  7455  7759  7767  7768  7769  7770  7771  7772
        7773  7774  7775  7776  7777  7778  7779  7780  7781
.MACRO      1  1507  1633  1634  7759
.MCALL   1481  1482  1483  1484  1621  2368
.NLIST      1  1477  1621  1627  1671  2355  2368  7455  7759  7767  7768  7769  7770  7771  7772
        7773  7774  7775  7776  7777  7778  7779  7780  7781
.PAGE    1634  1974  2135  5892  5964  6367  6402  6473  6505  6550  6615  6630
.REM        1
.PEPT    1627
.SBTTL   1497  1511  1621  1634  1678  1723  1757  1778  1809  1833  1852  1874  1892  1909  1922
        1936  1950  1964  1974  2005  2022  2044  2070  2103  2135  2279  2356  2414  2457  2514
        2535  2579  2686  2780  2855  2898  2929  2957  3007  3038  3174  3239  3319  3339  3379
        3466  3543  3680  3716  3747  3859  3915  3954  3990  4008  4051  4278  4372  4392  4478
        4537  4565  4574  4915  4985  5026  5050  5064  5084  5151  5210  5278  5296  5319  5459
```

```
            5497    5508    5613    5643    5692    5792    5813    5868    5883    5889    5892    5899    5964    6145    6204
            6300    6340    6367    6402    6473    6505    6550    6615    6630    6795    6935    6391    7052    7122    7189
            7266    7522    7584    7629    7665    7700    7743    7759
.TITLE      1486
.WORD       1627    1628    1629    1642    1645    1646    1647    1648    1651    1652    1653    1654    1655    1656    1657
            1660    1661    1662    1679    1680    1681    1682    1683    1684    1685    1686    1697    1688    1689    1690
            1691    1692    1693    1694    1708    1709    1711    1712    1713    1715    1736    1737    1739    1740    1741
            1742    1743    1744    1745    1746    1747    1748    1749    1750    1751    1752    1753    2074    2079    2080
            2081    2082    2083    2084    2085    2086    2087    2088    2089    2090    2091    2092    2093    2094    2095
            2096    2097    2098    2099    2100    2101    2107    2112    2113    2114    2115    2116    2117    2118    2119
            2120    2121    2122    2123    2124    2125    2126    2127    2128    2129    2130    2131    2132    2133    2134
            2295    2297    2299    2301    2303    2305    2307    2309    2311    2313    2315    2317    2319    2321    2323
            2325    2327    2329    2331    2333    2335    2337    2339    2341    2343    2345    2347    2349    2351    2836
            2880    3060    3673    4017    4019    4021    4023    4025    4027    4029    4031    4033    4035    4037    4039
            4041    4043    4045    4047    4048    4193    4463    4526    4554    5013    5060    5774    5775    5776    5777
            5778    5779    5780    5781    5782    5783    5784    5785    5786    5787    5788    5789    5790    5794    5796
            5798    5799    5800    5801    5802    5804    5805    5806    5807    5808    5809    5810    5811    5815    5816
            5817    5821    5841    5843    5844    5857    5859    5860    5861    5862    5891    6990    7119    7265    7270
            7271    7272    7558    7663    7664    7699    7735    7781

ERRORS DETECTED:  0
DEFAULT GLOBALS GENERATED:  0

*,DZR6RB.SEQ/SOL/CRF/NL:TOC=DRIVE8.P11/EQ:QNEWSW,DZR6RB.CMB
RUN-TIME:  85 77 11 SECONDS
RUN-TIME RATIO:  629/175=3.5
CORE USED:  43K  (85 PAGES)
```

Spooler runtime 28 Seconds, 118 KCS, 843 disk reads, 3 disk writes, 202 pages