

This microfiche card contains a grid of frames, likely representing a test procedure or data log. The frames are arranged in approximately 10 rows and 6 columns. Each frame contains text, possibly including test results, component identifiers, or timing information. The text is too small to read clearly but appears to be organized in a structured manner. The right half of the card is mostly blank, with some faint markings and a small grid of frames in the bottom right corner.

B01

EOF1DZTEFASQ

00010000

770608

PDP10 411

HDR1DZTEBASEQ

00010000

770608

.REM %

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZTEB-A-D
PRODUCT NAME: TMO3/TE16 CONTROL LOGIC TEST PART II
DATE CREATED: 21 FEB 77
MAINTAINER: DIAGNOSTIC ENGINEERING
AUTHOR: J. G. ADAMS

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

THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED TO THE PURCHASER UNDER A LICENSE FOR USE ON A SINGLE COMPUTER SYSTEM AND CAN BE COPIED (WITH INCLUSION OF DIGITAL'S COPYRIGHT NOTICE) ONLY FOR USE IN SUCH SYSTEM, EXCEPT AS MAY OTHERWISE BE PROVIDED IN WRITING BY DIGITAL.

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

COPYRIGHT (c) 1977, BY DIGITAL EQUIPMENT CORPORATION

TABLE OF CONTENTS

PARAGRAPH	SUBJECT	PAGE
1.	ABSTRACT	3
2.	REQUIREMENTS	3
3.	LOADING PROCEDURE	3
4.	STARTING PROCEDURE	3
5.	SWITCH SETTINGS	3
6.	ERROR PRINTOUTS	8
7.	OPERATION	8
8.	SUBTEST SUMMARIES	8
9.	LISTING	15

1. ABSTRACT

THIS PROGRAM IS DESIGNED TO SEQUENTIALLY TEST THE DATA FORMATTING FUNCTIONALITY OF THE TMO3 FORMATTER. EACH TEST WILL ATTEMPT TO ISOLATE FAILURES TO THE MODULE LEVEL AND PROVIDE PRINTOUT INFORMATION WHICH WILL IDENTIFY THE FAILING MODULE. THE LEVEL OF FAULT ISOLATION IS POSSIBLE BECAUSE OF TMO3 THE STRUCTURE AND ITS MAINTAINENCE MODES.

2. REQUIREMENTS (HARDWARE)

- A. ANY PDP-11 PROCESSOR
- B. BK OF CORE
- C. CONSOLE TTY
- D. TMO3 MAGTAPE CONTROLLER
- E. MASSBUS CONTROLLER (RH)
- F. TE16 MAGTAPE TRANSPORT

3. LOADING PROCEDURE

USE STANDARD PROCEDURE FOR LOADING BINARY PAPER TAPE.

4. STARTING PROCEDURE

THERE ARE TWO (2) STARTING ADDRESSES THAT MAY BE USED:
200(8) AND 210(8).

- A. 200(8): STARTING AT THIS ADDRESS WILL CAUSE A PROGRAM IDENTIFICATION HEADER TO BE PRINTED BEFORE TESTING IS BEGUN.
- B. 210(8): STARTING AT THIS ADDRESS WILL NOT PRINT THE IDENTIFICATION HEADER AND IS THEREFORE GENERALLY TO BE USED FOR RESTARTS RATHER THAN INITIAL START

** NOTE: SEE ALSO SEC 5. CONSOLE SWITCH SETTINGS
** TYPE 1C TO RESTART PROGRAM (2200)

4.1 AUTOMATIC MODE OPERATION

IF THIS PROGRAM IS LOADED AND RUN IN AUTOMATIC (CHAIN) MODES
 DEFAULT RESPONSES TO OPERATOR REQUESTS ARE USED. THE SOFTWARE
 SWR IS INVOKED WITH A SWITCH SETTING OF 10000 (HALT ON ERROR)
 IF IN ACT11 CHAIN MODE. NO OPERATOR INTERVENTION IS REQUIRED.

*EXCEPTION: IF THIS PROGRAM IS LOADED VIA TMDP CHAIN MODE THE
 PROGRAM WILL NOT TEST TMD3 DRIVE #0, TE16 SLAVE #0.

4.2 SAMPLE START AT 200

NOTE: DEFAULT RESPONSES ARE SHOWN IN ANGLE BRACKETS <>
 OPERATOR RESPONSES ARE SHOWN IN PARENTHESES (), AND
 MEMORY LOCATIONS CONTAINING THE DEFAULT ARE SHOWN IN
 SQUARE BRACKETS [].

PARAMETER REQUEST: <DEFAULT> (RESPONSE) [LOCATION:]

TMD3-TE16 CONTROL LOGIC TEST PART II (DZTEB-A)
 ASSURE TAPE IS AT BOT
 TYPE ^C TO RESTART

REGISTER START: <172440> (CR)	[REGS:]
VECTOR ADDRESS: <224> (CR)	[VECT:]
TMD3 DRIVE: <0> (CR)	[DRVN:]
TE16 SLAVE: <0> (CR)	[SLVN:]
IF THE SOFTWARE SWR IS INVOKED:	
SWR = <00000> NEW = (CR)	[SWREG:]

5. CONSOLE SWITCH SETTINGS

CONTROL:

- 1) CONTROL G (↑G):
SELECTS THE SOFTWARE SWR AND ALLOWS NEW SWITCH SETTINGS.

THE MACHINE WILL THEN TYPE: SWR=XXXXXXNEW=
WHERE: XXXXXX IS THE OCTAL CONTENTS OF THE SOFTWARE SWR.
AFTER 'NEW=' HAS BEEN TYPED THEN THE OPERATOR CAN DO ONE
OF THE FOLLOWING AT THE TTY:
A) TYPE AN OCTAL NUMBER TO BE LOADED INTO THE SOFTWARE SWR.
B) IF A (CR) IS THE FIRST KEY DEPRESSED THE SOFTWARE SWITCH
REGISTER CONTENTS WILL NOT BE CHANGED.
- 2) CONTROL A (↑A):
ALTERNATES THE SWITCH REGISTER FROM HARDWARE TO SOFTWARE & VICE VERSA.
- 3) CONTROL C (↑C):
RESTART PROGRAM AT 200
- 4) CONTROL U (↑U):
DELETES ALL CHARACTERS TYPED IN RESPONSE TO A REQUEST.

ALL SWITCHES ARE USED (0-15) AND THE NORMAL, OR DEFAULT, RUN
IS DONE WITH ALL SWITCHES SET TO ZERO (0).
ALL SWITCHES ARE DYNAMIC AND MAY BE CHANGED AT ANY TIME.

SW15: 1=HALT ON ERROR
0=CONTINUE
SW14: 1=LOOP ON ERROR (SCOPE)
0=CONTINUE
SW13: 1=DO NOT PRINT ERRORS
0=PRINT ALL ERRORS
SW12: 1=DO CONTINUOUS CYCLE
0=HALT AT END OF PASS
SW11: 1=INHIBIT ITERATIONS
0=ITERATE EACH TEST ITS ASSIGNED AMOUNT
SW10: 1=HALT AT END OF CURRENT TEST
0=CONTINUE TO NEXT TEST
SW8: 1=INHIBIT WRAP AROUND DATA CHECK
0=DO DATA CHECKS
SW7: 1=INHIBIT WRAP AROUND STATUS CHECK
0=DO STATUS CHECK
SW6: 1=SELECTABLE WRAP DATA PATTERN (IN SINGLE TEST)
0=AUTO PATTERN
SW5-0: SELECT INDIVIDUAL TEST ** 00=DO ALL TESTS

6. ERROR PRINTOUTS

ERROR PRINTOUTS WILL APPEAR IN TWO FORMS, ONE FOR THE CONTROL LOGIC TESTS AND ANOTHER FOR THE DATA TESTS.

CONTROL LOGIC PRINTOUTS WILL CONTAIN A HEADER WHICH CALLS OUT THE TEST NUMBER, FUNCTION BEING TESTED, AND THE SUSPECT MODULE, OR MODULES ON THE FIRST LINE. THE SECOND LINE WILL CONTAIN INFORMATION AS TO THE ACTUAL ERROR. BOTH THE EXPECTED RESULT AND THE ACTUAL RESULT OF THE TEST WILL BE GIVEN. LINE THREE WILL SHOW THE CONTENTS OF THE MAJOR REGISTERS AT THE TIME OF THE ERROR AND LINE FOUR WILL PRINT THE ITERATION NUMBER WHEN APPLICABLE.

DATA TESTS WILL PRINT A HEADER CONTAINING THE TEST NUMBER, AND A DESCRIPTION OF THE WRAP AROUND FUNCTION UNDER TEST. FOLLOWING THE HEADER WILL BE A LIST OF THE MAJOR REGISTERS WITH THE EXPECTED AND ACTUAL VALUES. ANY BAD DATA WILL BE PRINTED (PER CHARACTER) FOLLOWING THE REGISTER INFORMATION OR FOLLOWING THE HEADER IF NO STATUS ERRORS WERE ENCOUNTERED.

5. THE FOLOWING ARE TWO EXAMPLES OF ERRORS DETECTED BY THE WRAP AROUND DATA TESTS. NOTE THAT EACH WRAP AROUND TEST MAY BE ACCOMPANIED BY EITHER A STATUS ERROR OF A DATA ERROR OR BOTH.

LOGIC TEST 1: WRAP 3, NRZ, NORMAL, ODD
BAD STATUS
CS1 EXPT 004270 RCVD 144270
CS2 EXPT 000100 RCVD 000100
DS EXPT 010600 RCVD 150600
ER EXPT 000000 RCVD 000100

THIS MESSAGE INDICATES BAD STATUS OF VPE (BIT 6 OF ER)

LOGIC TEST 3: WRAP 2, NRZ, NORMAL, ODD
BAD DATA
CN:0
G: 11111111
B: 11111011
CN:10
G: 00000000
B: 00001000

THIS MESSAGE SHOWS THAT DATA RECEIVED WAS NOT AS EXPECTED. CHARACTER ZERO (CN: 0) SHOWS THAT BIT TWO (2) WAS DROPPED, WHILE CHARACTER TEN (CN: 10) SHOWS BIT THREE (3) HAS BEEN PICKED UP
G: = EXPECTED DATA (GOOD)
B: = ACTUAL DATA (BAD)

7. OPERATION

THE PROCEDURES FOR OPERATING THIS PROGRAM ARE QUITE SIMPLE AND REQUIRE ONLY A FEW STEPS:

1. LOAD ADDRESS 200 OR 210
2. SET SWITCHES FOR DESIRED TEST CYCLE
3. PRESS START

ALL CONSOLE SWITCHES ARE DYNAMIC AND MAY BE CHANGED AT ANY TIME. THE NORMAL OPERATING SEQUENCE IS ALL SWITCHES DOWN (0). THE TEST WILL TAKE APPROXIMATELY 3 MINUTES TO RUN; HOWEVER, IF ITERATIONS ARE INHIBITED (SW11=1) THE TEST WILL RUN IN ABOUT 30 SECONDS. THE END OF PASS IS NOTED BY A PRINTOUT STATING END OF PASS, AND THE NUMBER OF THAT PASS.

SINGLE TEST SELECTION: (SMD-SM5)

WHEN SMD-SM5 ARE SET TO ZERO (00), THE SCHEDULAR WILL EXECUTE ALL TESTS IN SEQUENCE. IF SMD-SM5 ARE SET TO SOME SPECIFIC TEST NUMBER THEN THAT PARTICULAR TEST ONLY WILL BE EXECUTED UNTIL THE TEST SELECT NUMBER IS CHANGED. WHEN YOU WISH TO SELECT A PARTICULAR TEST, SET SW10 TO A ONE (1) IN ORDER TO STOP AT THE END OF THE CURRENT TEST BEFORE SELECTING A DIFFERENT TEST NUMBER. YOU MAY SELECT THAT NUMBER IN ANY DIRECTION (HIGHER OR LOWER) BECAUSE EACH TEST IS SELF CONTAINED.

WRAP AROUND DATA PATTERNS MAY BE SELECTED VIA SW6 WHEN IN SINGLE TEST MODE. A TELETYPE REQUEST IS MADE FOR THE DESIRED DATA PATTERN WHENEVER SWITCH TEN (SW10) AND SWITCH SIX (SW6) ARE SET TO A ONE (1) WHILE ONE OF THE WRAP TESTS IS SELECTED IN SMD-SM5.

8. SUBTEST SUMMARIES

NOTE: FOR TESTS 1-15

FOR THE MOST PART, THIS DIAGNOSTIC TESTS PARTICULAR AREAS OF THE TMD3 LOGIC INDEPENDENT OF THE TE16. HOWEVER THERE ARE A FEW SIGNALS WHICH ARE REQUIRED FROM THE TE16 TO COMPLETE THE TESTS, AND AT LEAST ONE CASE WHERE TE16 FAILURES INTERFERE WITH THE TESTS. THE KNOWN CASES ARE LISTED HERE AND SHOULD BE CHECKED AS PART OF THE DEBUGGING.

1. MOL(SB)L: REQUIRED TO ENABLE CLOCK
2. CLOCK(SB)L: REQUIRED TO GENERATE ACCELERATION AND SHUTDOWN.
3. WRITE CLOCK(SB)L: USED IN WAND TO GENERATE DATA AND REC(SB)L
4. RSDO(SB)L: SHOULD NOT OCCUR DURING WRAP AROUND TESTS, BUT WILL INTERFERE WITH THEIR OPERATION IF CAUSED BY A FAILURE SUCH AS A GROUNDED OUTPUT FROM THE G056.

LOGIC TEST 1: WRP3, NRZ, NORMAL, ODD (BIT FIDDLER READ)

PROGRAMMED SEQUENCE:

TAPE CONTROL REGISTER IS LOADED WITH DENSITY 3, FORMAT 14, ODD PARITY WRP3 IS LOADED IN MAINT. REGISTER. READ FUNCTION IS LOADED. EXECUTING WRP3 CONSISTS LOADING DATA CHARACTERS INTO MAINT. REGISTER DATA FIELD, WHERE THERE ARE MULTI- PLEXERS TO BIT FIDDLER, MM CLK IS TOGGLED TO CREATE RDS. THE BIT FIDDLER TRANSMIT DATA ACCESS MASSBUS DATA LINES. WHEN ALL THE DATA HAS BEEN TRANSMITTED AN EOR CLK IS TRANSMITTED TO N REGISTER WHICH BRINGS OPERATION TO A CLOSE.

LIKELY FAULT LOCATIONS: M8906, M8905, MASSBUS P-LINES

CIRCUITS

PRINT REFERENCES

MASSBUS CHAR. ASSEMBLE
CLK. GENERATOR
MAINT. REGISTER DATA FIELD
RDS GENERATION

BF5
BF2
MR2, MR3
MRS

K01

LOGIC TEST 2: WARP3, PE, NORMAL, ODD

JUST LIKE TEST 1 EXCEPT FOR DENSITY BITS.

LOGIC TEST 3: WRAP2, NRZ, NORMAL, ODD

PROGRAMMED SEQUENCE:

WRAP2 IS BIT FIDDLER WRITE. MM CLOCK IS MULTIPLEXED INTO WRT CLK SO THAT IT FORMS WRT STROBE. THE OUTPUT OF THE BIT FIDDLER IS CLOCKED INTO THE DATA FIELD OF THE MAINTENANCE REGISTER. SET UP CONSISTS OF MOVING NRZ, NORMAL FORMAT, ODD PARITY TO UNIT DESCRIPTION MAINT. REGISTER IS LOADED WITH WAP2 WRITE COMMAND IS ISSUED. AFTER THE ACCELERATION DELAY, MM CLOCK ARE GENERATED UNTIL ALL THE DATA HAS BEEN CLOCKED. SEQUENCE IS COMPLETED BY LOADING MAINTENANCE REGISTER WITH EOR CLR. THE SEQUENCE IS REPEATED WITH VARYING DATA PATTERNS.

LIKELY FAULT LOCATIONS: M8906, M8905, M8903

CIRCUITS

BIT FIDDLER CHAR UNPACK
BIT FIDDLER DATA REQUEST
WRT STRB.
MAINT. REG. DATA FIELD

PRINT REFERENCES

BF4
BF2
TCCM4
MR2, MR3

LOGIC TEST 4: WRP2, PE, NORMAL, ODD

THE TEST IS EXACTLY LIKE TEST 43 EXCEPT THAT PE WRT CLK ENBL L MUST BE ASSERTED BY M8902 TO ENABLE WR TO STROBE. THIS DOES NOT HAPPEN UNTIL THE PE WRITE CONTROL CIRCUIT HAS CLOCKED THROUGH THE PREAMBLE.

CIRCUITS

PRINT REFERENCES

(IN ADDITION TO TEST 44)
PE WRITE CONTROL

TCPE3

LOGIC TEST 5: WRP1, NRZ, NORMAL, ODD

THIS TEST IS EXACTLY LIKE TEST 43 EXCEPT THE WRITE BUFFER (TCCM2) IS MULTIPLEXED TO THE MAINTENANCE REGISTER.

LIKELY FAULT LOCATIONS: M8903, M8904 (CRC GENERATOR)

CIRCUITS

PRINT REFERENCES

WRITE BUFFER
CRC GENERATOR

TCCM2
CNRZ2

LOGIC TEST 6: WRAP1, PE, NORMAL, ODD

IN PE MODE BOTH THE PREAMBLE AND POSAMBLE ARE CLOCKED THROUGH THE WRITE BUFFER IN ADDITION TO PHASE ENCODED DATA.

LIKELY FAULT LOCATIONS: M8902 (WRITE CONTROL STATES), M8903

CIRCUITS

PRINT REFERENCE

WRITE BUFFER
WRITE CONTROL

TCCM2
TCPE3

LOGIC TEST 7: WRAP0, NORMAL, ODD

WRAP 0 IS THE MOST COMPLETE OF THE TMO3 WRAPAROUND DATA PATH. IT CONSISTS OF A WRITE OPERATION IN WHICH THE OUTPUT OF THE WRITE DATA BUFFER IS MULTIPLEXED TO THE READ DATA INPUTS, CHECKED AND LOADED INTO THE MAINTENANCE REGISTER FOR RETRIEVAL BY THE PROCESSOR. THE WHOLE OPERATION USES THE TYPE SYSTEM CLOCKS AND HAPPENS AT THE PROPER DATA RATES. MM CLK SERVES AS A FLAG ANNOUNCING WHEN A NEW CHARACTER HAS BEEN LOADED INTO THE MAINTENANCE REGISTER. IN PE MODE EVERY OTHER CHARACTER IS READ TO ALLOW SUFFICIENT PROCESSOR LOOP TIME. IN NRZ WRAP 0 IS EXPECTED TO PRODUCE LRC ERRORS BECAUSE THE TMO3 DOES NOT WRITE THE LRC CHARACTER.

LIKELY FAULT LOCATIONS: M8904, M8903

CIRCUITSPRINT REFERENCES

CRC GENERATOR	CNRZ2
CRC CHECKOUT	CNRZ3
CRC, CRC STROBE	TCCM4
READ LINE MULTIPLEXERS	TCCM6
MM CLK	MRS
CRC READ TIMING	CNRZ4
SHUTDOWN CIRCUITRY	TCCM5

LOGIC TEST 10: WRPO, PE, NORMAL, ODD

REPEAT OF TEST 7 IN PE MODE.

LIKELY FAULT LOCATIONS: M8901, M8902, M8903

CIRCUITSPRINT REFERENCES

DATA DISCRIMINATOR	DS2, DS4, DS6
PHASE LOCKED CLOCK	DS3, DS5, DS7
SKREW BUFFER	DS3, DS5, DS7
PE WRITE MAJOR STATES	TCPE3
PE READ MAJOR STATES	TCPE5
WRAP 0 CIRCUIT TO BLOCK RLT RDS	TCPE3
DESKREW BUFFER READ COUNTER	TCPE4

NO1

LOGIC TEST 11: CORE DUMP WRITE, WAM2

REPEAT OF TEST 3 EXCEPT BIT FIDDLER OPERATES IN CORE DUMP
MODE!

LIKELY FAULT LOCATION: M8906

LOGIC TEST 12: CORE DUMP READ, WAM3

REPEAT OF TEST 1 EXCEPT BIT FIDDLER OPERATES IN CORE DUMP
MODE!

LIKELY FAULT LOCATION: M8906

LOGIC TEST 13: EVEN PARITY WRITE - WAM1

REPEAT OF TEST 5 EXCEPT EVEN PARITY IS SPECIFIED.

LIKELY FAULT LOCATION: M8903

LOGIC TEST 14: EVEN PARITY READ: WAM0,

REPEAT OF TEST 7 EXCEPT EVEN PARITY IS USED.

LIKELY FAULT LOCATIONS: M8903, M8904

LOGIC TEST 15: READ REVERSE, WAM3 (M8906)

REPEAT OF TEST 2 EXCEPT READ REVERSE COMMAND IS ISSUED.

LIKELY FAULT LOCATIONS: M8908, M8909

LOGIC TEST 16: CRC ERROR CORRECTION

THIS TEST SIMULATES A BAD TRACK ON TAPE RESULTING IN A CRC ERROR & SUBSEQUENT CORRECTION OF DATA IN THE FAILING TRACK. THE TEST PROCEEDS THROUGH THE FOLLOWING STEPS:

A:WRITE DATA USING WRAP 0
B:REWRITE DATA WITH DATA BITS IN ONE TRACK ALTERED USING WRAP 4
C:READ REVERSE USING WRAP3
D:REWRITE DATA AS IN STEPB USING WRAP4
AT THIS POINT THE DATA READ BACK HAS BEEN CORRECTED TO MATCH THE DATA WRITTEN IN STEP A
E:REPEAT STEPS A-D ABOVE FOR EACH TRACK
F:REPEAT STEPS A-E ABOVE FOR ALL 1'S,ALL 0'S & 125125 DATA PATTERNS.

523

%

.LIST BIN,LOC,SEQ

524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555

```

.TITLE TM03/TE16 CONTROL LOGIC TEST-PART II
:MAINDEC-11-DZTEB-A-D
:FEB 77
:R. BARNES
:MCALL .SACT11,.SEOP,$CATCH,$SAVE,$RESTORE,$CHAIN,$SCHNMODE
:NLIST MC
:LIST ME
:ENABLE ABS,AMA

:CONSOLE SWITCHES*****
:SW15: 1=HALT ON ERROR
        0=CONTINUE
:SW14: 1=LOOP ON ERROR
        0=CONTINUE
:SW13: 1=DO NOT PRINT ERRORS
        0=PRINT ERRORS
:SW12: 1=HALT AT END OF PASS
        0=CONTINUOUS CYCLE
:SW11: 1=INHIBIT ITERATIONS
        0=DO ITERATIONS
:SW10: 1=HALT AT END OF EACH TEST
        0=CONTINUE
:SW8:  1=NO WRAP DATA CHECK
        0=DO WRAP DATA CHECK
:SW7:  1=NO WRAP STATUS CHECK
        0=DO WRAP STATUS CHECK
:SW6:  1=SELECTABLE WRAP DATA PATTERN (IN SINGLE TEST)
        0=AUTO PATTERNS
:SW0-5: SELECT TEST NUMBER :: 00=ALL TESTS
;IF HARDWARE SWR <15::00> = 17777 OR NOT AVAILABLE USE SOFTWARE SWR

```



```

602                                     ;REGISTER EQUIVS*****
603
604                                     R0=%0
605                                     R1=%1
606                                     R2=%2
607                                     R3=%3
608                                     R4=%4
609                                     R5=%5
610                                     SP=%6
611                                     PC=%7
612
613
614
615                                     ;ACT11 HOOK *****
616                                     $SVPC=.                               ;SAVE CURRENT LOCATION CTR
617                                     .=46
618 000046 002310                       .WORD SENDAD                       ;SET LOCATION 46
619                                     .=52
620 000052 000000                       .WORD 0                               ;SET LOCATION 52 = 0
621                                     .=$SVPC                               ;RESTORE LOCATION CTR
622
623                                     ;TTY INTERRUPT VECTOR*****
624
625                                     .=60
626 000060 012250                       .WORD TTINT                          ;TTY INTERRUPT HANDLER ADDRESS
627 000062 000340                       .WORD 340                            ;PRIORITY LEVEL 7
628
629                                     ;SOFTWARE SWITCH REGISTER*****
630                                     ;USED IF HARDWARE SWR = 177777, OR NOT AVAIL.
631                                     .=176
632 000176 000000                       SWREG: .WORD 0
633
634                                     ;START ADDRESS*****
635                                     .=200
636 000200 000137 001200                 JMP START ;PROGRAM START
637
638                                     ;RESTART ADDRESS*****
639                                     .=210
640 000210 000137 001670                 JMP ST2
641
642                                     ;TMO3 INTERRUPT VECTOR*****
643
644                                     .=224
645 000224 012240                       MTINT                                ;TAPE INTERRUPT HANDLER ADDRESS
646 000226 000340
647
648

```

```

649
650          000510          .:=510
651                                     ;MASS BUS REGISTER EQUIVS*****
652
653 000510 172440          C1: 172440
654 000512 172442          WC: 172442
655 000514 172444          BA: 172444
656 000516 172446          FC: 172446
657 000520 172450          CS: 172450
658 000522 172452          DS: 172452
659 000524 172454          ER: 172454
660 000526 172456          AS: 172456
661 000530 172460          CC: 172460
662 000532 172462          DB: 172462
663 000534 172464          MR: 172464
664 000536 172466          DT: 172466
665 000540 172470          SN: 172470
666 000542 172472          TC: 172472
667
668                                     ;ILLEGAL FUNCTION CODES
669
670 000544 005405          ILFT: 5405
671 000546 007415          7415
672 000550 016423          16423
673 000552 020437          20437
674 000554 022443          22443
675 000556 025447          25447
676 000560 031455          31455
677 000562 033465          33465
678 000564 036473          36473
679
680                                     ;CONSTANTS*****
681
682 000566 177776          PSH: 177776          :PROCESSOR STATUS
683 000570 177570          SMR: 177570          :SWITCH REGISTER
684 000572 177560          TKS: 177560          :TTY READER STATUS
685 000574 177562          TKB: 177562          :TTY READ BUFFER
686 000576 177564          TPS: 177564          :TTY PUNCH STATUS
687 000600 177566          TPB: 177566          :TTY PUNCH BUFFER
688 000602 177777          SERNUM: 177777       :SERIAL NUMBER
689 000604 000011          DRVTP: 011           :DRIVE TYPE
690 000606 000020          ITAMT: 20            :ITERATION AMOUNT
691 000610 000224          VECT: 224            :INTERRUPT VECTOR(RH)
692 000612 172440          REGS: 172440         :STARTING REGISTER ADDRESS
  
```

693		
694		
695		
696		
697		
698	000614	
699	000614	000000
700	000616	000000
701	000620	000000
702	000622	000000
703	000624	000000
704	000626	000000
705	000630	000000
706	000632	000000
707	000634	000000
708	000636	000000
709	000640	000000
710	000642	000000
711	000644	000000
712	000646	000000
713	000650	000000
714	000652	000000
715	000654	000000
716	000656	000000
717	000660	000000
718	000662	000000
719	000664	000000
720	000666	000000
721	000670	000000
722	000672	000000
723	000674	000000
724	000676	000000
725	000700	000000
726	000702	000000
727	000704	000000
728	000706	000000
729	000710	000000
730	000712	000000
731	000714	000000
732	000716	000000
733	000720	000000
734	000722	000000
735	000724	000000
736	000726	000000
737	000730	000000
738	000732	000000
739	000734	000000
740	000736	000000
741	000740	000000
742	000742	000000
743	000744	000000
744	000746	000000
745	000750	000000
746	000752	000000
747	000754	000000
748	000756	000000

```

; FLAGS AND COUNTERS*****
; NOTE ALL FLAGS AND COUNTERS ARE CLEARED ON STARTUP. PUT ANY
; ADDITIONAL FLAGS BETWEEN STFLGS (START OF FLAGS) AND ENDFLG
; (END OF FLAGS)

```

```

STFLGS:
TOB: 0
TIB: 0
HDFL: 0
EMADR: 0
DRVN: 0
TROO: 0
TRO1: 0
TRO2: 0
TRO3: 0
TRO4: 0
TRO5: 0
TRO6: 0
TRO7: 0
TR10: 0
TR11: 0
TR12: 0
TR13: 0
TR14: 0
TR15: 0
NRZOF: 0
SLVN: 0
PFLG: 0
RTRN: 0
ERADD: 0
TEMP1: 0
TEMP2: 0
TEMP3: 0
ITCNT: 0
SAV1: 0
SAV2: 0
SAV3: 0
SCOLP: 0
ITRLP: 0
EXFL: 0
ATAF: 0
SLAF: 0
SSCF: 0
ERRF: 0
ASF: 0
SCF: 0
TREF: 0
PEXFL: 0
STFLG: 0
LTADD: 0
T24FL: 0
ADDFL: 0
WAM: 0
FUN: 0
DATC: 0
WTAD: 0

```

749 000760 000000
 750 000762 000000
 751 000764 000000
 752 000766 000000
 753 000770 000000
 754 000772 000000
 755 000774 000000
 756 000776 000000
 757 001000 000000
 758 001002 000000
 759 001004 000000
 760 001006 000000
 761 001010 000000
 762 001012 000000
 763 001014 000000
 764 001016 000000
 765 001020 000000
 766 001022 000000
 767 001024
 768
 769
 770
 771 001024 000000
 772 001026 000000
 773 001030 000000
 774 001032 000000
 775
 776
 777
 778 001034
 779 001034 005076
 780 001036 005116
 781 001040 005122
 782 001042 005130
 783
 784
 785
 786 001044 000005
 787 001046 000005
 788 001050 000012
 789 001052 000012
 790 001054 000000
 791 001056 000017
 792 001060 000017
 793 001062 000017
 794 001064 000017
 795 001066 000000

DATA0: 0
 ROAD: 00
 W2FLG: 0000
 DERFL: 0000
 PREFL: 0000
 SERFL: 0000
 CRCNT: 0000
 UDES: 0000
 WPGFL: 0000
 PATRN: 0000
 STATF: 0000
 RDRVF: 0000
 RCDP: 0000
 STATC: 0000
 SKAT: 0000
 PCNTR: 0 ;PASS COUNTER
 DCHKFL: .WORD 0 ;DATA CHECK FLAG 0/1 = CHECK/DO NOT CHECK
 CRCFLG: .WORD 0 ;CRC CORRECTION TEST IN PROGRESS
 ENDFLG:
 ;EXPT WRAP STATUS*****
 WCS1: 0
 WCS2: 0
 WDS: 0
 WER: 0
 ;DATA PATTERN GENERATORS*****
 DATBL:
 DATA0: DAT1 ;ALL ONE BITS
 DATA1: DAT2 ;ALL ZERO BITS
 DATA2: DAT3 ;ALTERNATING ONE/ZERO BITS
 DATA3: DAT4 ;ALTERNATING PARITY CHARACTERS
 ;CORE DUMP PATTERNS*****
 WCDP2: 5
 5
 12
 12
 0
 WCDPO: 17
 17
 17
 17
 0

```

796
797
798
799 001070 000000
800 001072 000000
801 001074 002352
802 001076 002352
803 001100 002464
804 001102 002464
805 001104 002536
806 001106 002536
807 001110 002644
808 001112 002644
809 001114 002716
810 001116 002716
811 001120 003024
812 001122 003024
813 001124 003102
814 001126 003102
815 001130 003210
816 001132 003210
817 001134 003262
818 001136 003262
819 001140 003374
820 001142 003374
821 001144 003522
822 001146 003522
823 001150 003572
824 001152 003572
825 001154 003642
826 001156 003642
827 001160 003724
828 001162 003724
829 001164 004312
830 001166 004312
831 001170 004624
832 001172 004624
833 001174 002244
834 001176 000020

```

;LOGIC TEST ENTRY TABLE*****

```

TSTTBL: 0
          0
          LT1
          LT1
          LT2
          LT2
          LT3
          LT3
          LT4
          LT4
          LT5
          LT5
          LT6
          LT6
          LT7
          LT7
          LT10
          LT10
          LT11
          LT11
          LT12
          LT12
          LT13
          LT13
          LT14
          LT14
          LT15
          LT15
          LT16
          LT16
          LT17
          LT17
          LT20
          LT20

```

```

TADX: .WORD TEND
TLAST: .WORD 20

```

;CONTAINS # OF TESTS

```

835          .EVEN
836          ;PROGRAM START AND HOUSEKEEPING*****
837
838 001200 012706 000500          START:  MOV    #500,SP          ;SET STACK POINTER
839 001204 013746 000004          MOV    @#4,-(SP)        ;SAVE ERROR TRAP
840 001210 013746 000006          MOV    @#6,-(SP)
841 001214 012737 001240 000004  MOV    @#15,@#4        ;SET TIME OUT TRAP TO GO TO 15
842 001222 005037 000006          CLR    @#6
843 001226 022777 177777 177334  CMP    @#177777,@SWR    ;USE SOFTWARE SWITCH IF SWR = 177777
844 001234 001402                BEQ    25                ;OR TIMES OUT
845 001236 000404                BR     35                ;OTHERWISE USE HARDWARE SWR
846 001240 022626                15:  CMP    (SP)+,(SP)+    ;RESET STACK
847 001242 012737 000176 000570  25:  MOV    @SWREG,SWR      ;SET SWR = TO ADDRESS OF SOFTWARE SWR
848 001250 012637 000006          35:  MOV    (SP)+,@#6        ;RESTORE ERROR TRAP
849 001254 012637 000004          MOV    (SP)+,@#4
850 001260 005027                CLR    (PC)+            ;: CLEAR CHAIN INDICATOR
851 001262 000000          CHNFLG: .WORD 0        ;: CHAIN MODE INDICATOR
852                                     ;: 1/0 = CHAIN/NOT CHAIN MODE
853 001264 022737 002310 000042  CMP    @SENDAD,@#42    ;: BRANCH IF LOADED VIA ACT11 CHAIN MODE
854 001272 001404                BEQ    50$              ;: BRANCH IF IN DUMP MODE
855 001274 005737 000042          TST    @#42
856 001300 001413                BEQ    52$              ;: BRANCH IF IN DUMP MODE
857 001302 000406                BR     51$              ;: BRANCH IF IN DUMP MODE
858 001304 012737 000176 000570  50$: MOV    @SWREG,SWR      ;: INVOKE SOFTWARE SWR
859 001312 012777 100000 177250  MOV    @#100000,@SWR  ;: WITH HALT ON ERROR SET
860 001320 005237 001262          51$: INC    CHNFLG        ;: SET CHNFLG = CHAIN MODE
861 001324 000137 001714          JMP    TSCD            ;: GO TO CHAIN ADDRESS
862 001330                52$:
863 001330 122737 000006 000041  45:  CMPB   @#6,@#41        ;: BRANCH IF NOT LOADED VIA TMDP
864 001336 001004                BNE    55
865 001340 012704 014433          MOV    @MSG62,R4       ;: ADVISE USER TO REMOVE MEDIA FROM UUT
866 001344 004737 012714          JSR   PC,TTOUT
867 001350 012704 013636          55:  MOV    @MSG1,R4
868 001354 004737 012714          JSR   PC,TTOUT        ;: PRINT TITLE
869 001360 112737 000043 013636  MOVB   #'@,MSG1        ;: DO NOT PRINT TITLE ON RESTART
870 001366 012704 014327          MOV    @MSG44,R4
871 001372 004737 012714          JSR   PC,TTOUT        ;: REQUEST REGISTER ADDRESS
872 001376 013703 000612          MOV    REGS,R3
873 001402 004737 013042          JSR   PC,OCTP         ;: PRINT CURRENT ADDRESS
874 001406 012705 000612          MOV    @REGS,R5
875 001412 012701 000007          MOV    #7,R1          ;: SET ADDRESS SAVE LOC
876 001416 012702 176400          MOV    @#176400,R2    ;: SET SIZE OF RESPONSE
877 001422 012703 172300          MOV    @#172300,R3    ;: SET UPPER LIMIT
878 001426 004737 012372          JSR   PC,TTR          ;: SET LOWER LIMIT
879 001432 012704 014351          MOV    @MSG45,R4      ;: GO GET RESPONSE
880 001436 004737 012714          JSR   PC,TTOUT        ;: REQUEST VECTOR
881 001442 013703 000610          MOV    VECT,R3
882 001446 004737 013042          JSR   PC,OCTP         ;: PRINT CURRENT VECTOR
883 001452 012705 000610          MOV    @VECT,R5
884 001456 012701 000004          MOV    #4,R1          ;: SET ADDRESS SAVE LOC
885 001462 012702 000224          MOV    @#224,R2       ;: SET SIZE OF RESPONSE
886 001466 012703 000150          MOV    @#150,R3       ;: SET UPPER LIMIT
887 001472 004737 012372          JSR   PC,TTR          ;: SET LOWER LIMIT
888 001476 013700 000610          MOV    VECT,R0        ;: GO GET RESPONSE
889 001502 012720 012240          MOV    @MTINT,(R0)+   ;: GET VECTOR
890 001506 012710 000340          MOV    @#340,(R0)    ;: LOAD INTERRUPT ADDRESS IN VECTOR
                        ;: LOAD PRIORITY

```

```

891 001512 013700 000612      MOV      REGS,R0      ;GET START OF REGS
892 001516 012701 000016      MOV      #16,R1      ;SET NUMBER OF REGS
893 001522 012702 000510      MOV      #C1,R2      ;GET START OF TABLE
894 001526 010022 000002      6S:     MOV      R0,(R2)+ ;BUILD TABLE
895 001530 062700 000002      ADD      #2,R0      ;BUMP ADDRESS
896 001534 005301 000002      DEC      R1        ;SEE IF DONE
897 001536 001373 000002      BNE      6S        ;IF NOT: BR
898 001540 012702 000614      MOV      #STFLG,R2
899 001544 012700 000210      MOV      #ENDFLG-STFLG,R0 ;GET # OF FLAGS TO CLEAR
900 001550 006200 000000      ASR      R0        ;FORM COUNT
901 001552 005022 000000      7S:     CLR      (R2)+     ;CLEAR FLAGS + COUNTERS
902 001554 005300 000000      DEC      R0
903 001556 001375 000000      BNE      7S
904 001560 012704 014373      MOV      #MSG57,R4   ;REQUEST TMO3 DRIVE #
905 001564 004737 012714      JSR      PC,TTOUT
906 001570 013703 000624      MOV      DRVN,R3    ;GET CURRENT DRIVE
907 001574 004737 013042      JSR      PC,OCTP    ;AND TYPE IT
908 001600 012705 000624      MOV      #DRVN,R5   ;TTR ROUTINE RETURNS DRIVE TO (R5)
909 001604 012701 000002      MOV      #2,R1     ;LIMIT RESPONSE TO 1 CHARACTER
910 001610 012702 000007      MOV      #7,R2     ;BETWEEN 0 AND 7
911 001614 012703 000000      MOV      #0,R3
912 001620 004737 012372      JSR      PC,TTR     ;GET RESPONSE & PUT IN DRVN
913 001624 012704 014411      MOV      #MSG58,R4   ;REQUEST SLAVE #
914 001630 004737 012714      JSR      PC,TTOUT
915 001634 013703 000664      MOV      SLVN,R3    ;GET CURRENT SLAVE #
916 001640 004737 013042      JSR      PC,OCTP    ;AND TYPE IT
917 001644 012705 000664      MOV      #SLVN,R5   ;TTR ROUTINE RETURNS REPOSENSE TO (R5)
918 001650 012701 000002      MOV      #2,R1     ;LIMIT RESPONSE TO 1 CHARACTER
919 001654 012702 000007      MOV      #7,R2     ;BETWEEN 0-7
920 001660 012703 000000      MOV      #0,R3
921 001664 004737 012372      JSR      PC,TTR     ;GET RESPONSE & PUT IT IN SLVN
922
923
924 001670 012706 000500      :START 210
925 001674 005037 001006      ST2:    MOV      #500,SP ;SET STACK PTR
926 001700 005037 001016      CLR      RDRVF     ;CLEAR READ REVERSE FLAG
927 001704 005037 001022      CLR      PCNTR
928 001710 004737 013474      CLR      CRCFLG    ;SET CRC FLAG = CRC NOT IN PROGRESS
                      JSR      PC,GTSWR ;GET SOFTWARE SWITCHES

```



```

929
930 ;TEST SCHEDULAR**:*
931
932 001714 052777 000100 176650 TSCD: BIS #100,ATKS ;SET KEYBOARD IE BIT
933 001722 005037 001000 CLR WPGFL ;CLEAR WRAP PATRN FLAG
934 001726 005037 000740 CLR STFLG ;CLEAR SINGLE TEST FLAG
935 001732 017700 176632 MOV @SWR,RO
936 001736 042700 177700 BIC #177700,RO ;BRANCH IF SINGLE TEST SELECTED
937 001742 001122 STSCD ;GO SELECT SINGLE TEST
938 001744 005737 001262 TST CHNFLG ;BRANCH IF NOT IN CHAIN MODE
939 001750 001457 BEQ TSCDA
940 001752 012737 177777 000624 MOV #1,DRVN ;;INITIALIZE DRIVE #
941 001760 012737 177777 000664 NXTDRV: MOV #1,SLVN ;;INITIALIZE SLAVE #
942 001766 012777 000040 176524 IS: MOV #40,@CS ;INIT CONTROLLER
943 001774 005237 000624 INC DRVN ;STEP DRIVE #
944 002000 022737 000010 000624 CMP #10,DRVN ;EXIT IF ALL DRIVES TESTED
945 002006 001521 BEQ $DONE ;FOR AVAILABILITY
946 002010 013777 000624 176502 MOV DRVN,@CS ;LOAD DRIVE #
947 002016 005777 176466 TST @C1 ;ACCESS DRIVE
948 002022 032777 010000 176470 BIT #10000,@CS ;BRANCH IF DRIVE NON EXISTANT
949 002030 001356 BNE IS ;(NED = 1)
950 002032 005237 000664 NXTSLV: INC SLVN ;STEP SLAVE # AND BRANCH
951 002036 001011 BNE IS ;IF NOT SLAVE 0
952 002040 005737 000624 TST DRVN ;BRANCH IF NOT DRIVE # 0
953 002044 001006 BNE IS
954 002046 122737 000006 000041 CMPB #6,@#41 ;;BRANCH IF NOT TMDP
955 002054 001002 BNE IS
956 002056 005237 000664 INC SLVN ;;STEP TO SLAVE # 1
957 002062 022737 000010 000664 IS: CMP #10,SLVN ;BRANCH IF ALL SLAVES TESTED
958 002070 001733 BEQ NXTDRV ;FOR AVAILABILITY
959 002072 013777 000664 176442 MOV SLVN,@C ;LOAD SLAVE UNIT #
960 002100 032777 002000 176430 BIT #2000,@C ;BRANCH IF SLAVE NOT
961 002106 001751 BEQ NXTSLV ;PRESENT (SPR = 0)
962 002110 012737 001070 000742 TSCDA: MOV #TSTTBL,LTADD
963 002116 062737 000004 000742 TSCD0: ADD #4,LTADD
964 002124 013737 000742 000714 TSCD1: MOV LTADD,ITRLP
965 002132 062737 000002 000714 ADD #2,ITRLP ;SET ITERATION ADDRESS
966 002140 005037 000620 CLR WPGFL ;CLEAR PRINT HEADER FLAG
967 002144 017700 176572 MOV @LTADD,RO ;SET POINTER TO TEST
968 002150 000110 JMP (RO) ;GO TO TEST
969 002152 032777 002000 176410 TSCD2: BIT #2000,@SWR ;SEE IF HALT ON TEST
970 002160 001403 BEQ TSCD3 ;IF NOT: BR
971 002162 000000 HALT
972 002164 005037 001000 CLR WPGFL ;CLEAR WRAP DATA GENERATOR FLAG
973 002170 005737 000740 TSCD3: TST STFLG ;SE IF SINGLE TEST
974 002174 001750 BEQ TSCD0 ;IF NOT: BR
975 002176 017700 176366 MOV @SWR,RO
976 002202 042700 177700 BIC #177700,RO ;MASK TEST NUMBER
977 002206 001642 BEQ TSCD ;IF SO: BR
978 002210 012737 000001 000740 STSCD: MOV #1,STFLG ;SET SINGLE TEST FLAG
979 002216 023700 001176 CMP TLAST,RO ;SEE IF EXCEEDED TESTS
980 002222 002410 BLT TEND ;IF SO: BR
981 002224 006300 RO
982 002226 006100 RO ;SET TABLE MODIFIER
983 002230 012737 001070 000742 MOV #TSTTBL,LTADD
984 002236 060037 000742 ADD RO,LTADD ;SET TEST POINTER

```

985	002242	000730			BR	TSCD1	
986	002244	005737	001262	TEND:	TST	CHNFLG	;BRANCH IF IN CHAIN MODE
987	002250	001270			BNE	NXTSLV	
988	002252	012704	014310	SDONE:	MOV	#MSG41,R4	
989	002256	004737	012714		JSR	PC,TTOUT	;PRINT END OF PASS
990	002262	013703	001016		MOV	PCNTR,R3	
991	002266	004737	013042		JSR	PC,OCTP	;PRINT PASS NUMBER
992	002272	005000			CLR	RO	
993	002274	005300		IS:	DEC	RO	;DELAY WAITING FOR
994	002276	001376			BNE	IS	;TTY TO FINISH
995	002300	013700	000042		MOV	#42,RO	;GET ACT11 RETURN ADDRESS
996	002304	001405			BEQ	HERE	;BRANCH IF NOT ACT11
997	002306	000005			RESET		
998	002310	004710		SENDAD:	JSR	PC,(RO)	
999	002312	000240			NOP		
1000	002314	000240			NOP		
1001	002316	000240			NOP		
1002	002320	000240		HERE:	NOP		
1003	002322	005737	001262		TST	CHNFLG	;BRANCH IF IN CHAIN MODE
1004	002326	001005			BNE	TENDX	
1005	002330	032777	010000 176232		BIT	#10000,#SWR	;SEE IF HALT ON PASS
1006	002336	001001			BNE	TENDX	;IF NOT: BR
1007	002340	000000			HALT		
1008	002342	005237	001016	TENDX:	INC	PCNTR	;BUMP PASS COUNTER
1009	002346	000137	001714		JMP	TSCD	;RESTART
1010							

```

1011 ;THESE TESTS CHECK DATA FORMATTING
1012 ;AND TRANSFER THROUGH THE TMO3 WRAP AROUND MODES
1013
1014 ;LOGIC TEST 1: WRAP 3, NRZ, NORMAL ODD *****
1015
1016 002352 012737 004270 001024 LT1: MOV #4270,WCS1 ;SET EXPT CS1
1017 002360 012737 000100 001026 MOV #100,WCS2 ;SET EXPT CS2
1018 002366 012737 010600 001030 MOV #10600,WDS ;SET EXPT DS
1019 002374 012737 000000 001032 MOV #0,WER ;SET EXPT WER
1020 002402 012737 014527 000622 MOV #MSLT1,EMADDR ;SET HEADER
1021 002410 012737 001700 000776 MOV #1700,ODES ;SET NRZ,NORMAL, ODD
1022 002416 005037 001002 LT1A: CLR PATRN ;POINT TO PATTERN 0
1023 002422 012737 002430 000712 MOV #LT1B,SCOLP ;SET SCOPE ADDRESS
1024 002430 004737 005256 LT1B: JSR PC,WAM3 ;GO DO WRAP 3
1025 002434 005237 001002 INC PATRN ;BUMP PATTERN POINTER
1026 002440 032737 000004 001002 BIT #4,PATRN ;SEE IF DONE
1027 002446 001770 BEQ LT1B ;IF NOT: BR
1028 002450 004737 012014 JSR PC,ITER ;GO SEE IF ITERATIONS
1029 002454 005037 001006 CLR RDRVF ;CLEAR READ REVERSE FLAG
1030 002460 000137 002152 JMP TSCD2 ;RETURN TO SCHEDULAR
1031
1032 ;LOGIC TEST 2: WRAP 3, PE, NORMAL, ODD*****
1033
1034 002464 000240 LT2: NOP
1035 002466 012737 004270 001024 LT2A: MOV #4270,WCS1 ;SET EXPT CS1
1036 002474 012737 000100 001026 MOV #100,WCS2 ;SET EXPT CS2
1037 002502 012737 010640 001030 MOV #10640,WDS ;SET EXPT DS
1038 002510 012737 000000 001032 MOV #0,WER ;SET EXPT WER
1039 002516 012737 014575 000622 MOV #MSLT2,EMADDR ;SET HEADER
1040 002524 012737 002300 000776 MOV #2300,ODES ;SET PE, NORMAL, ODD
1041 002532 000137 002416 JMP LT1A ;EXECUTE TEST SEQUENCE
1042
1043 ;LOGIC TEST 3: WRAP 2, NRZ, NORMAL, ODD*****
1044
1045 002536 012737 004260 001024 LT3: MOV #4260,WCS1 ;SET EXPT CS1
1046 002544 012737 000100 001026 MOV #100,WCS2 ;SET EXPT CS2
1047 002552 012737 010600 001030 MOV #10600,WDS ;SET EXPT DS
1048 002560 012737 000000 001032 MOV #0,WER ;SET EXPT WER
1049 002566 012737 014642 000622 MOV #MSLT3,EMADDR ;SET HEADER
1050 002574 012737 001700 000776 MOV #1700,ODES ;SET TO NRZ,NORMAL, ODD
1051 002602 005037 001002 LT3A: CLR PATRN ;POINT TO PATTERN 0
1052 002606 012737 002614 000712 MOV #LT3B,SCOLP ;SET SCOPE ADDRESS
1053 002614 004737 005212 LT3B: JSR PC,WAM2 ;GO DO WRAP 2
1054 002620 005237 001002 INC PATRN ;BUMP POINTER
1055 002624 032737 000004 001002 BIT #4,PATRN ;SEE IF DONE
1056 002632 001770 BEQ LT3B ;IF NOT: BR
1057 002634 004737 012014 JSR PC,ITER ;GO SEE IF ITERATIONS
1058 002640 000137 002152 JMP TSCD2 ;RETURN TO SCHEDULAR

```

```

1059
1060 ;LOGIC TEST 4: WRAP 2, PE, NORMAL, ODD*****
1061
1062 002644 000240 LT4: NOP
1063 002646 012737 004260 001024 LT4A: MOV #4260,WCS1 ;SET EXPT CS1
1064 002654 012737 000100 001026 MOV #100,WCS2 ;SET EXPT CS2
1065 002662 012737 010640 001030 MOV #10640,WDS ;SET EXPT DS
1066 002670 012737 000000 001032 MOV #0,WER ;SET EXPT WER
1067 002676 012737 014710 000622 MOV #MSLT4,EMADDR ;SET HEADER
1068 002704 012737 002300 000776 MOV #2300,UDES ;SET PE, NORMAL, ODD
1069 002712 000137 002602 JMP LT3A ;GO EXECUTE TEST SEQUENCES
1070
1071 ;LOGIC TEST 5: WRAP 1, NRZ, NORMAL, ODD*****
1072
1073 002716 012737 004260 001024 LT5: MOV #4260,WCS1 ;SET EXPT CS1
1074 002724 012737 000100 001026 MOV #100,WCS2 ;SET EXPT CS2
1075 002732 012737 010600 001030 MOV #10600,WDS ;SET EXPT DS
1076 002740 012737 000000 001032 MOV #0,WER ;SET EXPT WER
1077 002746 012737 014755 000622 MOV #MSLT5,EMADDR ;SET HEADER
1078 002754 012737 001700 000776 MOV #1700,UDES ;SET NRZ, NORMAL, ODD
1079 002762 005037 001002 LT5A: CLR PATRN ;POINT TO PATTERN ZERO
1080 002766 012737 002774 000712 MOV #LT5B,SCOLP ;SET SCOPE ADDRESS
1081 002774 004737 005202 LT5B: JSR PC,WAM1 ;GO DO WRAP 1
1082 003000 005237 001002 INC PATRN ;BUMP POINTER
1083 003004 032737 000004 001002 BIT #4,PATRN ;SEE IF DONE
1084 003012 001770 BEQ LT5B ;IF NOT: BR
1085 003014 004737 012014 JSR PC,ITER ;GO SEE IF ITERATIONS
1086 003020 000137 002152 JMP TSCD2 ;RETURN TO SCHEDULAR
1087
1088 ;LOGIC TEST 6: WRAP 1, PE, NORMAL, ODD*****
1089
1090 003024 000240 LT6: NOP
1091 003026 004737 011512 LT6A: JSR PC,PPGEN ;GO GENERATE PRE/POSTAMBLE
1092 003032 012737 004260 001024 MOV #4260,WCS1 ;SET EXPT CS1
1093 003040 012737 000100 001026 MOV #100,WCS2 ;SET EXPT CS2
1094 003046 012737 010640 001030 MOV #10640,WDS ;SET EXPT DS
1095 003054 012737 000000 001032 MOV #0,WER ;SET EXPT WER
1096 003062 012737 015023 000622 MOV #MSLT6,EMADDR ;SET HEADER
1097 003070 012737 002300 000776 MOV #2300,UDES ;SET PE, NORMAL, ODD
1098 003076 000137 002762 JMP LT5A ;GO EXECUTE TEST SEQUENCE

```

```

1099
1100 ;LOGIC TEST 7: WRAP 0, NRZ,NORMAL, ODD*****
1101
1102 003102 012737 144260 001024 LT7: MOV #144260,WCS1 ;SET EXPT CS1
1103 003110 012737 000100 001026 MOV #100,WCS2 ;SET EXPT CS2
1104 003116 012737 150600 001030 MOV #150600,WDS ;SET EXPT DS
1105 003124 012737 000200 001032 MOV #200,WER ;SET EXPT ER
1106 003132 012737 015070 000622 MOV #MSLT7,EMADDR ;SET HEADER
1107 003140 012737 001700 000776 MOV #1700,UDES ;SET NRZ, NORMAL, ODD
1108 003146 005037 001002 LT7A: CLR PATRN ;POINT TO PATTERN 0
1109 003152 012737 003160 000712 MOV #LT7B,SCOLP ;SET SCOPE ADDRESS
1110 003160 004737 005136 LT7B: JSR PC,WANO ;GO DO WRAP 0
1111 003164 005237 001002 INC PATRN ;BUMP POINTER
1112 003170 032737 000004 001002 BIT #4,PATRN ;SEE IF DONE
1113 003176 001770 BEQ LT7B ;IF NOT: BR
1114 003200 004737 012014 JSR PC,ITER ;GO SEE IF ITERATIONS
1115 003204 000137 002152 JMP TSC02 ;RETURN TO SCHEDULAR
1116
1117 ;LOGIC TEST 10: WRAP 0, PE, NORMAL, ODD*****
1118
1119 003210 000240 LT10: NOP
1120 003212 012737 004260 001024 LT10A: MOV #4260,WCS1 ;SET EXPT CS1
1121 003220 012737 000100 001026 MOV #100,WCS2 ;SET EXPT CS2
1122 003226 012737 010640 001030 MOV #10640,WDS ;SET EXPT DS
1123 003234 012737 000000 001032 MOV #0,WER ;SET EXPT ER
1124 003242 012737 015136 000622 MOV #MSLT10,EMADDR ;SET HEADER
1125 003250 012737 002300 000776 MOV #2300,UDES ;SET PE, NORMAL, ODD
1126 003256 000137 003146 JMP LT7A ;GO EXECUTE TEST SEQUENCE

```

```

1127                                     ;LOGIC TEST 11: CORE DUMP WRITE, WAM2*****
1128
1129 003262 012737 004260 001024 LT11: MOV      #4260,WCS1      ;SET EXPT CS1
1130 003270 012737 000100 001026      MOV      #100,WCS2      ;SET EXPT CS2
1131 003276 012737 010600 001030      MOV      #10600,WDS     ;SET EXPT DS
1132 003304 012737 000000 001032      MOV      #0,WER        ;SET EXPT ER
1133 003312 012737 015204 000622      MOV      #MSLT11,EMADDR ;SET HEADER
1134 003320 012737 001720 000776      MOV      #1720,UDES     ;SET NRZ, CORE DUMP, ODD
1135 003326 005037 001002                CLR      PATRN          ;POINT TO PATTERN 0
1136 003332 012737 003340 000712      MOV      #LT11A,SCOLP   ;SET SCOPE ADDRESS
1137 003340 004737 005212                JSR      PC,WAM2        ;GO DO WAM 2
1138 003344 022737 000002 001002      CMP      #2,PATRN       ;SEE IF DONE
1139 003352 001404                BEQ      LT11X          ;IF SO: BR
1140 003354 012737 000002 001002      MOV      #2,PATRN       ;SELECT PATTERN 2
1141 003362 000766                BR       LT11A          ;CONTINUE
1142 003364 004737 012014                JSR      PC,ITER        ;GO SEE IF ITERATIONS
1143 003370 000137 002152                JMP      TSCD2          ;RETURN TO SCHEDULES
1144
1145                                     ;LOGIC TEST 12: CORE DUMP READ, WAM 3*****
1146
1147 003374 012737 004270 001024 LT12: MOV      #4270,WCS1      ;SET EXPT CS1
1148 003402 012737 000100 001026      MOV      #100,WCS2      ;SET EXPT CS2
1149 003410 012737 010600 001030      MOV      #10600,WDS     ;SET EXPT DS
1150 003416 012737 000000 001032      MOV      #0,WER        ;SET EXPT ER
1151 003424 012737 015255 000622      MOV      #MSLT12,EMADDR ;SET HEADER
1152 003432 012737 001720 000776      MOV      #1720,UDES     ;SELECT NRZ, CORE DUMP, ODD
1153 003440 005037 001002                CLR      PATRN          ;SELECT PATTERN 0
1154 003444 012737 003460 000712      MOV      #LT12A,SCOLP   ;SET SCOPE ADDRESS
1155 003452 012737 001056 001010      MOV      #WCDP0,RCDP    ;POINT TO PATTERN 0
1156 003460 004737 005256                JSR      PC,WAM3        ;GO DO WAM3
1157 003464 022737 000002 001002      CMP      #2,PATRN       ;SEE IF DONE
1158 003472 001407                BEQ      LT12X          ;IF SO: BR
1159 003474 012737 000002 001002      MOV      #2,PATRN       ;SELECT PATTERN 2
1160 003502 012737 001044 001010      MOV      #WCDP2,RCDP    ;POINT TO PATTERN 2
1161 003510 000763                BR       LT12A          ;CONTINUE
1162 003512 004737 012014                JSR      PC,ITER        ;GO SEE IF ITERATION
1163 003516 000137 002152                JMP      TSCD2          ;RETURN TO SCHEDULE

```

```

1164
1165
1166 ;LOGIC TEST 13: EVEN PARITY WRITE: WAM 1(M8903)*****
1167 003522 012737 004260 001024 LT13: MOV #4260,WCS1 ;SET EXPT CS1
1168 003530 012737 000100 001026 MOV #100,WCS2 ;SET EXPT CS2
1169 003536 012737 010600 001030 MOV #10600,WDS ;SET EXPT DS
1170 003544 012737 000000 001032 MOV #0,WER ;SET EXPT ER
1171 003552 012737 015325 000622 MOV #MSLT13,EMADDR ;SET HEADER
1172 003560 012737 001710 000776 MOV #1710,UDES ;SET NRZ, NORMAL, EVEN
1173 003566 000137 002762 JMP LT5A ;GO EXECUTE WAM 1
1174
1175 ;LOGIC TEST 14: EVEN PARITY READ: WAM 0(M8903 M8904)*****
1176
1177 003572 012737 144260 001024 LT14: MOV #144260,WCS1 ;SET EXPT CS1
1178 003600 012737 000100 001026 MOV #100,WCS2 ;SET EXPT CS2
1179 003606 012737 150600 001030 MOV #150600,WDS ;SET EXPT DS
1180 003614 012737 000200 001032 MOV #200,WER ;SET EXPT ER
1181 003622 012737 015406 000622 MOV #MSLT14,EMADDR ;SET HEADER
1182 003630 012737 001710 000776 MOV #1710,UDES ;SET NRZ, NORMAL, EVEN
1183 003636 000137 003146 JMP LT7A ;GO DO WAM 0
1184
1185 ;LOGIC TEST 15: READ REVERSE: WAM 3(M8906)*****
1186
1187 003642 012737 004276 001024 LT15: MOV #4276,WCS1 ;SET EXPT CS1
1188 003650 012737 000100 001026 MOV #100,WCS2 ;SET EXPT CS2
1189 003656 012737 010640 001030 MOV #10640,WDS ;SET EXPT DS
1190 003664 012737 000000 001032 MOV #0,WER ;SET EXPT ER
1191 003672 012737 015465 000622 MOV #MSLT15,EMADDR ;SET HEADER
1192 003700 012737 002300 000776 MOV #2300,UDES ;SELECT PE,NORMAL,ODD
1193 003706 000240 NOP
1194 003710 000240 NOP
1195 003712 012737 000001 001006 MOV #1,RDRVF ;SET READ REVERSE FLAG
1196 003720 000137 002416 JMP LT1A ;GO DO WAM 3, REVERSE
1197

```

```

1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218 003724 005037 001002
1219 003730 012737 000401 004302
1220 003736 012737 015532 000622
1221 003744 012737 003752 000712
1222
1223 003752 012737 144260 001024
1224 003760 012737 000100 001026
1225 003766 012737 150600 001030
1226 003774 012737 000200 001032
1227 004002 012737 001700 000776
1228
1229 004010 005037 001022
1230 004014 004737 005136
1231 004020 012737 000001 001022
1232 004026 013737 017276 004304
1233
1234 004034 013737 004304 004306
1235 004042 013737 004302 004310
1236 004050 043737 004302 004306
1237 004056 043737 004304 004310
1238 004064 053737 004310 004306
1239 004072 013737 004306 004304
1240
1241 004100 012737 144260 001024
1242 004106 012737 000110 001026
1243 004114 012737 150600 001030
1244 004122 012737 100300 001032
1245 004130 004737 005364
1246 004134 000240
1247 004136 012737 000001 001006
1248 004144 012737 144276 001024
1249 004152 012737 000110 001026
1250 004160 012737 150600 001030
1251 004166 012737 001000 001032
1252 004174 004737 005256
1253 004200 000240

```

; LOGIC TEST 16: CRC CORRECTION-SINGLE TRACK, EVERY FRAME
; THIS IS A TEST OF THE CRC CORRECTION LOGIC. THE TEST WRITES
; A KNOWN PATTERN (ALL 1'S, ALL 0'S & 125252) WITH A DATA BIT
; ALTERED IN EACH OF THE DATA TRACKS. THIS TEST INSURES THAT A
; CRC CORRECTABLE ERROR IS CORRECTED.
; THE TEST PROCEEDS AS FOLLOWS:
; STEP A WRITE A KNOWN PATTERN USING WRAP AROUND MODE 0
; STEP B REWRITE THE PATTERN ABOVE WITH DATA BIT(S) MODIFIED
; IN TRACKS SPECIFIED BY CRCPAT USING WRAP AROUND MODE 4
; THIS WILL GENERATE A CRC ERROR
; STEP C EXECUTE A READ REVERSE USING WRAP AROUND MODE 3
; STEP D REPEAT STEP B ABOVE. UPON COMPLETION THE DATA READ
; BACK WILL MATCH THE DATA WRITTEN IN STEP A.

```

LT16: CLR PATRN ;SELECT PATTERN # 0 (ALL 1'S)
      MOV #401,CRCPAT ;SELECT BITS TO BE ALTERED (TRACK 1)
      MOV #MSL16,EMADDR ;SET ERROR MESSAGE HEADER
      MOV #LT16A,SCOLP ;SET SCOPE LOOP

LT16A: MOV #144260,WCS1 ;SET EXPECTED VALUES UPON COMPLETION
      MOV #100,WCS2
      MOV #150600,WDS
      MOV #200,WER
      MOV #1700,UDES ;SET UNIT DESCRIPTION-NRZ,800BPI,000
                    ;PARITY & POP11 NORMAL MODE
      CLR CRCFLG ;CLEAR CRC CORRECTION FLAG
      JSR PC,WAM0 ;DO A WRAP 0 --- STEP A
      MOV #1,CRCFLG ;SET CRC ERROR CORRECTION IN PROGRESS
      MOV #BUFF,CRCDAT ;GET DATA WRITTEN BY WRAP 0
;XOR CRCPAT WITH CRCDAT
      MOV CRCDAT,XORDAT ;GET DATA TO BE MODIFIED
      MOV CRCPAT,XORPAT ;GET MODIFIER
      BIC CRCPAT,XORDAT ;CLEAR SET BITS IN DATA TO BE MODIFIED
      BIC CRCDAT,XORPAT ;CLEAR SETTING BITS
      BIS XORPAT,XORDAT ;SET CLEAR BITS IN DATA TO BE MODIFIED
      MOV XORDAT,CRCDAT ;RESTORE MODIFIED DATA

      MOV #144260,WCS1 ;SET EXPECTED VALUES UPON COMPLETION
      MOV #110,WCS2 ;OF WRAP 4
      MOV #150600,WDS
      MOV #100300,WER
      JSR PC,WAM4 ;DO A WRAP 4 --- STEP B
      NOP
      MOV #1,RDRVF ;SET TO READ REVERSE
      MOV #144276,WCS1 ;SET EXPECTED VALUES UPON COMPLETION
      MOV #110,WCS2
      MOV #150600,WDS
      MOV #1000,WER
      JSR PC,WAM3 ;DO A WRAP 3 --- STEP C
      NOP

```


1254	004202	012737	004260	001024	MOV	#4260, WCS1	
1255	004210	012737	000110	001026	MOV	#110, WCS2	
1256	004216	012737	010600	001030	MOV	#10600, WDS	
1257	004224	012737	000000	001032	MOV	#0, WER	
1258	004232	005037	001022		CLR	CRCFLG	; CLEAR CRC CORRECTION IN PROGRESS FLAG
1259	004236	004737	005364		JSR	PC, WAMP4	; GO TO WRAP 4 --- STEP D
1260							
1261	004242	006337	004302		ASL	CRCPAT	; SELECT NEXT TRACK TO BE ALTERED
1262	004246	103241			BCC	LT16A	; CONTINUE FOR ALL TRACKS
1263	004250	012737	000401	004302	MOV	#401, CRCPAT	; RESET BITS TO TRACK 1
1264	004256	005237	001002		INC	PATRN	; SELECT NEXT PATTERN
1265	004262	022737	000003	001002	CMP	#3, PATRN	; BRANCH IF NOT DONE
1266	004270	001230			BNE	LT16A	
1267	004272	004737	012014		JSR	PC, ITER	; ITERATION LOOP
1268	004276	000137	002152		JMP	TSCD2	; RETURN TO SCHEDULER
1269							
1270							
1271	004302	000000			CRCPAT: .WORD	0	; CONTAINS BITS TO BE ALTERED
1272	004304	000000			CRCDAT: .WORD	0	; CONTAINS DATA TO BE WRITTEN BY WRAP4
1273	004306	000000			XORDAT: .WORD	0	; TEMPRARY STORAGE FOR XOR
1274	004310	000000			XORPAT: .WORD	0	; TEMPORARY STORAGE FOR XOR
1275							

1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321

004312 012737 000002 001002
004320 012737 001001 004302
004326 012737 015622 000622
004334 012737 004342 000712
004342 012737 144260 001024
004350 012737 000100 001026
004356 012737 150600 001030
004364 012737 000200 001032
004372 012737 001700 000776
004400 005037 001022
004404 004737 005136
004410 000240
004412 012737 000002 001022
004420 013737 017276 004304
004426 043737 004302 004304
004434 012737 144260 001024
004442 012737 000110 001026
004450 012737 150600 001030
004456 012737 100300 001032
004464 004737 005364
004470 000240
004472 012737 000001 001006
004500 012737 144276 001024
004506 012737 000110 001026
004514 012737 150600 001030
004522 012737 001000 001032
004530 004737 005256
004534 000240
004536 012737 144260 001024
004544 012737 000110 001026
004552 012737 150600 001030
004560 012737 100100 001032
004566 005037 001022
004572 013701 004304
004576 012703 017276
004602 004737 005102
004606 004737 005364
004612 000240
004614 004737 012014
004620 000137 002152

```

; LOGIC TEST 17: CRC CORRECTION - MULTIPLE BAD TRACKS
; THIS TEST CHECKS THAT CRC ERROR CORRECTION IS NOT PERFORMED WHEN
; MULTIPLE BAD TRACKS ARE DETECTED.

LT17:  MOV      #2,PATRN      ; SELECT PATTERN #2 (125125)
      MOV      #1001,CRCPAT  ; SELECT 2 BAD TRACKS
      MOV      #MSLT17,EMADDR ; SET ERROR MESSAGE HEADER
      MOV      #LT17A,SCOLP  ; SET SCOPE LOOP ADDRESS
LT17A: MOV      #144260,WCS1  ; SET EXPECTED VALUES ON COMPLETION
      MOV      #100,WCS2     ; OF WRAP 0 BELOW
      MOV      #150600,WDS
      MOV      #200,WER
      MOV      #1700,UDES    ; SET UNIT DESCRIPTION
      CLR      CRCFLG       ; SET CRC CORRECTION NOT IN PROGRESS
      JSR      PC,WAM0      ; DO A WRAP 0 --- STEP A
      NOP
      MOV      #2,CRCFLG    ; SET CRC CORRECTION IN PROGRESS
      MOV      #WBUFF,CRCDAT ; GET DATA TO BE WRITTEN
      BIC      CRCPAT,CRCDAT ; MODIFY DATA TO BE WRITTEN
      MOV      #144260,WCS1 ; SET EXPECTED VALUES ON COMPLETION
      MOV      #110,WCS2    ; OF WRAP 4 BELOW
      MOV      #150600,WDS
      MOV      #100300,WER
      JSR      PC,WAM4      ; DO A WRAP 4 --- STEP B
      NOP
      MOV      #1,RDRVF     ; SET READ REVERSE FLAG
      MOV      #144276,WCS1 ; SET EXPECTED VALUES ON COMPLETION
      MOV      #110,WCS2    ; OF WRAP 3 BELOW
      MOV      #150600,WDS
      MOV      #1000,WER
      JSR      PC,WAM3      ; DO A WRAP 3 --- STEP C
      NOP
      MOV      #144260,WCS1 ; SET EXPECTED VALUES ON COMPLETION
      MOV      #110,WCS2    ; OF WRAP 4 BELOW
      MOV      #150600,WDS
      MOV      #100100,WER
      CLR      CRCFLG       ; CLEAR CRC IN PROGRESS FLAG
      MOV      CRCDAT,R1    ; GET DATA THAT WAS WRITTEN IN STEP B
      MOV      #WBUFF,R3    ; SET START OF WRITE BUFFER
      JSR      PC,DAT1A     ; GO SET WRITE BUFFER
      JSR      PC,WAM4      ; GO DO A WRAP 4 --- STEP D
      NOP
      JSR      PC,ITER      ; ITERATE TEST
      JMP      TSCD2       ; RETURN TO SCHEDULER
    
```

1322
 1323
 1324
 1325
 1326
 1327
 1328
 1329
 1330
 1331
 1332
 1333
 1334
 1335
 1336
 1337
 1338
 1339
 1340
 1341
 1342

004624	005037	001002	
004630	012737	144276	001024
004636	012737	000100	001026
004644	012737	150600	001030
004652	012737	100000	001032
004660	012737	015706	000622
004666	012737	001700	000776
004674	012737	004702	000712
004702	012737	000001	001006
004710	012737	000001	001020
004716	004737	005256	
004722	005037	001006	
004726	005037	001020	
004732	004737	012014	
004736	000137	002152	

```

;LOGIC TEST 20: READ REVERSE NRZ WRAP 3
;THIS TEST TESTS THAT A CRC ERROR OCCURS AFTER A READ REVERSE USING
;WRAP AROUND MODE 3 IN NRZ MODE

LT20:  CLR      PATRN      ;SET PATTERN # 0 (ALL 1'S)
        MOV      #144276,WCS1 ;SET EXPECTED VALUES ON COMPLETION
        MOV      #100,WCS2
        MOV      #150600,WDS
        MOV      #100000,WER
        MOV      #MSLT20,EMADDR ;SET ERROR MESSAGE ADDRESS
        MOV      #1700,UDES    ;SET UNIT DESCRIPTION
        MOV      #LT20A,SCOLP  ;SET SCOPE LOOP ADDRESS
LT20A:  MOV      #1,RDRVF      ;SET READ REVERSE FLAG
        MOV      #1,DCHKFL    ;SET DATA CHECK FLAG TO NOT CHACK DATA
        JSR      PC,WAN3
        CLR      RDRVF        ;CLEAR READ REVERSE FLAG
        CLR      DCHKFL      ;CLEAR DATA CHECK FLAG
        JSR      PC,ITER
        JMP      TSCD2        ;RETURN TO SCHEDULER
  
```

1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398

004742 005737 000740
004746 001434
004750 032777 000100 173612
004756 001430
004760 012704 016416
004764 004737 012714
004770 013703 001002
004774 004737 013042
005000 012705 001002
005004 012701 000002
005010 012702 000003
005014 012703 000000
005020 004737 012372
005024 112737 000001 001001
005032 113737 001002 001000
005040 012703 017276
005044 013701 001002
005050 006301
005052 004771 001034
005056 012702 000202
005062 012701 017710
005066 005021
005070 005302
005072 001375
005074 000207

005076 012701 177777
005102 012702 000202
005106 010123
005110 005302
005112 001375
005114 000207

005116 005001
005120 000770

005122 012701 125125

DSUP: TST STFLG
BEQ DSO
BIT #100,DSWR
BEQ DSO
MOV #MSG3,R4
JSR PC,TTOUT
MOV PATRN,R3
JSR PC,OCTP
MOV #PATRN,R5
MOV #2,R1
MOV #3,R2
MOV #0,R3
JSR PC,TTR
MOVB #1,MPGFL+1
MOVB PATRN,MPGFL
DSO: MOV #RBUF,R3
MOV PATRN,R1
ASL R1
JSR PC,DATBL(R1)
DS3: MOV #202,R2
MOV #RBUF,R1
DS4: CLR (R1)+
DEC R2
BNE DS4
RTS PC

;ALL ONES*****
DAT1: MOV #-1,R1
DAT1A: MOV #202,R2
IS: MOV R1,(R3)+
DEC R2
BNE IS
RTS PC

;ALL ZEROS*****
DAT2: CLR R1
BR DAT1A

;ONE/ZERO IN ALTERNATING CHARACTERS*****
DAT3: MOV #125125,R1

```

*****
:DATA SETUP ROUTINE:
:THIS ROUTINE IS USED TO GENERATE THE DATA PATTERNS
:THE PATTERN TO BE WRITTEN IS SPECIFIED BY:
:PATRN =0 ALL 1'S
:       =1 ALL 0'S
:       =2 ALTERNATE 1'S & 0'S (125252)
:       =3 ALTERNATING PARITY (177377)
*****

:SEE IF SINGLE TEST
:IF NOT: BR
:SEE IF SELECT PATTERN
:IF NOT: BR

:REQUEST PATTERN NUMBER

:PRINT PATTERN NUMBER
:GET ADDRESS OF PATRN ENTRY
:SET SIZE OF ENTRY
:SET UPPER LIMIT
:SET LOWER LIMIT
:GO GET PATTERN NUMBER
:SET FLAG
:SET PATTERN NUMBER
:R3 = ADDR OF WRITE BUFFER
:R1 = PATTERN SELECTOR
:MAKE PATTERN SELECTOR EVEN
:GO GENERATE PATTERN
:R2=BUFFER SIZE +2
:R1=READ DATA START
:CLEAR BUFFER
:SEE IF DONE ALL
:IF NOT: BR
:EXIT
    
```

K03

TMO3/TE16 CONTROL LOGIC TEST-PART II
DZTEBA.P11 31-MAR-77 17:30

MACY11 27(1006) 31-MAR-77 17:30 PAGE 35

```
1399 005126 000765          BR      DAT1A          ;LOAD BUFFER
1400
1401                          ;ALTERNATING PARITY CHARACTERS*****
1402
1403 005130 012701 177377    DAT4:  MOV      #177377,R1    ;R1=ALTERNATING PARITY DATA
1404 005134 000762          BR      DAT1A          ;GO LOAD BUFFER
1405
```

```

1406
1407 ;WRAP AROUND MODE 0 GLOBAL*****
1408
1409 005136 012737 000006 000750 WAM0: MOV #6,WAM ;SET WAM NUMBER
1410 005144 012737 000060 000752 WAM01: MOV #60,FUN
1411 005152 005037 000754 CLR DATC
1412 005156 012737 017276 000760 MOV #BUBFF,DATAD ;SET BUFFER ADDRESS
1413 005164 012737 017710 000762 MOV #RUBFF,ROAD ;SET POINTER TO READ BUFFER
1414 005172 004737 005430 JSR PC,SETUP ;GO SET UP
1415 005176 000137 006010 * JMP EXEC
1416
1417 ;WRAP AROUND MODE 1 WRITE BUFFER*****
1418
1419 005202 012737 000010 000750 WAM1: MOV #10,WAM
1420 005210 000755 BR WAM01
1421
1422 ;WRAP AROUND MODE 2 BIT FIDDLER WRITE*****
1423
1424 005212 012737 000012 000750 WAM2: MOV #12,WAM
1425 005220 012737 000060 000752 MOV #60,FUN
1426 005226 005037 000754 CLR DATC
1427 005232 012737 017276 000760 MOV #BUBFF,DATAD
1428 005240 012737 017710 000762 MOV #RUBFF,ROAD
1429 005246 004737 005430 WAM2A: JSR PC,SETUP
1430 005252 000137 006010 JMP EXEC
1431
1432 ;WRAP AROUND MODE 3 BIT FIDDLER READ*****
1433
1434 005256 012737 000014 000750 WAM3: MOV #14,WAM ;SET WAM NUMBER
1435 005264 012737 000070 000752 MOV #70,FUN ;SET FUNCTION
1436 005272 012737 017710 000760 MOV #RUBFF,DATAD ;SET BUFFER ADDRESS
1437 005300 012737 017276 000756 MOV #BUBFF,WTAD ;SET POINTER TO WRITE BUFFER
1438 005306 005737 001006 TST RDRVF
1439 005312 001411 BEQ WAM3A
1440 005314 062737 000376 000760 ADD #376,DATAD
1441 005322 062737 000377 000756 ADD #377,WTAD
1442 005330 012737 000076 000752 MOV #76,FUN ;SET READ REVERSE CODE
1443 005336 032737 000020 000776 WAM3A: BIT #20,UDES
1444 005344 001403 BEQ WAM3B
1445 005346 013737 001010 000756 WAM3B: MOV RCDP,WTAD
1446 005354 004737 005430 JSR PC,SETUP ;GO SET UP
1447 005360 000137 006010 JMP EXEC ;GO EXECUTE
1448
1449 ;WRAP AROUND MODE 4 CRC CORRECTION
1450 ;CALL: JSR PC,WAM4
1451
1452 005364 012737 000030 000750 WAM4: MOV #30,WAM ;SET MAINTENANCE MODE FUNCTION = WRAP 4
1453 005372 012737 000060 000752 MOV #60,FUN ;SET TAPE FUNCTION = WRITE FWD
1454 005400 005037 000754 CLR DATC
1455 005404 012737 004304 000760 MOV #CRCDAT,DATAD ;SET ADRS OF WRITE BUFFER
1456 005412 012737 017710 000762 MOV #RUBFF,ROAD ;SET READ BUFFER ADDRESS
1457 005420 004737 005430 JSR PC,SETUP ;GO SETUP REGISTERS
1458 005424 000137 006010 JMP EXEC ;GO EXECUTE

```

```

;REGISTER SETUP ROUTINE*****
1459
1460
1461 005430 005737 001022 SETUP: TST CRCFLG ;DO NOT DO DATA SETUP NOR INIT
1462 005434 001004 BNE IS ;IF CRC CORRECTION IS IN PROGRESS
1463 005436 022737 000030 000750 CMP #30,WAM ;DO NOT INIT IF DOING WAM 4 --- STEP D
1464 005444 001004 BNE 2S
1465 005446 012777 000011 173034 1S: MOV #11,AC1 ;DO DRIVE CLEAR IF WAM4---STEP D
1466 005454 000412 BR SET1A
1467 005456 005737 000740 2S: TST STFLG ;SEE IF SINGLE TEST
1468 005462 001403 BEQ SET0 ;IF NOT: BR
1469 005464 005737 001000 TST WPGFL ;SEE IF HAVE SELECTED PATTERN
1470 005470 001002 BNE SET1 ;IF SO: BR
1471 005472 004737 004742 SET0: JSR PC,DSUP ;GO DO DATA SETUP
1472 005476 004737 012064 SET1: JSR PC,INIT ;INIT CONTROLLER,SELECT UNIT & DRIVE
1473 ;LOAD SLAVE DESC AND MOVE OFF BOT
1474 005502 012777 177400 173006 SET1A: MOV #400,2FC ;SET FC=WCX2
1475 005510 032737 000020 000776 BIT #20,UDES ;SEE IF CORE DUMP
1476 005516 001403 BEQ SET2 ;IF NOT: BR
1477 005520 012777 177000 172770 MOV #1000,2FC ;SET FC=WCX4
1478 005526 012777 177600 172756 SET2: MOV #200,2MC ;SET MC
1479 005534 013777 000760 172752 MOV DATAD,2BA ;SET BUS ADDRESS
1480 005542 032777 010000 172750 BIT #10000,2CS ;ASSURE DRIVE THERE
1481 005550 001417 BEQ SPI ;IF SO: BR
1482 005552 032777 020000 173010 BIT #20000,2SWR ;SEE IF PRINT ERRORS
1483 005560 001004 BNE SPO1 ;IF NOT: BR
1484 005562 012704 016437 MOV #WMSG4,R4
1485 005566 004737 012714 JSR PC,TTOUT ;PRINT NON-EXISTANT DRIVE
1486 005572 032777 100000 172770 SPO1: BIT #100000,2SWR ;SEE IF HALT ON ERROR
1487 005600 001401 BEQ SPO ;IF NOT: BR
1488 005602 000000 HALT
1489 005604 000137 005476 SPO: JMP SET1 ;RESETUP
1490 005610 022737 000014 000750 SPI: CMP #14,WAM ;SEE IF WAM 3
1491 005616 001031 BNE SPIB ;IF NOT: BR
1492 005620 117737 173132 000754 MOVB 2WTAD,DATC ;GET FIRST CHAR
1493 005626 042737 177400 000754 BIC #177400,DATC
1494 005634 000337 000754 SWAB DATC
1495 005640 052737 000200 000754 BIS #200,DATC ;SET PARITY
1496 005646 005737 001006 TST RDRVF ;SEE IF READ REVERSE
1497 005652 001403 BEQ SPIA ;IF NOT: BR
1498 005654 005337 000756 DEC WTAD ;DECREMENT POINTER
1499 005660 000410 BR SPIB
1500 005662 005237 000756 SPIA: INC WTAD ;BUMP POINTER
1501 005666 032737 000020 000776 BIT #20,UDES ;SEE IF CORE DUMP
1502 005674 001402 BEQ SPIB ;IF NOT: BR
1503 005676 005237 000756 INC WTAD ;BUMP POINTER AGAIN
1504 005702 053777 000776 172632 SPIB: BIS UDES,2TC ;SET UNIT DESCRIPTION (DEN,PAR,FMT)
1505 005710 052777 000001 172616 BIS #1,2MR ;SET MAINT MODE
1506 005716 053777 000750 172610 BIS WAM,2MR ;SET WAM
1507 005724 053777 000754 172602 BIS DATC,2MR ;SET DATA
1508 005732 013777 000752 172550 MOV FUN,2C1 ;SET FUNCTION
1509 005740 032777 040000 172554 BIT #40000,2DS ;ASSURE NO ERROR
1510 005746 001002 BNE SP3 ;IF ERROR: BR
1511 005750 000240 NOP
1512 005752 000207 RTS PC ;RETURN
1513 005754 032777 020000 172606 SP3: BIT #20000,2SWR ;SEE IF PRINT ERRORS
1514 005762 001004 BNE SP4 ;IF NOT: BR

```

1515	005764	012704	016400			MOV	#WMSG2,R4		
1516	005770	004232	012714			JSB	PC,TTOUT		:PRINT SETUP ERROR
1517	005774	032777	100000	172566	SP4:	BIT	#100000,SWR		:SEE IF HALT ON ERROR
1518	006002	001401				BEG	SP5		:IF NOT: BR
1519	006004	000000				HALT			
1520	006006	000207			SP5:	RTS	PC		:RETURN


```

;EXECUTE WAM ROUTINE*****
1521
1522
1523 006010 000240 EXEC: NOP
1524 006012 000240 NOP
1525 006014 032777 000040 172512 BIT #40,2MR
1526 006022 001403 BEQ EX0 #40,2MR ;ASSURE MAINT CLOCK IS ZERO
1527 006024 042777 000040 172502 BIC #40,2MR ;IF NOT: CLEAR IT
1528 006032 022737 000010 000750 EX0: CMP #10,WAM ;SEE IF WAM 1 OR 2 OR 3
1529 006040 003402 BLE 2$
1530 006042 000137 006416 1$: JMP EXM2 ;GO DO WAM 0
1531 006046 022737 000030 000750 2$: CMP #30,WAM ;BRANCH IF WRAP AROUND MODE 4
1532 006054 001772 BEQ 1$
1533 006056 052777 000001 172424 EX1: BIS #1,2C1 ;SET GO BIT
1534 006064 005000 CLR R0
1535 006066 012701 000002 MOV #2,R1 ;SET DELAY
1536 006072 032777 100000 172442 EX1A: BIT #100000,2TC ;SEE IF ALPHA
1537 006100 001404 BEQ EX2 ;IF 50: BR
1538 006102 005300 DEC R0
1539 006104 001372 BNE EX1A ;AWAIT ALPHA
1540 006106 005301 DEC R1
1541 006110 001370 BNE EX1A
1542 006112 005077 172450 EX2: CLR 2PSW
1543 006116 012701 000400 MOV #400,R1 ;SET NUMBER OF CLKS
1544 006122 032737 000020 000776 BIT #20,UDES ;SEE IF CORE DUMP
1545 006130 001402 BEQ EX3 ;IF NOT: BR
1546 006132 012701 001000 MOV #1000,R1 ;SET CLOCKS LMCX4
1547 006136 022737 000014 000750 EX3: CMP #14,WAM ;SEE IF WAM 3
1548 006144 001412 BEQ EX5A ;IF 50: BR
1549 006146 032737 002000 000776 BIT #2000,UDES ;SEE IF PE
1550 006154 001404 BEQ EX5 ;IF NOT PE: BR
1551 006156 006301 ASL R1
1552 006160 062701 000246 ADD #246,R1 ;SET TO ALLOW FOR PRE/POSTAMBLE
1553 006164 000402 BR EX5A
1554 006166 062701 000010 000750 EX5: ADD #10,R1 ;ADD CLOCKS FOR CRC AND LRC
1555 006172 022737 000014 000750 EX5A: CMP #14,WAM ;SEE IF WAM 3
1556 006200 001053 BNE EX5C ;IF NOT: BR
1557 006202 117700 172550 MOVB 2MTAD,R0
1558 006206 042700 177400 BIC #177400,R0
1559 006212 005737 001006 TST RDRVF ;SEE IF REVERSE
1560 006216 001403 BEQ EX5A1 ;IF NOT: BR
1561 006220 005337 000756 DEC WTAD ;DEC POINTER
1562 006224 000416 BR EX5B
1563 006226 005237 000756 EX5A1: INC WTAD
1564 006232 032737 000020 000776 BIT #20,UDES ;SEE IF CORE DUMP
1565 006240 001410 BEQ EX5B ;IF NOT: BR
1566 006242 005237 000756 INC WTAD ;BUMP POINTER
1567 006246 005777 172504 TST 2MTAD ;SEE IF END
1568 006252 001003 BNE EX5B ;IF NOT: BR
1569 006254 162737 000010 000756 SUB #10,WTAD ;RESTORE POINTER
1570 006262 052777 000040 172244 EX5B: BIS #40,2MR ;CLOCK UP
1571 006270 017702 172240 MOV 2MR,R2 ;READ MR
1572 006274 042702 177600 BIC #177600,R2 ;MASK OUT DATA
1573 006300 000300 SWAB R0 ;POSITION DATA
1574 006302 022700 177000 CMP #177000,R0 ;SEE IF PATTERN 3
1575 006306 001404 BEQ 2$ ;IF 50: BR
1576 006310 000240 NOP
  
```

1577	006312	000240								
1578	006314	052700	000200		1S:	BIS	#200, R0			; SET ODD PARITY
1579	006320	050002			2S:	BIS	R0, R2			; LOAD NEW DATA
1580	006322	010277	172206			MOV	R2, 2MR			; CLOCK DOWN AND LOAD NEW DATA
1581	006326	000426				BR	EX50			
1582	006330	052777	000040	172176	EXSC:	BIS	#40, 2MR			; CLOCK UP
1583	006336	042777	000040	172170		BIC	#40, 2MR			; CLOCK DOWN
1584	006344	017700	172164			MOV	2MR, R0			; GET MR
1585	006350	000300				SWAB	R0			
1586	006352	032737	000010	000776		BIT	#10, UDES			; SEE IF EVEN PAR
1587	006360	001405				BEG	EXSC0			; IF NOT: BR
1588	006362	010077	172374			MOV	R0, 2RDAD			
1589	006366	005237	000762			INC	RDAD			
1590	006372	000402				BR	EXSC1			
1591	006374	110077	172362		EXSC0:	MOVB	R0, 2RDAD			; PUT CHAR IN CORE
1592	006400	005237	000762		EXSC1:	INC	RDAD			
1593	006404	000240			EX50:	NOP				
1594	006406	005301				DEC	R1			; SEE IF DONE CLKS
1595	006410	001270				BNE	EX5A			; IF NOT: BR
1596	006412	000137	011570			JMP	EORP			; GO DO EOR

```

1597                                     ;EXECUTE WAM 0*****
1598
1599 006416 000240                               EXW2:  NOP
1600 006420 012737 006604 000670             MOV     #EXW2H,RTRN      ;SET INTERRUPT RETURN ADDRESS
1601 006426 012701 000200                     MOV     #200,R1          ;SET NUMBER OF CLOCKS = FC/2
1602 006432 032737 002000 000776             BIT     #2000,UDES      ;SEE IF PE
1603 006440 001402                               BEQ     EXW2A            ;IF NOT: BR
1604 006442 012701 000100                     MOV     #100,R1         ;ELSE SET CLKS = FC/4
1605 006446 012702 017710                     EXW2A: MOV     #RBUF,R2      ;SET BUFFER ADDRESS
1606 006452 022737 000030 000750             CMP     #30,WAM        ;BRANCH IF NOT WRAP 4
1607 006460 001003                               BNE     1$
1608 006462 052777 000010 172030             BIS     #10,ACS        ;SET INHIBIT BUS ADDRESS INCREMENT
1609 006470 012777 000161 172012 1$:      MOV     #161,AC1       ;SET WRITE COMMAND AND GO
1610 006476 005077 172064                     CLR     APSW
1611 006502 032777 000040 172024             EXW2B: BIT     #40,AMR
1612 006510 001774                               BEQ     EXW2B            ;AWAIT CLOCK UP
1613 006512 017722 172016                     MOV     AMR,(R2)+       ;GET DATA
1614 006516 032777 000040 172010             EXW2C: BIT     #40,AMR
1615 006524 001374                               BNE     EXW2C            ;AWAIT CLOCK DOWN
1616 006526 017722 172002                     MOV     AMR,(R2)+       ;GET DATA
1617 006532 005301                               DEC     R1
1618 006534 001362                               BNE     EXW2B            ;SEE IF DONE ALL
1619 006536 012701 000003                     EXW2E: MOV     #3,R1
1620 006542 005000                               CLR     R0
1621 006544 005300                               EXW2F: DEC     R0
1622 006546 001376                               BNE     EXW2F
1623 006550 005301                               DEC     R1
1624 006552 001374                               BNE     EXW2F            ;DELAY
1625 006554 032777 020000 172006             BIT     #20000,ASWR     ;SEE IF ERROR PRINT
1626 006562 001004                               BNE     EXW2G            ;IF NOT: BR
1627 006564 012704 016642                     MOV     #MSG24,R4
1628 006570 004737 012714                     JSR     PC,TTOUT
1629 006574 005777 171770                     EXW2G: TST     ASWR
1630 006600 100001                               BPL     EXW2H
1631 006602 000000                               HALT
1632 006604 000240                               EXW2H: NOP
1633 006606 012701 017710                     MOV     #RBUF,R1        ;GET START OF READ BUFFER
1634 006612 012700 000400                     MOV     #400,R0        ;SET SIZE
1635 006616 010102                               MOV     R1,R2
1636 006620 012203                     EXW2J: MOV     (R2)+,R3
1637 006622 000303                               SWAB   R3
1638 006624 032737 000010 000776             BIT     #10,UDES        ;SEE IF EVEN PAR
1639 006632 001402                               BEQ     EXW2JO           ;IF NOT: BR
1640 006634 010321                               MOV     R3,(R1)+       ;SAVE PAR + DATA
1641 006636 000401                               BR     EXW2J1
1642 006640 110321                     EXW2JO: MOVB   R3,(R1)+ ;ASSEMBLE DATA IN BYTES
1643 006642 005300                     EXW2J1: DEC     R0
1644 006644 001365                               BNE     EXW2J
1645 006646 032777 000200 171714             BIT     #200,ASWR      ;CONTINUE FOR ALL
1646 006654 001002                               BNE     EXW2K            ;SEE IF STATUS CHECK
1647 006656 004737 006704                     JSR     PC,WSTCK        ;IF NOT: BR
1648 006662 000240                               EXW2K: NOP
1649 006664 032777 000400 171676             BIT     #400,ASWR      ;ELSE GO CHECK STATUS
1650 006672 001002                               BNE     EXW2X
1651 006674 004737 007274                     JSR     PC,DCHK
1652 006700 000240                     EXW2X: NOP

```

E04

TMD3/TE16 CONTROL LOGIC TEST-PART II
DZTEBA.P11 31-MAR-77 17:30

MACY11 27(1006) 31-MAR-77 17:30 PAGE 42

1653 006702 000207

RTS PC

;EXIT

```

1654
1655
1656
1657 006704 000240
1658 006706 005037 000772
1659 006712 005037 000620
1660 006716 022737 014575 000622
1661 006724 001404
1662 006726 022737 015465 000622
1663 006734 001025
1664 006736 005737 001000 2$:
1665 006742 001405
1666 006744 122737 000003 001000
1667 006752 001016
1668 006754 000404
1669 006756 022737 000003 001002 3$:
1670 006764 001011
1671 006766 052737 140000 001024 4$:
1672 006774 052737 140000 001030
1673 007002 052737 000100 001032
1674 007010 012737 016464 000672 5$:
1675 007016 017702 171466
1676 007022 013705 001024
1677 007026 004737 007150
1678 007032 012737 016511 000672
1679 007040 017702 171454
1680 007044 013705 001026
1681 007050 053705 000624
1682 007054 004737 007150
1683 007060 012737 016517 000672
1684 007066 017702 171430
1685 007072 013705 001030
1686 007076 004737 007150
1687 007102 012737 016524 000672 1$:
1688 007110 017702 171410
1689 007114 000240
1690 007116 000240
1691 007120 013705 001032
1692 007124 004737 007150
1693 007130 005737 000772
1694 007134 001456
1695 007136 005777 171426
1696 007142 100053
1697 007144 000000
1698 007146 000451
1699 007150 000240
1700 007152 020205
1701 007154 001446
1702 007156 005237 000772
1703 007162 032777 020000 171400
1704 007170 001040
1705 007172 005737 000620
1706 007176 001010
1707 007200 013704 000622
1708 007204 004737 012714
1709 007210 012704 016626

```

;WRAP AROUND STATUS CHECK ROUTINE*****

```

WSTCK: NOP
CLR SERFL ;CLEAR ERROR FLAG
CLR HDRFL ;CLEAR HEADER FLAG
CMP #MSLT2,EMADDR ;SEE IF TEST 2
BEQ 2$ ;IF SO: BR
CMP #MSLT15,EMADDR ;SEE IF TEST 15
BNE 5$ ;IF NOT: BR
TST WPGFL ;SEE IF SINGLE PATTERN
BEQ 3$ ;IF NOT: BR
CMPB #3,WPGFL ;SEE IF PATTERN 3
BNE 5$ ;IF NOT: BR
BR 4$ ;ELSE GO DO EXPT CHANGE
CMP #3,PATRN ;SEE IF PATTERN 3
BNE 5$ ;IF NOT: BR
BIS #140000,WCS1
BIS #140000,WDS
BIS #100,WER ;SET EXPT PARITY ERROR
MOV #MSG6,ERADD ;SET CODE=CS1
MOV @C1,R2 ;GET RCVD CS1
MOV WCS1,R5 ;GET EXPT CS1
JSR PC,WSTG ;GO CHK
MOV #MSG6D,ERADD ;SET CODE=CS2
MOV @CS,R2 ;SET RCVD CS2
MOV WCS2,R5 ;GET EXPT CS2
BIS DRVN,R5 ;SET DRIVE NUMBER IN EXPT CS2
JSR PC,WSTG ;GO CHK
MOV #MSG6E,ERADD ;SET CODE=DS
MOV @DS,R2 ;SET RCVD DS
MOV WDS,R5 ;GET EXPT DS
JSR PC,WSTG ;GO CHK
MOV #MSG6F,ERADD ;SET CODE=ER
MOV @ER,R2 ;GET RCVD ER
NOP
NOP
MOV WER,R5 ;GET EXPT ER
JSR PC,WSTG ;GO CHK
TST SERFL ;SEE IF ANY ERRORS
BEQ WSTX ;IF NOT: BR
TST @SWR ;SEE IF HALT ON ERROR
BPL WSTX ;IF NOT: BR
HALT
BR WSTX ;CONTINUE
WSTG: NOP
CMP R2,R5 ;SEE IF EXPT=RCVD
BEQ WSTX ;IF SO: BR
INC SERFL ;SET ERROR FLAG
BIT #20000,@SWR ;SEE IF PRINT ERRORS
BNE WSTX ;IF NOT: BR
TST HDRFL ;SEE IF DONE HEADER
BNE WSTGO ;IF SO: BR
MOV EMADDR,R4
JSR PC,TTOUT ;PRINT TEST HEADER
MOV #MSG23,R4

```

1710	007214	004737	012714		JSR	PC, TTOUT	;PRINT STATUS TAG
1711	007220	012737	000001	000620	WSTGO:	MOV #1, HDRFL	;SET HEADER FLAG
1712	007226	013704	000672		MOV ERADD, R4		
1713	007232	004737	012714		JSR PC, TTOUT	;PRINT CODE	
1714	007236	012704	014165		MOV #MSG12, R4		
1715	007242	004737	012714		JSR PC, TTOUT	;PRINT EXPT TAG	
1716	007246	010503			MOV R5, R3		
1717	007250	004737	013042		JSR PC, OCTP	;PRINT EXPT STATUS	
1718	007254	012704	014174		MOV #MSG13, R4		
1719	007260	004737	012714		JSR PC, TTOUT	;PRINT RCVD TAG	
1720	007264	010203			MOV R2, R3		
1721	007266	004737	013042		JSR PC, OCTP	;PRINT RCVD STATUS	
1722	007272	000207			WSTX: RTS	;RETURN	
1723							

```

1724
1725
1726
1727 007274 000240
1728 007276 005737 001022
1729 007302 001402
1730 007304 000137 010134
1731 007310 005737 001020
1732 007314 001373
1733 007316 005037 000620
1734 007322 005037 000766
1735 007326 005037 000774
1736 007332 032737 000010 000776
1737 007340 001402
1738 007342 000137 010156
1739 007346 022737 000010 000750
1740 007354 001006
1741 007356 032737 002000 000776
1742 007364 001402
1743 007366 000137 010660
1744 007372 012700 177400
1745 007376 022737 000012 000750
1746 007404 001006
1747 007406 032737 000020 000776
1748 007414 001402
1749 007416 012700 177000
1750 007422 022737 000030 000750
1751 007430 001404
1752 007432 022737 000006 000750
1753 007440 001007
1754 007442 012700 177744
1755 007446 012701 017306
1756 007452 005037 000766
1757 007456 000431
1758 007460 022737 000012 000750
1759 007466 001021
1760 007470 032737 002000 000776
1761 007476 001415
1762 007500 012700 177653
1763 007504 012737 000001 000764
1764 007512 004737 007532
1765 007516 004737 011160
1766 007522 005037 000764
1767 007526 000137 010116
1768 007532 005037 000766
1769 007536 012701 017276
1770 007542 012702 017710
1771 007546 032737 000020 000776
1772 007554 001416
1773 007556 022737 000012 000750
1774 007564 001011
1775 007566 005737 001002
1776 007572 001003
1777 007574 012701 001056
1778 007600 000404
1779 007602 012701 001044
;DATA CHECK ROUTINE*****
DCHK: NOP
TST CRCFLG ;DO NOT DO A DATA CHACK
BEQ ZS ;IF CRC CORRECTION IN PROGRESS
1S: JMP DCHKX1
2S: TST DCHKFL ;BRANCH IF DATA IS NOT TO BE CHECKED
BNE 1S
CLR HDREL ;CLEAR HEADER FLAG
CLR DERFL ;CLEAR DATA ERROR FLAG
CLR CRCNT ;CLEAR CHAR CNTR
BIT #10,UDES ;SEE IF EVEN PARITY
BEQ DCHKAD ;IF NOT: BR
JMP DCHKE ;ELSE GO CHECK EVEN
DCHKAD: CMP #10,WAM ;SEE IF WAM 1
BNE DCHKA ;IF NOT: BR
BIT #2000,UDES ;SEE IF PE
BEQ DCHKA ;IF NOT: BR
JMP PRCHK ;GO CHK DATA
DCHKA: MOV #400,R0 ;SET NUMBER OF CHARACTERS
CMP #12,WAM
BNE DCHKA1 ;IF NOT WRAP 2: BR
BIT #20,UDES
BEQ DCHKA1 ;IF NOT CORE DUMP: BR
MOV #1000,R0
DCHKA1: CMP #30,WAM ;BRANCH IF WRAP 4
BEQ 1S
CMP #6,WAM ;SEE IF WRAP 0
BNE DCHKA2 ;IF NOT: BR
1S: MOV #34,R0 ;SET NUMBER OF CHARACTERS READ
MOV #BUBFF+10,R1 ;SET POINTER
CLR DERFL ;CLEAR DATA ERROR FLAG
BR DCHKB0
DCHKA2: CMP #12,WAM ;SEE IF WRAP 2
BNE DCHKB ;IF NOT: BR
BIT #2000,UDES ;SEE IF PE
BEQ DCHKB ;IF NOT: BR
MOV #125,R0 ;POINT TO START OF DATA
MOV #1,W2FLG ;SET WRAP 2 FLAG
JSR PC,DCHKB ;GO CHECK DATA
JSR PC,WIDCHK ;GO CHECK WRAP 1 DATA
CLR W2FLG
JMP DCHKX
DCHKB: CLR DERFL
MOV #BUBFF,R1 ;SET GOOD POINTER
DCHKB0: MOV #RUBFF,R2 ;SET READ POINTER
BIT #20,UDES ;SEE IF CORE DUMP
BEQ DCHKD ;IF NOT: BR
CMP #12,WAM ;SEE IF WAM 2
BNE DCHKD ;IF NOT: BR
TST PATRN ;SEE IF PATTERN 0
BNE DCHKC ;IF NOT: BR
MOV #WCDPO,R1 ;SET CORE DUMP PATTERN 0
BR DCHKD ;GO CHECK DATA
DCHKC: MOV #WCDP2,R1 ;SET CORE DUMP WRITE PATTERN 2

```

1780	007606	000401				BR	DCHK0		;GO CHECK DATA
1781	007610	000240				DCHK0: NOP			
1782	007612	121112				DCHK0: CMPB	(R1), (R2)		;SEE IF DATA OK
1783	007614	001466				BEG	DCHK2		;IF SO: BR
1784	007616	032777	020000	170744		BIT	#20000, JSWR		;SEE IF PRINT ERRORS
1785	007624	001062				BNE	DCHK2		;IF NOT: BR
1786	007626	005737	000620			TST	HDRFL		;SEE IF DONE HEADER
1787	007632	001004				BNE	DCHK1		;IF SO: BR
1788	007634	013704	000622			MOV	EMADDR, R4		
1789	007640	004737	012714			JSR	PC, TTOUT		;PRINT HEADER
1790	007644	005737	000766			TST	DERFL		;SEE IF FIRST ERROR
1791	007650	001014				BNE	DCHK1A		;IF NOT: BR
1792	007652	012704	016574			MOV	#MSG16, R4		
1793	007656	004737	012714			JSR	PC, TTOUT		;PRINT DATA ERROR TAG
1794	007662	012704	017021			MOV	#MSG32, R4		
1795	007666	004737	012714			JSR	PC, TTOUT		;PRINT PATRN TAG
1796	007672	013703	001002			MOV	PATRN, R3		
1797	007676	004737	013042			JSR	PC, OCTP		;PRINT PATTERN NUMBER
1798	007702	012737	000001	000620		MOV	#1, HDRFL		;SET HEADER FLAG
1799	007710	012737	000001	000766		MOV	#1, DERFL		;SET DATA ERROR FLAG
1800	007716	012704	016620			MOV	#MSG21, R4		
1801	007722	004737	012714			JSR	PC, TTOUT		;PRINT CHARACTER NUMBER TAG
1802	007726	013703	000774			MOV	CRCNT, R3		
1803	007732	004737	013042			JSR	PC, OCTP		;PRINT CHARACTER NUMBER
1804	007736	012704	016606			MOV	#MSG17, R4		
1805	007742	004737	012714			JSR	PC, TTOUT		;PRINT GOOD TAG
1806	007746	111103				MOVB	(R1), R3		
1807	007750	004737	013270			JSR	PC, DOUT		;PRINT GOOD DATA
1808	007754	012704	016613			MOV	#MSG20, R4		
1809	007760	004737	012714			JSR	PC, TTOUT		;PRINT BAD TAG
1810	007764	111203				MOVB	(R2), R3		
1811	007766	004737	013270			JSR	PC, DOUT		;PRINT BAD DATA
1812	007772	005737	000764			TST	W2FLG		;SEE IF WRAP 2 NRZ
1813	007776	001020				BNE	DCHK2B		;IF SO: BR
1814	010000	005201				INC	R1		;BUMP POINTER
1815	010002	032737	000020	000776		BIT	#20, UDES		;SEE IF CORE DUMP
1816	010010	001413				BEG	DCHK2B		;IF NOT: BR
1817	010012	022737	000012	000750		CMP	#12, WAM		;SEE IF WAM 2
1818	010020	001006				BNE	DCHK2A		;IF NOT: BR
1819	010022	005201				INC	R1		;BUMP POINTER
1820	010024	005711				TST	(R1)		;SEE IF END OF PATTERN
1821	010026	001004				BNE	DCHK2B		;IF NOT: BR
1822	010030	162701	000010			SUB	#10, R1		;RESET POINTER TO START OF PATTERN
1823	010034	000401				BR	DCHK2B		;CONTINUE CHECK
1824	010036	000240				DCHK2A: NOP			
1825	010040	005202				DCHK2B: INC	R2		
1826	010042	022737	000030	000750		CMP	#30, WAM		;BRANCH IF WAM 4
1827	010050	001404				BEG	15		
1828	010052	022737	000006	000750		CMP	#6, WAM		;SEE IF WAM 0
1829	010060	001002				BNE	DCHK3		;IF NOT: BR
1830	010062	062701	000010			ADD	#10, R1		;BUMP POINTER
1831	010066	005237	000774			INC	CRCNT		;BUMP CHAR CNTR
1832	010072	032777	000400	170470		BIT	#400, JSWR		;SEE IF CONT DATA CHK
1833	010100	001006				BNE	DCHKX		;IF NOT: BR
1834	010102	005200				INC	R0		;SEE IF DONE
1835	010104	001242				BNE	DCHK0		;IF NOT: BR

1836	010106	005737	000764
1837	010112	001401	
1838	010114	000207	
1839	010116	005777	170446
1840	010122	100004	
1841	010124	005737	000766
1842	010130	001401	
1843	010132	000000	
1844	010134	005037	000774
1845	010140	005037	000620
1846	010144	005037	000766
1847	010150	005037	000770
1848	010154	000207	

	TST	W2FLG
	BEG	DCHKX
	RTS	PC
DCHKX:	TST	2SMR
	BPL	DCHKX1
	TST	DERFL
	BEG	DCHKX1
	HALT	
DCHKX1:	CLR	CRCNT
	CLR	HDRFL
	CLR	DERFL
	CLR	PREFL
	RTS	PC

```

;SEE IF HALT ON ERROR
;IF NOT: BR
;SEE IF DATA ERROR OCCURED
;IF NOT: BR

;CLEAR CHAR CNTR
;CLEAR HEADER FLAG
;CLEAR DATA ERROR FLAG
;CLEAR PREAMBLE FLAG
;RETURN

```

```

1849
1850
1851
1852 010156 000240
1853 010160 022737 000006 000750
1854 010166 001005
1855 010170 012700 177744
1856 010174 012701 017306
1857 010200 000404
1858 010202 012700 177400
1859 010206 012701 017276
1860 010212 012702 017710
1861 010216 111105
1862 010220 005003
1863 010222 012704 000010
1864 010226 032705 000001
1865 010232 001401
1866 010234 005203
1867 010236 005304
1868 010240 001402
1869 010242 005005
1870 010244 000770
1871 010246 000240
1872 010250 111105
1873 010252 042705 177400
1874 010256 005703
1875 010260 001003
1876 010262 012705 100020
1877 010266 000405
1878 010270 032703 000001
1879 010274 001402
1880 010276 052705 100000
1881 010302 042712 077400
1882 010306 020512
1883 010310 001477
1884 010312 032777 020000 170250
1885 010320 001073
1886 010322 005737 000620
1887 010326 001004
1888 010330 013704 000622
1889 010334 004737 012714
1890 010340 005737 000766
1891 010344 001014
1892 010346 012704 016574
1893 010352 004737 012714
1894 010356 012704 017021
1895 010362 004737 012714
1896 010366 013703 001002
1897 010372 004737 013042
1898 010376 000240
1899 010400 012737 000001 000766
1900 010406 012737 000001 000620
1901 010414 012704 016620
1902 010420 004737 012714
1903 010424 013703 000774
1904 010430 004737 013042

```

;EVEN PARITY DATA CHECK*****

```

DCHKE: NOP
      CMP      #6, WAM      ;SEE IF WRAP 0
      BNE     1S           ;IF NOT: BR
      MOV     #8-34, R0     ;SET NUMBER OF CHARACTERS READ
      MOV     @#BUFF+10, R1 ;SET POINTER
      BR      2S
1S:   MOV     #8-400, R0    ;SET NUMBER OF CHARACTERS
      MOV     @#BUFF, R1    ;R1=START OF WRITE BUFFER
2S:   MOV     @#BUFF, R2    ;R2=START OF READ BUFFER
DCKE0: MOVB   (R1), R5     ;GET EXPT DATA
      CLR     R3
      MOV     #10, R4      ;SET NUMBER OF BITS
DCKE1: BIT     #1, R5       ;SEE IF ONE BIT
      BEQ     DCKE2        ;IF NOT: BR
      INC     R3           ;COUNT ONE BITS FOR PARITY CHECK
DCKE2: DEC     R4          ;SEE IF DONE
      BEQ     DCKE3        ;IF SO: BR
      ROR     R5           ;POINT TO NEXT BIT
      BR      DCKE1
DCKE3: NOP
      MOVB   (R1), R5     ;GET EXPT DATA
      BIC     #177400, R5  ;MASK DATA FIELD
      TST     R3
      BNE     DCKE4        ;IF NO ONE BITS SET: BR
      MOV     #100020, R5
      BR      DCKE5
DCKE4: BIT     #1, R3      ;SEE IF ODD NUMBER OF ONE BITS
      BEQ     DCKE5        ;IF NOT: BR
      BIS     #100000, R5  ;SET EVEN PARITY BIT=1
DCKE5: BIC     #77400, (R2) ;MASK DATA FIELD
      CMP     R5, (R2)     ;SEE IF DATA + PARITY GOOD
      BEQ     DCKE10       ;IF SO: BR
      BIT     #20000, #SWR ;SEE IF ERROR PRINT
      BNE     DCKE10       ;IF NOT: BR
      TST     #HORFL      ;SEE IF DONE HEADER
      BNE     DCKE6        ;IF SO: BR
      MOV     #EMADDR, R4
      JSR    PC, TTOUT     ;PRINT HEADER
DCKE6: TST     #DERFL      ;SEE IF FIRST BAD CHAR
      BNE     DCKE7        ;IF NOT: BR
      MOV     ##MSG16, R4
      JSR    PC, TTOUT     ;PRINT BAD DATA TAG
      MOV     ##MSG32, R4
      JSR    PC, TTOUT     ;PRINT PATTERN TAG
      MOV     #PATRN, R3
      JSR    PC, OCTP      ;PRINT PATTERN NUMBER
DCKE7: NOP
      MOV     #1, #DERFL   ;SET DATA ERROR FLAG
      MOV     #1, #HORFL   ;SET HEADER FLAG
      MOV     ##MSG21, R4
      JSR    PC, TTOUT     ;PRINT CHAR NUMBER TAG
      MOV     #CRCNT, R3
      JSR    PC, OCTP      ;ORINT CHAR NUMBER

```

1905	010434	012704	016606		MOV	#MSG17,R4	
1906	010440	004737	012714		JSR	PC,TTOUT	;PRINT GOOD DATA TAG
1907	010444	110503			MOVB	R5,R3	
1908	010446	004737	013270		JSR	PC,DOUT	;PRINT EXPT DATA
1909	010452	010503			MOV	R5,R3	
1910	010454	004737	010564		JSR	PC,DCKEP	;GO PRINT PARITY BIT
1911	010460	000240			NOP		
1912	010462	012704	016613		MOV	#MSG20,R4	
1913	010466	004737	012714		JSR	PC,TTOUT	;PRINT BAD TAG
1914	010472	111203			MOVB	(R2),R3	
1915	010474	004737	013270		JSR	PC,DOUT	;PRINT BAD DATA
1916	010500	011203			MOV	(R2),R3	
1917	010502	004737	010564		JSR	PC,DCKEP	;GO PRINT PARITY BIT
1918	010506	000240			NOP		
1919	010510	005201			INC	R1	
1920	010512	002737	000006	000750	DCKE10: CMP	#6,WAM	;SEE IF WRAP 0
1921	010520	001002			BNE	IS	;IF NOT: BR
1922	010522	002701	000010		ADD	#10,R1	;BUMP POINTER
1923	010526	005722			IS: TST	(R2)+	;BUMP POINTERS
1924	010530	005237	000774		INC	CRCNT	;BUMP CHAR CNTR
1925	010534	002777	000400	170026	BIT	#400,JSWR	;SEE IF CONTINUE DATA CHECK
1926	010542	001402			BEQ	DCKE11	;IF SO: BR
1927	010544	000137	010116		JMP	DCHKX	;GO TO END OF DATA CHECK
1928	010550	005200			DCKE11: INC	R0	;SEE IF DONE
1929	010552	001402			BEQ	DCKE12	;IF SO: BR
1930	010554	000137	010216		JMP	DCKED	;ELSE CONTINUE
1931	010560	000137	010116		DCKE12: JMP	DCHKX	;GO TO END OF DATA CHECK
1932	010564	000240			DCKEP: NOP		
1933	010566	012737	000240	000614	MOV	#240,TOB	
1934	010574	004737	013014		JSR	PC,TOG	;SPACE
1935	010600	012737	000260	000614	MOV	#260,TOB	;SET PAR=0
1936	010606	005703			TST	R3	;SEE IF PARITY REALLY=0
1937	010610	100002			BPL	DCKEPO	;IF SO: BR
1938	010612	005237	000614		INC	TOB	;ELSE SET TO 1
1939	010616	004737	013014		DCKEPO: JSR	PC,TOG	;PRINT PARITY BIT
1940	010622	000207			RTS	PC	;RETURN
1941							

```

1942
1943
1944
1945 010624 012700 000051 PSCHK: MOV #51,R0 ;SET SIZE OF POSTAMBLE
1946 010630 012701 017154 MOV #POST,R1 ;SET POINTER TO POSTAMBLE
1947 010634 005037 000620 CLR HDRFL ;CLEAR HEADER FLAG
1948 010640 005037 000774 CLR CRCNT ;CLEAR CHAR CNTR
1949 010644 005037 000766 CLR DERFL ;CLEAR DATA ERROR FLAG
1950 010650 000240 NOP
1951 010652 000240 NOP
1952 010654 000137 010676 JMP P00 ;GO CHECK POSTAMBLE
1953
1954 010660 012700 000051 PRCHK: MOV #51,R0 ;SET SIZE OF PREAMBLE
1955 010664 012701 017032 MOV #PRE,B1 ;SET POINTER TO PREAMBLE
1956 010670 012702 017710 MOV #RBUF,R2 ;SET POINTER TO START OF READ BUFFER
1957 010674 022122 CMP (R1)+,(R2)+ ;BUMP ADDRESS POINTERS
1958 010676 121112 P00: CMPB (R1),(R2) ;CHECK DATA
1959 010700 001004 BNE PD1 ;IF NOT GOOD: BR
1960 010702 126162 000001 000001 CMPB 1(R1),1(R2) ;COMPARE COMPLIMENT BYTE
1961 010710 001477 BEQ P05 ;IF GOOD: BR
1962 010712 032777 020000 167650 PD1: BIT #20000,#SWR ;SEE IF PRINT INHIBIT
1963 010720 001073 BNE P05 ;IF SO: BR
1964 010722 005737 000620 TST HDRFL ;SEE IF DONE HEADER
1965 010726 001020 BNE PD4 ;IF SO: BR
1966 010730 013704 000622 MOV EMADDR,R4
1967 010734 004737 012714 JSR PC,TTOUT ;PRINT TEST HEADER
1968 010740 005737 000770 TST PRFL ;SEE IF PREAMBLE CHECK
1969 010744 001403 BEQ PD2 ;IF NOT: BR
1970 010746 012704 016745 MOV #MSG29,R4 ;SET POSTAMBLE HEADER
1971 010752 000402 BR PD3
1972 010754 012704 016727 PD2: MOV #MSG28,R4 ;SET PREAMBLE HEADER
1973 010760 004737 012714 PD3: JSR PC,TTOUT ;PRINT HEADER
1974 010764 005237 000620 INC HDRFL
1975 010770 012704 016620 PD4: MOV #MSG21,R4
1976 010774 004737 012714 JSR PC,TTOUT ;PRINT CHAR NUMBER TAG
1977 011000 013703 000774 MOV CRCNT,R3 ;PRINT CHAR NUMBER
1978 011004 004737 013042 JSR PC,OCIP
1979 011010 012704 016606 MOV #MSG17,R4 ;PRINT GOOD TAG
1980 011014 004737 012714 JSR PC,TTOUT
1981 011020 116103 000001 MOVB 1(R1),R3 ;PRINT GOOD CHAR
1982 011024 004737 013270 JSR PC,DOUT
1983 011030 012737 000240 000614 MOV #240,T0B
1984 011036 004737 013014 JSR PC,T0G
1985 011042 111103 MOVB (R1),R3
1986 011044 004737 013270 JSR PC,DOUT ;PRINT COMPLEMENT
1987 011050 012704 016613 MOV #MSG20,R4
1988 011054 004737 012714 JSR PC,TTOUT ;PRINT BAD TAG
1989 011060 116203 000001 MOVB 1(R2),R3
1990 011064 004737 013270 JSR PC,DOUT ;PRINT BAD CHAR
1991 011070 012737 000240 000614 MOV #240,T0B
1992 011076 004737 013014 JSR PC,T0G
1993 011102 111203 MOVB (R2),R3
1994 011104 004737 013270 JSR PC,DOUT ;PRINT COMPLEMENT
1995 011110 022122 P05: CMP (R1)+,(R2)+ ;BUMP ADDRESS POINTERS
1996 011112 005237 000774 INC CRCNT ;BUMP CHAR NUMBER
1997 011116 005300 DEC R0 ;SEE IF DONE

```

1998	011120	001266	
1999	011122	005737	000770
2000	011126	001402	
2001	011130	000137	010116
2002	011134	005237	000770
2003	011140	005037	000620
2004	011144	005037	000774
2005	011150	005037	000766
2006	011154	000137	011160
2007			

PD6:

BNE	P00
TST	PREFL
BEQ	PD6
JMP	DCHKX
INC	PREFL
CLR	HDRFL
CLR	CRCNT
CLR	DERFL
JMP	WIDCHK

```

:IF NOT: BR
:SEE IF PREAMBLE
:IF SO: BR
:GO TO EXIT ROUTINE
:SET PREAMBLE FLAG
:CLEAR HEADER FLAG
:CLEAR CHAR CNTR
:CLEAR DATA ERROR FLAG
:GO CHECK WRAP 1 DATA
  
```

```

2008
2009
2010 ;WAM 1 PE DATA CHECK*****
2011 011160 012700 177400 WIDCHK: MOV #400,R0 ;SET NUMBER OF CHAR TO CHECK
2012 011164 012701 017276 MOV #WBUF,R1 ;SET WRITE DATA POINTER
2013 011170 012702 017710 MOV #RBUF,R2 ;SET READ DATA POINTER
2014 011174 062702 000124 ADD #124,R2 ;POINT TO START OF DATA
2015 011200 005737 000764 TST W2FLG ;SEE IF WRAP 2
2016 011204 001401 BEQ WID0 ;IF NOT WAM 2: BR
2017 011206 005302 DEC R2 ;RESET POINTER
2018 011210 111105 WID0: MOVB (R1),R5
2019 011212 120512 CMPB R5,(R2) ;CHECK DATA
2020 011214 001007 BNE WID1 ;IF NOT GOOD:BR
2021 011216 005737 000764 TST W2FLG ;SEE IF WRAP 2
2022 011222 001001 BNE WID0A ;IF SO: BR
2023 011224 105105 COMB R5 ;COMPLIMENT EXPT DATA
2024 011226 120562 000001 WID0A: CMPB R5,1(R2) ;CHECK COMPLIMENT DATA
2025 011232 001510 BEQ WID3 ;IF GOOD: BR
2026 011234 032777 020000 167326 WID1: BIT #20000,25MR ;SEE IF PRINT INHIBIT
2027 011242 001104 BNE WID3 ;IF SO: BR
2028 011244 005737 000620 TST HDRFL ;SEE IF DONE HEADER
2029 011250 001020 BNE WID2 ;IF SO: BR
2030 011252 013704 000622 MOV EMADR,R4
2031 011256 004737 012714 JSR PC,TTOUT ;PRINT TEST HEADER
2032 011262 012704 016574 MOV #MSG16,R4
2033 011266 004737 012714 JSR PC,TTOUT ;PRINT BAD DATA TAG
2034 011272 012704 017021 MOV #MSG32,R4
2035 011276 004737 012714 JSR PC,TTOUT ;PRINT PATRN TAG
2036 011302 013703 001002 MOV PATRN,R3
2037 011306 004737 013042 JSR PC,OC1P ;PRINT PATTERN NUMBER
2038 011312 012737 000001 000620 WID2: MOV #1,HDRFL ;SET HEADER FLAG
2039 011320 012704 016620 MOV #MSG21,R4
2040 011324 004737 012714 JSR PC,TTOUT ;PRINT CHAR NUMBER TAG
2041 011330 013703 000774 MOV CRCNT,R3
2042 011334 004737 013042 JSR PC,OC1P ;PRINT CHAR NUMBER
2043 011340 012704 016606 MOV #MSG17,R4
2044 011344 004737 012714 JSR PC,TTOUT ;PRNT GOOD TAG
2045 011350 111105 MOVB (R1),R5
2046 011352 110503 MOVB R5,R3 ;GET GOOD CHAR
2047 011354 005737 000764 TST W2FLG ;SEE IF WRAP 2
2048 011360 001001 BNE WID2A ;IF SO: BR
2049 011362 105103 COMB R3 ;ELSE COMPLIMENT CHAR
2050 011364 004737 013270 WID2A: JSR PC,DOUT ;PRINT CHARACTER
2051 011370 012737 000240 000614 MOV #240,T0B
2052 011376 004737 013014 JSR PC,T0G ;SPACE
2053 011402 110503 MOVB R5,R3
2054 011404 004737 013270 JSR PC,DOUT ;PRINT CHAR
2055 011410 012704 016613 MOV #MSG20,R4
2056 011414 004737 012714 JSR PC,TTOUT ;PRINT BAD TAG
2057 011420 116203 000001 MOVB 1(R2),R3
2058 011424 004737 013270 JSR PC,DOUT ;PRINT BAD CHAR
2059 011430 012737 000240 000614 MOV #240,T0B
2060 011436 004737 013014 JSR PC,T0G ;SPACE
2061 011442 111203 MOVB (R2),R3
2062 011444 004737 013270 JSR PC,DOUT ;PRINT CHAR
2063 011450 005237 000766 INC DERFL ;SET DATA ERROR FLAG

```

```

2064 011454 122122
2065 011456 105722
2066 011460 005237 000774
2067 011464 000406
2068 011466 005737 000764
2069 011472 001401
2070 011474 000207
2071 011476 000137 010624
2072 011502 005200
2073 011504 001770
2074 011506 000137 011210
2075
2076
2077
2078 011512 000240
2079 011514 012700 000050
2080 011520 012701 017032
2081 011524 005721
2082 011526 012721 177400
2083 011532 005300
2084 011534 001374
2085 011536 012701 017154
2086 011542 012700 000050
2087 011546 012721 000377
2088 011552 012721 177400
2089 011556 005300
2090 011560 001374
2091 011562 000207
  
```

```

WID3:  CMPB  (R1)+,(R2)+  ;BUMP ADDRESS
        TSTB  (R2)+      ;BUMP ADDRESS
        INC   CRCNT      ;BUMP CHAR CNTR
        BR    WID5
WID4:  TST   W2FLG       ;SEE IF WRAP 2
        BEQ  WID4A      ;IF NOT: BR
        RTS  PC         ;ELSE RETURN
WID4A: JMP   PSCHK      ;GO CHECK POSTAMBLE
WID5:  INC   RO
        BEQ  WID4
        JMP  WID0

;PREAMBLE/POSTAMBLE GENERATE SUBROUTINE*****

PPGEN: NOP
        MOV  #50,RO      ;SET SIZE OF PREAMBLE
        MOV  #PRE,R1
        TST  (R1)+
        MOV  #177400,(R1)+ ;SET ADDRESS OF PRE
        DEC  RO          ;FILL TABLE
        BNE  1$         ;SEE IF DONE
        MOV  #POST,R1   ;IF NOT: BR
        MOV  #50,RO     ;SET ADDRESS OF POST
        MOV  #377,(R1)+ ;SET SIZE OF POST
        MOV  #177400,(R1)+ ;SET SYNC CHAR
        DEC  RO          ;FILL TABLE
        BNE  2$         ;SEE IF DONE
        RTS  PC         ;IF NOT: BR
        ;RETURN
  
```

```

2092
2093
2094
2095 011564 005237 000676 EORPA: INC TEMP2 ;SET WRAP FLAG
2096 011570 017700 166740 EORP: MOV @MR,RO ;GET MAINT REG
2097 011574 042700 000036 BIC #36,RO ;CLEAR CURRENT OP CODE
2098 011600 052700 000024 BIS #24,RO ;SET EOR CLEAR OP CODE
2099 011604 010077 166724 MOV RO,@MR ;DO EOR
2100 011610 042777 000037 166716 BIC #37,@MR ;CLEAR EOR AND MM
2101 011616 005000 CLR RO
2102 011620 012701 000002 MOV #2,R1
2103 011624 032777 000001 166656 EORP1: BIT #1,@C1 ;SEE IF GO GONE
2104 011632 001427 BEQ EORP2 ;IF SO: BR
2105 011634 005300 DEC RO
2106 011636 001372 BNE EORP1 ;AWAIT GO RESET
2107 011640 005301 DEC R1
2108 011642 001370 BNE EORP1
2109 011644 032777 020000 166716 BIT #20000,@SWR ;SEE IF ERROR PRINT INHIBIT
2110 011652 001017 BNE EORP2 ;IF SO: BR
2111 011654 005737 000620 TST HDRFL ;SEE IF DONE HEADER
2112 011660 001004 BNE EORP1A ;IF SO: BR
2113 011662 013704 000622 MOV EMADDR,R4
2114 011666 004737 012714 JSR PC,TTOUT ;PRINT HEADER
2115 011672 012704 016764 EORP1A: MOV @MSG31,R4
2116 011676 004737 012714 JSR PC,TTOUT ;PRINT EOR GO BIT ERROR
2117 011702 005777 166662 TST @SWR ;SEE IF HALT ON ERROR
2118 011706 100001 BPL EORP2 ;IF NOT: BR
2119 011710 000000 HALT
2120 011712 000240 EORP2: NOP
2121 011714 005737 000676 TST TEMP2 ;SEE IF WAM
2122 011720 001015 BNE EORPX ;IF NOT: BR
2123 011722 032777 000200 166640 BIT #200,@SWR ;SEE IF STATUS CHECK
2124 011730 001002 BNE EORP3 ;IF NOT: BR
2125 011732 004737 006704 JSR PC,WSTCK ;ELSE GO CHECK STATUS
2126 011736 000240 EORP3: NOP
2127 011740 032777 000400 166622 BIT #400,@SWR ;SEE IF DATA CHECK
2128 011746 001002 BNE EORPX ;IF NOT: BR
2129 011750 004737 007274 JSR PC,DCHK ;ELSE GO CHECK DATA
2130 011754 000240 EORPX: NOP
2131 011756 005037 000676 CLR TEMP2 ;CLEAR FLAG
2132 011762 000207 RTS PC ;RETURN
2133

```



```

2134
2135
2136
2137 011764 000240
2138 011766 032777 040000 166574
2139 011774 001001
2140 011776 000207
2141 012000 000240
2142 012002 005726
2143 012004 000240
2144 012006 000240
2145 012010 000177 166676
2146
2147
2148
2149 012014 032777 004000 166546
2150 012022 001403
2151 012024 005037 000702
2152 012030 000207
2153 012032 005737 001016
2154 012036 001772
2155 012040 005237 000702
2156 012044 023737 000702 000606
2157 012052 001764
2158 012054 005726
2159 012056 017700 166632
2160 012062 000110
2161

```

```

;SCOPE LOOP ON ERROR SUBROUTINE*****
SCOPE: NOP
BIT #40000,@SWR ;SEE IF LOOP ON ERROR
BNE 1$ ;IF SO: BR
RTS PC ;ELSE EXIT
1$: NOP
TST (SP)+ ;RESET STACK
NOP
NOP
JMP @SCOLP ;LOOP ON ERROR
;TEST ITERATION SUBROUTINE*****
ITER: BIT #4000,@SWR ;SEE IF ITERATIONS
BEQ 2$ ;IF SO: BR
1$: CLR ITCNT ;CLEAR ITERATION COUNTER
RTS PC ;ELSE EXIT
2$: TST @PCNTR ;DO SINGLE SUBTEST ITERATION
BEQ 1$ ;ON FIRST PASS
INC ITCNT ;BUMP COUNTER
CMP ITCNT,ITAMT ;SEE IF DONE ALL
BEQ 1$ ;IF SO: BR
TST (SP)+ ;RESET STACK
MOV @ITRLP,R0 ;SET ITERATION POINTER
JMP (R0) ;GO ITERATE

```

```

2162
2163
2164 ;INITIALIZE SUBROUTINE*****
2165 012064 012777 000040 166426 INIT: MOV #40,ACS ;INIT
2166 012072 013777 000624 166420 MOV DRVN,ACS ;SELECT DRIVE
2167 012100 013777 000664 166434 MOV SLVN,ATC ;SELECT SLAVE
2168 012106 013746 000776 MOV UDES,-(SP) ;GET TEST'S UNIT DESCRIPTION
2169 012112 042716 174377 BIC #174377,(SP) ;CLEAR ALL BUT DENSITY SELECT BITS
2170 012116 022726 001400 CMP #1400,(SP)+ ;BRANCH IF NOT NRZ (800 BPI)
2171 012122 001005 BNE 1$
2172 012124 032777 000040 166370 BIT #40,ADS ;BRANCH IF SLAVE IS IN NRZ MODE
2173 012132 001420 BEQ 4$ ;(PES = 0)
2174 012134 000404 BR 2$ ;GO CHANGE DENSITY
2175 012136 032777 000040 166356 1$: BIT #40,ADS ;BRANCH IF SLAVE IS IN PE MODE
2176 012144 001013 BNE 4$ ;(PES = 1)
2177 012146 012777 000007 166334 2$: MOV #7,AC1 ;REWIND SLAVE
2178 012154 032777 000200 166340 20$: BIT #200,ADS ;WAIT FOR READY
2179 012162 001774 BEQ 20$
2180 012164 032777 020000 166330 3$: BIT #20000,ADS ;LOOP UNTIL REWIND IS COMPLETE
2181 012172 001374 BNE 3$ ;(PIP = 0)
2182 012174 053777 000776 166340 4$: BIS UDES,ATC ;LOAD UNIT DESCRIPTION
2183 012202 032777 000002 166312 BIT #2,ADS ;BRANCH IF NOT AT BOT
2184 012210 001407 BEQ 6$
2185 012212 012777 000025 166270 MOV #25,AC1 ;ERASE TO GET OFF BOT
2186 012220 032777 000200 166274 5$: BIT #200,ADS ;LOOP UNTIL DONE
2187 012226 001774 BEQ 5$
2188 012230 012777 000011 166252 6$: MOV #11,AC1 ;DO A DRIVE CLEAR
2189 012236 000207 RTS PC ;RETURN TO CALLER
2190
2191 ;MAG TAPE INTERRUPT HANDLER*****
2192
2193 012240 000240 MTINT: NOP
2194 012242 013716 000670 MOV RTRN,(SP) ;SET RETURN TO (RTRN)
2195 012246 000002 RTI ;RETURN
2196
2197 ;TTY INTERRUPT HANDLER*****
2198
2199 012250 017746 166320 TTINT: MOV #ATKB,-(SP) ;GET CHARACTER
2200 012254 042716 000200 BIC #200,(SP) ;CLEAR PARITY BIT
2201 012260 122716 000003 CMPB #3,(SP) ;BRANCH IF NOT CONTROL C
2202 012264 001006 BNE 1$
2203 012266 005737 001262 TST CHNFLG ;INHIBIT ↑C IF CHAIN MODE
2204 012272 001003 BNE 1$
2205 012274 000005 RESET
2206 012276 000137 000200 JMP #200 ;RESTART PROGRAM
2207 012302 122716 000001 1$: CMPB #1,(SP) ;BRANCH IF NOT ↑A
2208 012306 001017 BNE 2$
2209 012310 022737 000176 000570 CMP #SWREG,SWR ;BRANCH IF HARDWARE SWR INVOKED
2210 012316 001016 BNE 3$
2211 012320 012737 177570 000570 MOV #177570,SWR ;INVOKE HARDWARE SWR
2212 012326 004737 013572 JSR PC,SAVE ;SAVE REGISTERS ON THE STACK
2213 012332 012704 014501 MOV #MSG63,R4 ;TYPE 'HARDWARE SWR IN USE'
2214 012336 004737 012714 JSR PC,TTOUT
2215 012342 004737 013614 JSR PC,RESTORE ;RESTORE REGISTERS
2216 012346 022716 000007 2$: CMP #7,(SP) ;BRANCH IF NOT ↑G
2217 012352 001005 BNE 4$

```



```

2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240 012372 010146
2241 012374 011601
2242 012376 005037 000674
2243 012402 005000
2244 012404 004737 012652
2245 012410 122737 000003 000616
2246 012416 001003
2247 012420 000005
2248 012422 000137 000200
2249 012426 122737 000015 000616
2250 012434 001004
2251 012436 005737 000674
2252 012442 001471
2253 012444 000457
2254 012446 122737 000025 000616
2255 012454 001005
2256 012456 012704 014427
2257 012462 004737 012714
2258 012466 000742
2259 012470 122737 000177 000616
2260 012476 001012
2261 012500 000241
2262 012502 006000
2263 012504 006200
2264 012506 006200
2265 012510 012704 014431
2266 012514 004737 012714
2267 012520 005201
2268 012522 000730
2269 012524 122737 000060 000616
2270 012532 101402
2271 012534 000137 012632
2272 012540 122737 000070 000616
2273 012546 101002
2274 012550 000137 012632
2275 012554 005237 000674
2276 012560 006300
2277 012562 006300
2278 012564 006300

;*****
;TTY ENTRY SUBROUTINE:
;THIS SUBROUTINE IS USED BY THE TEST CONDITION
;ENTRY ROUTINE TO READ THE RESPONSE ENTERED
;AT THE TTY AND CHECK THEM FOR LEGALITY AND
;LIMITS. ALL RESPONSE MUST BE TYPED IN OCTAL
;(0-7) AND MUST FALL WITHIN THE LIMITS SET BY
;THE CALLING ROUTINE.
;IF AN ENTRY IS ILLEGAL OR OUTSIDE THE LIMITS,
;A QUESTION MARK IS TYPED (?) AND THE RESPONSE
;MAY BE REENTERED.
;ENTRIES MAY NOT EXCEED SIX (6) CHARACTERS AND
;MAY BE TERMINATED AT LESS THAN SIX BY TYPING A
;CARRIAGE RETURN
;*****

TTR: MOV R1, -(SP) ;SAVE CHAR COUNT
10$: MOV (SP), R1 ;RESET CHAR COUNT (FOR ↑U)
CLR TEMP1 ;CLEAR FIRST CHARACTER FLAG
CLR RO
1$: JSR PC, TTIN ;GO READ CHARACTER
CMPB #3, TIB ;BRANCH IF NOT ↑C
BNE 11$
RESET
JMP #200 ;RESTART AT 200
11$: CMPB #15, TIB ;SEE IF CR
BNE 2$ ;IF NOT: BR
TST TEMP1 ;SEE IF FIRST CHARACTER
BEQ 9$ ;IF SO: BR
BR 6$ ;ELSE GO LOAD VALUE
2$: CMPB #25, TIB ;BRANCH IF NOT CONTROL U
BNE 21$
MOV #MSG59, R4 ;TYPE<CR><LF>
JSR PC, TTOUT
BR 10$
21$: CMPB #177, TIB ;BRANCH IF NOT 'RUBOUT'
BNE 3$
CLC ;REMOVE LAST TYPED CHAR
ROR RO
ASR RO
ASR RO
MOV #MSG60, R4 ;TYPE '\ '
JSR PC, TTOUT
INC R1 ;DECREMENT CHAR RECEIVED COUNT
BR 1$ ;GET NEXT CHAR
3$: CMPB #60, TIB ;SEE IF CHAR IS LESS THAN 0
BLOS 4$ ;IF NOT: BR
JMP TIB ;ELSE GO TO ERROR
4$: CMPB #70, TIB ;SEE IF CHAR IS GREATER THAN 7
BHI 5$ ;IF NOT: BR
JMP TIB ;ELSE GO TO ERROR
5$: INC TEMP1 ;SET FIRST CHARACTER FLAG
ASL RO ;SHIFT 3 LEFT
ASL RO
ASL RO

```

2279	012566	042737	177770	000616		BIC	#177770,T1B	:STRIP ASCII
2280	012574	053700	000616			BIS	T1B,R0	:LOAD CHARACTER
2281	012600	005301				DEC	R1	:SEE IF DONE
2282	012602	001300				BNE	1\$:IF NOT: BR
2283	012604	020002			6\$:	CMP	R0,R2	:SEE IF EXCEEDED MAXIMUM LIMIT
2284	012606	101402				BLOS	7\$:IF NOT: BR
2285	012610	000137	012632			JMP	T1NER	:ELSE GO TO ERROR
2286	012614	020300			7\$:	CMP	R3,R0	:SEE IF BELOW MINIMUM LIMIT
2287	012616	101402				BLOS	8\$:IF NOT: BR
2288	012620	000137	012632			JMP	T1NER	:ELSE GO TO ERROR
2289	012624	010015			8\$:	MOV	R0,(R5)	:LOAD VALUE
2290	012626	005726			9\$:	TST	(SP)+	:POP CHAR COUNT OFF STACK
2291	012630	000207				RTS	PC	:EXIT

```

2292
2293 ;TTY ENTRY ERROR SUBROUTINE*****
2294
2295 012632 012704 014304 T1NER: MOV #MSG40,B4
2296 012636 004737 012714 JSR PC,TTOUT ;PRINT?
2297 012642 005726 TST (SP)+ ;POP CHAR COUNT OFF STACK
2298 012644 162716 000020 SUB #20,(SP) ;RESET SP TO START OF VALUE ROUTINE
2299 012650 000207 RTS PC ;REDO VALUE ENTRY
2300
2301 ;TTY READ SUBROUTINE*****
2302
2303 012652 005277 165714 TTIN: INC #TKS
2304 012656 105777 165710 IS: TSTB #TKS
2305 012662 100375 BPL IS
2306 012664 017737 165704 000616 MOV #TKB,TIB
2307 012672 042737 000200 000616 BIC #200,TIB
2308 012700 013737 000616 000614 MOV TIB,TOB ;MOVE CHAR TO TTY OUPUT BFR
2309 012706 004737 013014 JSR PC,TOG ;ECHO CHARACTER
2310 012712 000207 RTS PC
2311
2312 ;TTY OUTPUT SUBROUTINE*****
2313
2314 012714 112437 000614 TTOUT: MOVB (R4)+,TOB
2315 012720 122737 000043 000614 CMPB #43,TOB
2316 012726 001440 BEQ TEX
2317 012730 122737 000045 000614 CMPB #45,TOB
2318 012736 001403 BEQ IS
2319 012740 004737 013014 JSR PC,TOG
2320 012744 000763 BR TTOUT
2321 012746 112737 000015 000614 IS: MOVB #15,TOB
2322 012754 004737 013014 JSR PC,TOG
2323 012760 012703 000004 MOV #4,R3
2324 012764 005037 000614 25: CLR TOB
2325 012770 004737 013014 JSR PC,TOG
2326 012774 005303 DEC R3
2327 012776 001372 BNE 25 ;DO FILLERS
2328 013000 112737 000012 000614 MOVB #12,TOB
2329 013006 004737 013014 JSR PC,TOG
2330 013012 000740 BR TTOUT
2331 013014 105777 165556 TOG: TSTB #TPS
2332 013020 100375 BPL TOG
2333 013022 113777 000614 165550 MOVB TOB,#TPB
2334 013030 000207 TEX: RTS PC
2335
2336 ;OCTAL OUTPUT SUBROUTINE*****
2337
2338
2339 013032 012737 000001 013266 OCTPE: MOV #1,OFL
2340 013040 000402 BR OCTPE1
2341 013042 005037 013266 OCTP: CLR OFL ;CLEAR FLAG FOR LEADING ZERO
2342 013046 010304 OCTPE1: MOV R3,R4 ;SEE IF NUMBER IS ZERO
2343 013050 001007 BNE OCTPO ;IF NOT ZERO: BR
2344 013052 005737 013266 TST OFL ;SEE IF PRINT ALL 0
2345 013056 001004 BNE OCTPO ;IF SO: BR
2346 013060 004737 013246 JSR PC,OCTPG1 ;ELSE PRINT ZERO
2347 013064 000137 013210 JMP OCTP3 ;SPACE AND EXIT

```



```

2395
2396 ;DATA CHARACTER OUTPUT SUBROUTINE*****
2397
2398 013270 012704 000010 DOUT: MOV #10,R4 ;SET NUMBER TO PRINT
2399 013274 110337 000614 MOVB R3,TOB
2400 013300 105777 165272 1$: TSTB @TPS
2401 013304 100375 BPL 1$
2402 013306 132737 000200 000614 BITB #200,TOB
2403 013314 001404 BEQ 2$
2404 013316 012777 000061 165254 MOV #061,@TPB
2405 013324 000403 BR 3$
2406 013326 012777 000060 165244 2$: MOV #060,@TPB
2407 013334 006337 000614 3$: ASL TOB
2408 013340 005304 DEC R4
2409 013342 001356 BNE 1$
2410 013344 000207 RTS PC
2411
2412 013346 013703 000700 DOUTD: MOV TEMP3,R3
2413 013352 000303 SWAB R3
2414 013354 004737 013270 JSR PC,DOUT
2415 013360 013703 000700 MOV TEMP3,R3
2416 013364 004737 013270 JSR PC,DOUT
2417 013370 000207 RTS PC
2418
2419 ;TE16 SERIAL NUMBER PRINT SUBROUTINE*****
2420
2421 013372 010304 SNPT: MOV R3,R4
2422 013374 000304 SWAB R4
2423 013376 006004 ROR R4
2424 013400 006004 ROR R4
2425 013402 006004 ROR R4
2426 013404 006004 ROR R4 ;GET FIRST DIGIT
2427 013406 004737 013450 JSR PC,SNPG ;PRINT
2428 013412 010304 MOV R3,R4
2429 013414 000304 SWAB R4 ;GET SECOND DIGIT
2430 013416 004737 013450 JSR PC,SNPG ;PRINT
2431 013422 010304 MOV R3,R4
2432 013424 006004 ROR R4
2433 013426 006004 ROR R4
2434 013430 006004 ROR R4
2435 013432 006004 ROR R4
2436 013434 004737 013450 JSR PC,SNPG ;PRINT THIRD DIGIT
2437 013440 010304 MOV R3,R4
2438 013442 004737 013450 JSR PC,SNPG ;PRINT FOURTH DIGIT
2439 013446 000207 RTS PC ;EXIT
2440 013450 012737 000260 000614 SNPG: MOV #260,TOB ;SET BASE = 0
2441 013456 042704 177760 BIC #177760,R4 ;MASK DIGIT
2442 013462 050437 000614 BIS R4,TOB ;SET ASCII
2443 013466 004737 013014 JSR PC,TOG ;TYPE DIGIT
2444 013472 000207 RTS PC ;RETURN

```



```

2475 ; THIS ROUTINE GETS THE NEW VALUE FOR THE SOFTWARE SWITCH REG
2476 013474 022737 000176 000570 GTSWR: CMP      #SWREG,SWR      ; BRANCH IF SOFTWARE SWR NOT
2477 013502 001032                      BNE      IS          ; INVOKED
2478 013504 004737 013572          JSR      PC,SAVE     ; SAVE REGISTERS ON THE STACK
2479 013510 012704 015757          MOV      #SWR,R4     ; TYPE 'SWR = '
2480 013514 004737 012714          JSR      PC,TTOUT    ;
2481 013520 017703 165044          MOV      #SWR,R3     ; GET CURRENT SETTING
2482 013524 004737 013032          JSR      PC,OCPE     ; AND TYPE THEM
2483 013530 012704 015767          MOV      #SWNEW,R4   ; TYPE 'NEW = '
2484 013534 004737 012714          JSR      PC,TTOUT    ;
2485 013540 013705 000570          MOV      SWR,R5      ; TTR ROUTINE RETURN NEW VALUE TO (R5)
2486 013544 012701 000007          MOV      #7,R1       ; LIMIT RESPONSE TO 7 CHARS
2487 013550 012702 177777          MOV      #177777,R2  ; BETWEEN 0 AND 177777
2488 013554 012703 000000          MOV      #0,R3       ;
2489 013560 004737 012372          JSR      PC,TTR      ; GET RESPONSE
2490 013564 004737 013614          JSR      PC,.RESTORE ; RESTORE REGISTERS
2491 013570 000207                      RTS          ; RETURN TO CALLER

2492 ;: ROUTINE TO SAVE REGISTERS ON THE STACK
2493 .SAVE: MOV      %5,-(SP)      ;: R5 IS SAVED AT 12(SP)
2494          MOV      %4,-(SP)      ;: R4 IS SAVED AT 10(SP)
2495          MOV      %3,-(SP)      ;: R3 IS SAVED AT 6(SP)
2496          MOV      %2,-(SP)      ;: R2 IS SAVED AT 4(SP)
2497          MOV      %1,-(SP)      ;: R1 IS SAVED AT 2(SP)
2498          MOV      %0,-(SP)      ;: R0 IS SAVED AT (SP)
2499          MOV      14(SP),-(SP)  ;: PUSH RETURN PC ON THE STACK
2500          RTS      PC          ;: RETURN TO CALLER

2501 ;: ROUTINE TO RESTORE REGISTERS SAVED ON THE STACK
2502 .RESTORE: MOV     (SP)+,14(SP)  ;: STORE RETURN PC ON STACK
2503          MOV     (SP)+,%0
2504          MOV     (SP)+,%1
2505          MOV     (SP)+,%2
2506          MOV     (SP)+,%3
2507          MOV     (SP)+,%4
2508          MOV     (SP)+,%5
2509          RTS      PC          ;: RETURN

```

013636
013644
013652
013660
013666
013674
013702
013710
013716
013717
013724
013732
013740
013746
013753
013760
013766
013774
014002
014010
014016
014024
014032
014035
014042
014050
014054
014062
014070
014072
014100
014106
014114
014122
014130
014136
014140
014146
014154
014162
014165
014172
014174
014202
014203
014210
014216
014220
014226
014234
014236
014244
014246

022445
052055
047503
020114
020103
050040
044511
052132
051
045
051523
040524
020123
052117
045
036040
047524
044515
051040
051516
041536
042522
022524
105
047516
053103
041522
052117
021524
047045
044530
040514
051045
047503
051525
021440
053445
041440
052502
020122
040
021440
051040
043
045
052111
021460
046445
051524
021467
044445
021440
052045

046524
030505
052116
047514
042524
051101
020040
041105
025052
051123
042520
052101
025052
052
054524
051103
052040
040516
051505
020105
052040
052123
043
050130
020124
021504
042126
042440
047117
052123
042526
040505
052116
050040
044522
047117
020123
043
054105
053103
051115
020123
020122
030440
042524
035122
020103

031460
020066
047522
044507
052123
020124
042050
040455
040452
020105
044440
041040
042520
020076
051105
042524
047520
020046
020117
051101
026524
042522
047055
050130
042455
051440
021440
020104
041040
051101
042524
020124
040520
052120
020104
041040
026464
044502
026465
035122
044502

;MESSAGE TABLE*****

- MSG1: .ASCII/%TM03-TE16 CONTROL LOGIC TEST PART II (DZTEB-A)/
- .ASCII /%***ASSURE TAPE IS AT BOT***/
- .ASCII /%TYPE <CR> TO TERMINATE RESPONSE & tC TO RESTART%/
- MSG6: .ASCII /EXPT-NOT RECVD#/
- MSG7: .ASCII /RCVD-NOT EXPT#/
- MSG9: .ASCII /%NON-EXIST SLAVE #/
- MSG10: .ASCII /%READ CONT BUS PAR #/
- MSG11: .ASCII /%WRITE CONT BUS PAR #/
- MSG12: .ASCII / EXPT #/
- MSG13: .ASCII / RCVD #/
- MSG14: .ASCII /%MR BITS 4-0#/
- MSG15: .ASCII /%MR BITS 15-7#/
- MSG16: .ASCII /%ITER: #/
- MSG18: .ASCII /%TC BITS 12-0 #/

2541	014254	051524	030440	026462	
2542	014262	020060	043		
2543	014265	045	041506	041040	MSG19: .ASCII /%FC BITS 15-0 #/
2544	014272	052111	020123	032461	
2545	014300	030055	021440		
2546	014304	037440	021440		MSG40: .ASCII / ? #/
2547	014310	022445	047105	020104	MSG41: .ASCII /%END OF PASS #/
2548	014316	043117	050040	051501	
2549	014324	020123	043		
2550	014327	045	042522	044507	MSG44: .ASCII /%REGISTER START: #/
2551	014334	052123	051105	051440	
2552	014342	040524	052122	020072	
2553	014350	043			
2554	014351	045	042526	052103	MSG45: .ASCII /%VECTOR ADDRESS: #/
2555	014356	051117	040440	042104	
2556	014364	042522	051523	020072	
2557	014372	043			
2558	014373	045	046524	031460	MSG57: .ASCII /%TMD3 DRIVE: #/
2559	014400	042040	044522	042526	
2560	014406	020072	043		
2561	014411	045	042524	033061	MSG58: .ASCII /%TE16 SLAVE: #/
2562	014416	051440	040514	042526	
2563	014424	020072	043		
2564	014427	045	043		MSG59: .ASCII /%#/
2565	014431	134	043		MSG60: .ASCII /\#/
2566	014433	045	042522	047515	MSG62: .ASCII /%REMOVE TMDP FROM SLAVE TO BE TESTED%#/
2567	014440	042526	052040	042115	
2568	014446	020120	051106	046517	
2569	014454	051440	040514	042526	
2570	014462	052040	020117	042502	
2571	014470	052040	051505	042524	
2572	014476	022504	043		
2573	014501	045	040510	042122	MSG63: .ASCII /%HARDWARE SWR IN USE%#/
2574	014506	040527	042522	051440	
2575	014514	051127	044440	020116	
2576	014522	051525	022505	043	

```

;TEST HEADER*****
2577
2578
2579 014527 045 046045 043517 MSLT1: .ASCII /%LOGIC TEST 1: WRAP 3,NRZ,NORMAL,ODD#/
2580 014534 041511 052040 051505
2581 014542 020124 035061 053440
2582 014550 040522 020120 026063
2583 014556 051116 026132 047516
2584 014564 046522 046101 047454
2585 014572 042104 043
2586 014575 045 046045 043517 MSLT2: .ASCII /%LOGIC TEST 2: WRAP 3,PE,NORMAL,ODD#/
2587 014602 041511 052040 051505
2588 014610 020124 035062 053440
2589 014616 040522 020120 026063
2590 014624 042520 047054 051117
2591 014632 040515 026114 042117
2592 014640 021504
2593 014642 022445 047514 044507 MSLT3: .ASCII /%LOGIC TEST 3: WRAP 2,NRZ,NORMAL,ODD#/
2594 014650 020103 042524 052123
2595 014656 031440 020072 051127
2596 014664 050101 031040 047054
2597 014672 055122 047054 051117
2598 014700 040515 026114 042117
2599 014706 021504
2600 014710 022445 047514 044507 MSLT4: .ASCII /%LOGIC TEST 4: WRAP 2,PE,NORMAL,ODD#/
2601 014716 020103 042524 052123
2602 014724 032040 020072 051127
2603 014732 050101 031040 050054
2604 014740 026105 047516 046522
2605 014746 046101 047454 042104
2606 014754 043
2607 014755 045 046045 043517 MSLT5: .ASCII /%LOGIC TEST 5: WRAP 1,NRZ,NORMAL,ODD#/
2608 014762 041511 052040 051505
2609 014770 020124 035065 053440
2610 014776 040522 020120 026061
2611 015004 051116 026132 047516
2612 015012 046522 046101 047454
2613 015020 042104 043
2614 015023 045 046045 043517 MSLT6: .ASCII /%LOGIC TEST 6: WRAP 1,PE,NORMAL,ODD#/
2615 015030 041511 052040 051505
2616 015036 020124 035066 053440
2617 015044 040522 020120 026061
2618 015052 042520 047054 051117
2619 015060 040515 026114 042117
2620 015066 021504
2621 015070 022445 047514 044507 MSLT7: .ASCII /%LOGIC TEST 7: WRAP 0,NRZ,NORMAL,ODD#/
2622 015076 020103 042524 052123
2623 015104 033440 020072 051127
2624 015112 050101 030040 047054
2625 015120 055122 047054 051117
2626 015126 040515 026114 042117
2627 015134 021504
2628 015136 022445 047514 044507 MSLT10: .ASCII /%LOGIC TEST 10: WRAP 0,PE,NORMAL,ODD#/
2629 015144 020103 042524 052123
2630 015152 030440 035060 053440
2631 015160 040522 020120 026060
2632 015166 042520 047054 051117

```

2633	015174	040515	026114	042117	
2634	015203	021504			
2635	015204	022445	047514	044507	MSLT11: .ASCII /%%LOGIC TEST 11: CORE DUMP WRITE (M8906)#/
2636	015212	020103	042524	052123	
2637	015220	030440	035061	041440	
2638	015226	051117	020105	052504	
2639	015234	050115	053440	044522	
2640	015242	042524	024040	034115	
2641	015250	030071	024466	043	
2642	015255	045	046045	043517	MSLT12: .ASCII /%%LOGIC TEST 12: CORE DUMP READ (M8906)#/
2643	015262	041511	052040	051505	
2644	015270	020124	031061	020072	
2645	015276	047503	042522	042040	
2646	015304	046525	020120	042522	
2647	015312	042101	024040	034115	
2648	015320	030071	024466	043	
2649	015325	045	046045	043517	MSLT13: .ASCII /%%LOGIC TEST 13: EVEN PARITY WRITE (M8903 M8904)#/
2650	015332	041511	052040	051505	
2651	015340	020124	031461	020072	
2652	015346	053105	047105	050040	
2653	015354	051101	052111	020131	
2654	015362	051127	052111	020105	
2655	015370	046450	034470	031460	
2656	015376	046440	034470	032060	
2657	015404	021451			
2658	015406	022445	047514	044507	MSLT14: .ASCII /%%LOGIC TEST 14: EVEN PARITY READ(M8903 M8904)#/
2659	015414	020103	042524	052123	
2660	015422	030440	035064	042440	
2661	015430	042526	020116	040520	
2662	015436	044522	054524	051040	
2663	015444	040505	024104	034115	
2664	015452	030071	020063	034115	
2665	015460	030071	024464	043	
2666	015465	045	046045	043517	MSLT15: .ASCII /%%LOGIC TEST 15: READ REVERSE(M8906)#/
2667	015472	041511	052040	051505	
2668	015500	020124	032461	020072	
2669	015506	042522	042101	051040	
2670	015514	053105	051105	042523	
2671	015522	046450	034470	033060	
2672	015530	021451			
2673	015532	022445	047514	044507	MSLT16: .ASCII /%%LOGIC TEST 16: CRC CORRECTION SINGLE TRACK, ALL FRAMES#/
2674	015540	020103	042524	052123	
2675	015546	030440	035066	041440	
2676	015554	041522	041440	051117	
2677	015562	042522	052103	047511	
2678	015570	020116	044523	043516	
2679	015576	042514	052040	040522	
2680	015604	045503	040454	046114	
2681	015612	043040	040522	042515	
2682	015620	021523			
2683	015622	022445	047514	044507	MSLT17: .ASCII /%%LOGIC TEST 17: CRC CORRECTION MULTIPLE BAD TRACKS#/
2684	015630	020103	042524	052123	
2685	015636	030440	035067	041440	
2686	015644	041522	041440	051117	
2687	015652	042522	052103	047511	
2688	015660	020116	052515	052114	

2689	015666	050111	042514	041040
2690	015674	042101	052040	040522
2691	015702	045503	021523	
2692	015706	022445	047514	044507
2693	015714	020103	042524	052123
2694	015722	031040	035060	051040
2695	015730	040505	020104	042522
2696	015736	042526	051522	026105
2697	015744	051116	026132	051127
2698	015752	050101	031440	043

MSLT20: .ASCII /%LOGIC TEST 20: READ REVERSE,NRZ,WRAP 3#/
/

```

2699
2700 ;TAG MESSAGE
2701
2702 015757 045 053523 020122 SMSWR: .ASCII /%SWR = #/
2703 015764 020075 043
2704 015767 040 042516 020127 SMNEW: .ASCII / NEW = #/
2705 015774 020075 043
2706
2707 016000 .EVEN
2708 ;WRITE BUFFER
2709
2710 016000 000100 WDATA:
2711 016000 177777 -1
2712 016002 177777 -1
2713 016004 177777 -1
2714 016006 177777 -1
2715 016010 177777 -1
2716 016012 177777 -1
2717 016014 177777 -1
2718 016016 177777 -1
2719 016020 177777 -1
2720 016022 177777 -1
2721 016024 177777 -1
2722 016026 177777 -1
2723 016030 177777 -1
2724 016032 177777 -1
2725 016034 177777 -1
2726 016036 177777 -1
2727 016040 177777 -1
2728 016042 177777 -1
2729 016044 177777 -1
2730 016046 177777 -1
2731 016050 177777 -1
2732 016052 177777 -1
2733 016054 177777 -1
2734 016056 177777 -1
2735 016060 177777 -1
2736 016062 177777 -1
2737 016064 177777 -1
2738 016066 177777 -1
2739 016070 177777 -1
2740 016072 177777 -1
2741 016074 177777 -1
2742 016076 177777 -1
2743 016100 177777 -1
2744 016102 177777 -1
2745 016104 177777 -1
2746 016106 177777 -1
2747 016110 177777 -1
2748 016112 177777 -1
2749 016114 177777 -1
2750 016116 177777 -1
2751 016120 177777 -1
2752 016122 177777 -1
2753 016124 177777 -1
2754 016126 177777 -1

```

2755	016130	177777	-1
2756	016132	177777	-1
2757	016134	177777	-1
2758	016136	177777	-1
2759	016140	177777	-1
2760	016142	177777	-1
2761	016144	177777	-1
2762	016146	177777	-1
2763	016150	177777	-1
2764	016152	177777	-1
2765	016154	177777	-1
2766	016156	177777	-1
2767	016160	177777	-1
2768	016162	177777	-1
2769	016164	177777	-1
2770	016166	177777	-1
2771	016170	177777	-1
2772	016172	177777	-1
2773	016174	177777	-1
2774	016176	177777	-1

2775			
2776			
2777			; READ BUFFER
2778			

2779	016200	000100	RDATA:
2780	016200	000000	0
2781	016202	000000	0
2782	016204	000000	0
2783	016206	000000	0
2784	016210	000000	0
2785	016212	000000	0
2786	016214	000000	0
2787	016216	000000	0
2788	016220	000000	0
2789	016222	000000	0
2790	016224	000000	0
2791	016226	000000	0
2792	016230	000000	0
2793	016232	000000	0
2794	016234	000000	0
2795	016236	000000	0
2796	016240	000000	0
2797	016242	000000	0
2798	016244	000000	0
2799	016246	000000	0
2800	016250	000000	0
2801	016252	000000	0
2802	016254	000000	0
2803	016256	000000	0
2804	016260	000000	0
2805	016262	000000	0
2806	016264	000000	0
2807	016266	000000	0
2808	016270	000000	0
2809	016272	000000	0
2810	016274	000000	0

2811	016276	000000	0
2812	016300	000000	0
2813	016302	000000	0
2814	016304	000000	0
2815	016306	000000	0
2816	016310	000000	0
2817	016312	000000	0
2818	016314	000000	0
2819	016316	000000	0
2820	016320	000000	0
2821	016322	000000	0
2822	016324	000000	0
2823	016326	000000	0
2824	016330	000000	0
2825	016332	000000	0
2826	016334	000000	0
2827	016336	000000	0
2828	016340	000000	0
2829	016342	000000	0
2830	016344	000000	0
2831	016346	000000	0
2832	016350	000000	0
2833	016352	000000	0
2834	016354	000000	0
2835	016356	000000	0
2836	016360	000000	0
2837	016362	000000	0
2838	016364	000000	0
2839	016366	000000	0
2840	016370	000000	0
2841	016372	000000	0
2842	016374	000000	0
2843	016376	000000	0

;WRAP AROUND MESSAGES*****

2847	016400	051445	052105	050125	WMSG2: .ASCII /%SETUP ERROR%#/ 042440 051122 051117
2848	016406	021445			
2849	016414	050045	052101	047122	WMSG3: .ASCII /%PATRN NUMBER = #/ 047040 046525 042502
2850	016416	020122	020075	043	
2851	016424	045	047516	026516	WMSG4: .ASCII /%NON-EXISTANT DRIVE%#/ 045 047516 026516
2852	016432	054105	051511	040524	
2853	016437	052116	042040	044522	
2854	016444	042526	021445		
2855	016452	041445	030523	021440	WMSG6: .ASCII /%CS1 #/ 041445 030523 021440
2856	016464	053445	020103	043	WMSG6A: .ASCII /%WC #/ 053445 020103 043
2857	016472	045	040502	021440	WMSG6B: .ASCII /%BA #/ 045 040502 021440
2858	016477	043045	020103	043	WMSG6C: .ASCII /%FC #/ 043045 020103 043
2859	016504	045	051503	020062	WMSG6D: .ASCII /%CS2 #/ 045 051503 020062
2860	016511	043			
2861	016516	045	051504	021440	WMSG6E: .ASCII /%DS #/ 045 051504 021440
2862	016517	042445	020122	043	WMSG6F: .ASCII /%ER #/ 042445 020122 043
2863	016524	045	051501	021440	WMSG6G: .ASCII /%AS #/ 045 051501 021440
2864	016531	041445	020103	043	WMSG6H: .ASCII /%CC #/ 041445 020103 043
2865	016536				

2867	016543	045	041104	021440	WMSG6I: .ASCII	/%DB #/
2868	016550	046445	020122	043	WMSG6J: .ASCII	/%MR #/
2869	016555	045	052104	021440	WMSG6K: .ASCII	/%DT #/
2870	016562	052045	020103	043	WMSG6L: .ASCII	/%TC #/
2871	016567	045	047123	021440	WMSG6M: .ASCII	/%SN #/
2872	016574	041045	042101	042040	WMSG16: .ASCII	/%BAD DATA#/
2873	016602	052101	021501			
2874	016606	043445	020072	043	WMSG17: .ASCII	/%G: #/
2875	016613	045	035102	021440	WMSG20: .ASCII	/%B: #/
2876	016620	041445	035116	021440	WMSG21: .ASCII	/%CN: #/
2877	016626	041045	042101	051440	WMSG23: .ASCII	/%BAD STATUS#/
2878	016634	040524	052524	021523		
2879	016642	047045	020117	047111	WMSG24: .ASCII	/%NO INTERRUPT#/
2880	016650	042524	051122	050125		
2881	016656	021524				
2882	016660	047045	020117	046103	WMSG25: .ASCII	/%NO CLOCK UP#/
2883	016666	041517	020113	050125		
2884	016674	043				
2885	016675	045	047516	041440	WMSG26: .ASCII	/%NO CLOCK DOWN#/
2886	016702	047514	045503	042040		
2887	016710	053517	021516			
2888	016714	042045	052101	020101	WMSG27: .ASCII	/%DATA PAT: #/
2889	016722	040520	035124	043		
2890	016727	045	040502	020104	WMSG28: .ASCII	/%BAD PREAMBLE#/
2891	016734	051120	040505	041115		
2892	016742	042514	043			
2893	016745	045	040502	020104	WMSG29: .ASCII	/%BAD POSTAMBLE#/
2894	016752	047520	052123	046501		
2895	016760	046102	021505			
2896	016764	042445	051117	041440	WMSG31: .ASCII	/%EOR CLEAR DID NOT CLEAR GO%#/
2897	016772	042514	051101	042040		
2898	017000	042111	047040	052117		
2899	017006	041440	042514	051101		
2900	017014	043440	022517	043		
2901	017021	040	040520	051124	WMSG32: .ASCII	/ PATRN #/
2902	017026	020116	043			
2903						
2904		017032				
2905	017032	000000			PRE:	0
2906	017034	000000				0
2907	017036	000000				0
2908	017040	000000				0
2909	017042	000000				0
2910	017044	000000				0
2911	017046	000000				0
2912	017050	000000				0
2913	017052	000000				0
2914	017054	000000				0
2915	017056	000000				0
2916	017060	000000				0
2917	017062	000000				0
2918	017064	000000				0
2919	017066	000000				0
2920	017070	000000				0
2921	017072	000000				0
2922	017074	000000				0

ADDFL	000746	DCKE2	010236	EXSC	006330	MSG10	014114	PATRN	001002
AS	000526	DCKE3	010246	EXSC0	006374	MSG11	014140	PCNTR	001016
ASF	000730	DCKE4	010270	EXSC1	006400	MSG12	014165	P00	010676
ATAF	000720	DCKE5	010302	EXSD	006404	MSG13	014174	PD1	010712
BA	000514	DCKE6	010340	FC	000516	MSG14	014203	PD2	010754
CC	000530	DCKE7	010376	FUN	000752	MSG15	014220	PD3	010760
CHNFLG	001262	DERFL	000766	GTSWR	013474	MSG16	014236	PD4	010770
CRCDAT	004304	DOUT	013270	HDRFL	000620	MSG18	014246	PD5	011110
CRCFLG	001022	DOUTD	013346	HERE	002320	MSG19	014265	PD6	011134
CRCNT	000774	DRVN	000624	ILFT	000544	MSG40	014304	PEXFL	000736
CRCPAT	004302	DRVTP	000604	INIT	012064	MSG41	014310	PFLG	000666
CS	000520	DS	000522	ITAMT	000606	MSG44	014327	POST	017154
C1	000510	DSUP	004742	ITCNT	000702	MSG45	014351	PPGEN	011512
DATAD	000760	DSO	005040	ITER	012014	MSG57	014373	PRCHK	010660
DATAD	001034	DS3	005056	ITRLP	000714	MSG58	014411	PRE	017032
DAT1	001036	DS4	005066	LTADD	000742	MSG59	014427	PREFL	000770
DAT2	001040	DT	000536	LT1	002352	MSG6	014035	PSCHK	010624
DAT3	001042	EMADDR	000622	LT1A	002416	MSG60	014431	PSW	000566
DATBL	001034	ENDFLG	001024	LT1B	002430	MSG62	014433	RBUFF	017710
DATC	000754	EORP	011570	LT10	003210	MSG63	014501	RCDP	001010
DAT1	005076	EORPA	011564	LT10A	003212	MSG7	014054	R0AD	000762
DAT1A	005102	EORPX	011754	LT11	003262	MSG9	014072	RDATA	016200
DAT2	005116	EORP1	011624	LT11A	003340	MSLT1	014527	RDRVF	001006
DAT3	005122	EORP1A	011672	LT11X	003364	MSLT10	015136	REGS	000612
DAT4	005130	EORP2	011712	LT12	003374	MSLT11	015204	RTRN	000670
DB	000532	EORP3	011736	LT12A	003460	MSLT12	015255	SAV1	000704
DCHK	007274	ER	000524	LT12X	003512	MSLT13	015325	SAV2	000706
DCHKA	007372	ERADD	000672	LT13	003522	MSLT14	015406	SAV3	000710
DCHKAD	007346	ERRF	000726	LT14	003572	MSLT15	015465	SCF	000732
DCHKA1	007422	EXEC	006010	LT15	003642	MSLT16	015532	SCOLP	000712
DCHKA2	007460	EXFL	000716	LT16	003724	MSLT17	015622	SCOPE	011764
DCHKB	007532	EXM2	006416	LT16A	003752	MSLT2	014575	SERFL	000772
DCHKBO	007542	EXM2A	006446	LT17	004312	MSLT20	015706	SERNUM	000602
DCHKC	007602	EXM2B	006502	LT17A	004342	MSLT3	014642	SETUP	005430
DCHKD	007610	EXM2C	006516	LT2	002464	MSLT4	014710	SET0	005472
DCKE	010156	EXM2E	006536	LT2A	002466	MSLT5	014755	SET1	005476
DCHKFL	001020	EXM2F	006544	LT20	004624	MSLT6	015023	SET1A	005502
DCHKX	010116	EXM2G	006574	LT20A	004702	MSLT7	015070	SET2	005526
DCHKX1	010134	EXM2H	006604	LT3	002536	MTINT	012240	SKAT	001014
DCHK0	007612	EXM2J	006620	LT3A	002602	NR20F	000662	SLAF	000722
DCHK1	007644	EXM2JO	006640	LT3B	002614	NXTDRV	001760	SLVN	000664
DCHK1A	007702	EXM2J1	006642	LT4	002644	NXTSLV	002032	SN	000540
DCHK2	007772	EXM2K	006662	LT4A	002646	OCTP	013042	SNPG	013450
DCHK2A	010036	EXM2X	006700	LT5	002716	OCTPE	013032	SNPT	013372
DCHK2B	010040	EX0	006032	LT5A	002762	OCTPE1	013046	SPO	005604
DCHK3	010066	EX1	006056	LT5B	002774	OCTPG	013224	SPO1	005572
DCKEP	010564	EX1A	006072	LT6	003024	OCTPG0	013242	SPI	005610
DCKEPO	010616	EX2	006112	LT6A	003026	OCTPG1	013246	SPIA	005662
DCKED	010216	EX3	006136	LT7	003102	OCTPO	013070	SPIB	005702
DCKE1	010226	EX5	006166	LT7A	003146	OCTP1	013112	SP3	005754
DCKE10	010510	EX5A	006172	LT7B	003160	OCTP2	013120	SP4	005774
DCKE11	010550	EX5A1	006226	MR	000534	OCTP3	013210	SP5	006006
DCKE12	010560	EX5B	006262	MSG1	013636	OFL	013266	SSCF	000724

START	001200	TR00	000626	VECT	000610	WMSG25	016660	WSTX	007272
STATC	001012	TR01	000630	WAM	000750	WMSG26	016675	WTAD	000756
STATF	001004	TR02	000632	WAM0	005136	WMSG27	016714	W1DCHK	011160
STFLG	000740	TR03	000634	WAM01	005144	WMSG28	016727	W1D0	011210
STFLGS	000614	TR04	000636	WAM1	005202	WMSG29	016745	W1D0A	011226
STSC0	002210	TR05	000640	WAM2	005212	WMSG3	016416	W1D1	011234
ST2	001670	TR06	000642	WAM2A	005246	WMSG31	016764	W1D2	011312
SWR	000570	TR07	000644	WAM3	005256	WMSG32	017021	W1D2A	011364
SWREG	000176	TR10	000646	WAM3A	005336	WMSG4	016437	W1D3	011454
TADX	001174	TR11	000650	WAM3B	005354	WMSG6	016464	W1D4	011466
TC	000542	TR12	000652	WAM4	005364	WMSG6A	016472	W1D4A	011476
TEMP1	000674	TR13	000654	WBUFF	017276	WMSG6B	016477	W1D5	011502
TEMP2	000676	TR14	000656	WC	000512	WMSG6C	016504	W2FLG	000764
TEMP3	000700	TR15	000660	WCDP0	001056	WMSG6D	016511	XORDAT	004306
TEND	002244	TSC0	001714	WCDP2	001044	WMSG6E	016517	XORPAT	004310
TENDX	002342	TSC0A	002110	WCS1	001024	WMSG6F	016524	SDONE	002252
TEX	013030	TSC00	002116	WCS2	001026	WMSG6G	016531	SENDAD	002310
TIB	000616	TSC01	002124	WDATA	016000	WMSG6H	016536	SHNEW	015767
TINER	012632	TSC02	002152	WDS	001030	WMSG6I	016543	SHSWR	015757
TKB	000574	TSC03	002170	WER	001032	WMSG6J	016550	SSVPC =	000764
TKS	000572	TSTTBL	001070	WMSG16	016574	WMSG6K	016555	. =	017712
TLAST	001176	TTIN	012652	WMSG17	016606	WMSG6L	016562	.RESTO	013614
T08	000614	TTINT	012250	WMSG2	016400	WMSG6M	016567	.SAVE	013572
TOG	013014	TTOUT	012714	WMSG20	016613	WPGFL	001000		
TPB	000600	TTR	012372	WMSG21	016620	WSTCK	006704		
TPS	000576	T24FL	000744	WMSG23	016626	WSTG	007150		
TREF	000734	UDES	000776	WMSG24	016642	WSTGO	007220		

. ABS. 017712 000

ERRORS DETECTED: 0
DEFAULT GLOBALS GENERATED: 0DZTEBA, DZTEBA/SOL+DZTEBA.P11
RUN-TIME: 3 5 .4 SECONDS
RUN-TIME RATIO: 33/9=3.6
CORE USED: 6K (11 PAGES)