

TM03 TU77 CLT2  
TE16 TU77 CZTEBCO

COPYRIGHT (c) 1977-84  
AH-A795C-MC  
FICHE 01 OF 01

JUL 1984  
digital  
Made In USA



.REM •

IDENTIFICATION

PRODUCT CODE: AC-A794C-MC  
PRODUCT NAME: CZTEBCO TM03-TE16/TU77 CONTROL LOGIC TEST PART II  
DATE CREATED: 15 MARCH 1984  
MAINTAINER: DIAGNOSTIC ENGINEERING  
AUTHOR: J. HITT

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, 1984 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	5
6.	ERROR PRINTOUTS	5
7.	OPERATION	7
8.	SUBTEST SUMMARIES	8
9.	LISTING	15

1. ABSTRACT  
-----

THIS PROGRAM IS DESIGNED TO SEQUENTIALLY TEST THE DATA FORMATTING FUNCTIONALITY OF THE TM03 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 TM03 THE STRUCTURE AND ITS MAINTAINENCE MODES.

2. REQUIREMENTS (HARDWARE)  
-----

- A. ANY PDP-11 PROCESSOR
- B. 8K OF CORE
- C. CONSOLE TTY
- D. TM03 MAGTAPE CONTROLLER
- E. MASSBUS CONTROLLER (RM)
- F. TE16 OR TU77 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 'C TO RESTART PROGRAM (@200)



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 000000  
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 TMO3 DRIVE #0, TE16/TU77 SLAVE #0.

\*\*NOTE: IN OREDR TO CHANGE THE DEFAULT SETTING OF THE SOFTWARE  
SWR, SET LOC: 176(SWREG:) TO THE DESIRED SETTING.

\*\*NOTE: WHEN RUN IN CHAIN MODE, THE DEFAULT SLAVE TYPE (TE16 OR TU77)  
MUST BE SET. SET LOC: 602(SLV TYP:) TE16=0; TU77=1.

4.2 SAMPLE START AT 200  
-----

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

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

TMO3-TE16/TU77 CONTROL LOGIC TEST PART II (DZTEB-B)  
\*\*\*ASSURE TAPE IS AT BOT\*\*\*  
TYPE ^C TO RESTART

REGISTER START: <172440> (CR)	[REGS:]
VECTOR ADDRESS: <224> (CR)	[VECT:]
TMO3 DRIVE: <0> (CR)	[DRVN:]
TE16 SLAVE: <0> (CR)	[SLVN:]
SLAVE TYPE (0=TE16,1=TU77): <0> (CR)	[SLV TYP:]
IF THE SOFTWARE SWR IS INVOKED:	
SWR = <000000> 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=HALT AT END OF PASS  
0=DO CONTINUOUS CYCLE
- 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 (?) 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: (SW0-SW5)

WHEN SW0-SW5 ARE SET TO ZERO (00), THE SCHEDULAR WILL EXECUTE ALL TESTS IN SEQUENCE. IF SW0-SW5 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 SW0-SW5.



8. SUBTEST SUMMARIES  
-----

\*\*\*NOTE: FOR TESTS 1-15\*\*

FOR THE MOST PART, THIS DIAGNOSTIC TESTS PARTICULAR AREAS OF THE TMO3 LOGIC INDEPENDENT OF THE TE16/TU77. HOWEVER THERE ARE A FEW SIGNALS WHICH ARE REQUIRED FROM THE TE16/TU77 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 WAMO 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 WRAP3 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-YB, MASSBUS P-LINES  
-----

CIRCUITS  
-----

PRINT REFERENCES  
-----

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

BF5  
BF2  
MR2, MR3  
MR5

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 WAM2 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-YB, M8933  
-----

CIRCUITS  
-----

PRINT REFERENCES  
-----

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

BF4  
BF2  
TCCM4  
MR2, MR3





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: M8934, M8933  
-----

CIRCUITS  
-----

PRINT REFERENCES  
-----

CRC GENERATOR	CNRZ2
CRC CHECKOUT	CNRZ3
CRC, CRC STROBE	TCCM4
READ LINE MULTIPLEXERS	TCCM6
MM CLK	MR5
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, M8932, M8933  
-----

CIRCUITS  
-----

PRINT REFERENCES  
-----

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

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: M8933  
-----

LOGIC TEST 14: EVEN PARITY READ: WAMO,  
-----

REPEAT OF TEST 7 EXCEPT EVEN PARITY IS USED.

LIKELY FAULT LOCATIONS: M8933, M8934  
-----



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.

LOGIC TEST 17: CRC ERROR CORRECTION  
-----

THIS TEST SIMULATES MULTIPLE FAILING TRACKS & TEST THAT  
NO ERROR CORRECTION IS PERFORMED. THE TEST SEQUENCE IS THE  
SAME AS TEST 16 STEP A--STEP E. THE DATA PATTERN USED IS  
125125.

LOGIC TEST 20: READ REVERSE WAM3 NRZ  
-----

REPEAT OF TEST 15 ABOVE (SEE ALSO TEST 2) EXCEPT THE TEST IS  
PERFORMED IN NRZ MODE.

545  
546

.LIST BIN,LOC,SEQ

547  
548  
549  
550  
551  
552  
553  
554  
555  
556  
557  
558  
559  
560  
561  
562  
563  
564  
565  
566  
567  
568  
569  
570  
571  
572  
573  
574  
575  
576  
577  
578  
579  
580  
581  
582  
583  
584

.TITLE CZTEBCO TM03-TE16/TU77 CLT II  
;CONTROL LOGIC TEST PART II  
;AC-A794C-MC  
;FEB 77  
;J.G. ADAMS  
;REVISED MAY 1978 BY J.G.ADAMS ;\*\*B CHANGED MODULE TYPEOUTS TO  
; ;\*\*B REFLECT TM03 REGISTER SET  
; ;\*\*B ADDED TU77 TEST CAPABILITY  
;REVISED MARCH 1984 BY J. HITT ;ADDED XON/XOFF FUNCTIONALITY

.MCALL .%ACT11,.\$EOP,\$CATCH,\$SAVE,\$RESTORE,\$CHAIN,\$CHNMODE  
.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: 0=HALT AT END OF PASS  
; 1=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> = 177777 OR NOT AVAILABLE USE SOFTWARE SWR





```

631                                     ;REGISTER EQUIVS*****
632
633          000000          R0=#0
634          000001          R1=#1
635          000002          R2=#2
636          000003          R3=#3
637          000004          R4=#4
638          000005          R5=#5
639          000006          SP=#6
640          000007          PC=#7
641
642
643
644                                     ;ACT11 HOOK *****
645          000764          $SVPC =.          ;SAVE CURRENT LOCATION CTR
646          000042          . =42
647          000042          .WORD 0
648          000046          . =46
649          000046          .WORD $ENDAD          ;SET LOCATION 46
650          000052          . =52
651          000052          .WORD 0          ;SET LOCATION 52 = 0
652          000764          . = $SVPC          ;RESTORE LOCATION CTR
653
654                                     ;TTY INTERRUPT VECTOR*****
655
656          000060          . =60
657          000060          .WORD TTINT          ;TTY INTERRUPT HANDLER ADDRESS
658          000062          .WORD 340          ;PRIORITY LEVEL 7
659
660                                     ;SOFTWARE SWITCH REGISTER*****
661                                     ;USED IF HARDWARE SWR = 177777, OR NOT AVAIL.
662          000176          . =176
663          000176          SWREG: .WORD 0
664
665
666                                     ;START ADDRESS*****
667          000200          . =200
668          000200          000137 001200          JMP START          ;PROGRAM START
669
670                                     ;RESTART ADDRESS*****
671          000210          . =210
672          000210          000137 001732          JMP ST2
673
674                                     ;TMO3 INTERRUPT VECTOR*****
675
676          000224          . =224
677          000224          012446          MTINT          ;TAPE INTERRUPT HANDLER ADDRESS
678          000226          000340
679

```

```

680
681          000510          .-510
682                                     ;MASS BUS REGISTER EQUIVS*****
683
684 000510 172440          C1: 172440
685 000512 172442          WC: 172442
686 000514 172444          BA: 172444
687 000516 172446          FC: 172446
688 000520 172450          CS: 172450
689 000522 172452          DS: 172452
690 000524 172454          ER: 172454
691 000526 172456          AS: 172456
692 000530 172460          CC: 172460
693 000532 172462          DB: 172462
694 000534 172464          MR: 172464
695 000536 172466          DT: 172466
696 000540 172470          SN: 172470
697 000542 172472          TC: 172472
698
699                                     ;ILLEGAL FUNCTION CODES
700
701 000544 005405          ILFT: 5405
702 000546 007415          7415
703 000550 016423          16423
704 000552 020437          20437
705 000554 022443          22443
706 000556 025447          25447
707 000560 031455          31455
708 000562 033465          33465
709 000564 036473          36473
710
711                                     ;CONSTANTS*****
712
713 000566 177776          PSW: 177776          ;PROCESSOR STATUS
714 000570 177570          SWR: 177570          ;SWITCH REGISTER
715 000572 177560          TKS: 177560          ;TTY READER STATUS
716 000574 177562          TKB: 177562          ;TTY READ BUFFER
717 000576 177564          TPS: 177564          ;TTY PUNCH STATUS
718 000600 177566          TPB: 177566          ;TTY PUNCH BUFFER
719 000602 000000          SLVTYP: .WORD 0          ;INDICATES SLAVE TYPE (0/1 = TE16/TU77)
720 000604 000020          ITAMT: 20          ;ITERATION AMOUNT
721 000606 000224          VECT: 224          ;INTERRUPT VECTOR(RH)
722 000610 172440          REGS: 172440          ;STARTING REGISTER ADDRESS

```

723  
724  
725  
726  
727  
728 000612  
729 000612 000000  
730 000614 000000  
731 000616 000000  
732 000620 000000  
733 000622 000000  
734 000624 000000  
735 000626 000000  
736 000630 000000  
737 000632 000000  
738 000634 000000  
739 000636 000000  
740 000640 000000  
741 000642 000000  
742 000644 000000  
743 000646 000000  
744 000650 000000  
745 000652 000000  
746 000654 000000  
747 000656 000000  
748 000660 000000  
749 000662 000000  
750 000664 000000  
751 000666 000000  
752 000670 000000  
753 000672 000000  
754 000674 000000  
755 000676 000000  
756 000700 000000  
757 000702 000000  
758 000704 000000  
759 000706 000000  
760 000710 000000  
761 000712 000000  
762 000714 000000  
763 000716 000000  
764 000720 000000  
765 000722 000000  
766 000724 000000  
767 000726 000000  
768 000730 000000  
769 000732 000000  
770 000734 000000  
771 000736 000000  
772 000740 000000  
773 000742 000000  
774 000744 000000  
775 000746 000000  
776 000750 000000  
777 000752 000000  
778 000754 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
HDRFL: 0
EMADDR: 0
DRVN: 0
TR00: 0
TR01: 0
TR02: 0
TR03: 0
TR04: 0
TR05: 0
TR06: 0
TR07: 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

```



779 000756 000000  
 780 000760 000000  
 781 000762 000000  
 782 000764 000000  
 783 000766 000000  
 784 000770 000000  
 785 000772 000000  
 786 000774 000000  
 787 000776 000000  
 788 001000 000000  
 789 001002 000000  
 790 001004 000000  
 791 001006 000000  
 792 001010 000000  
 793 001012 000000  
 794 001014 000000  
 795 001016 000000  
 796 001020 000000  
 797 001022  
 798  
 799  
 800  
 801 001022 000000  
 802 001024 000000  
 803 001026 000000  
 804 001030 000000  
 805  
 806  
 807  
 808 001032  
 809 001032 005140  
 810 001034 005160  
 811 001036 005164  
 812 001040 005172  
 813  
 814  
 815  
 816 001042 000005  
 817 001044 000005  
 818 001046 000012  
 819 001050 000012  
 820 001052 000000  
 821 001054 000017  
 822 001056 000017  
 823 001060 000017  
 824 001062 000017  
 825 001064 000000

DATAD: 0  
 RDAD: 0  
 W2FLG: 0  
 DERFL: 0  
 PREFL: 0  
 SERFL: 0  
 CRCNT: 0  
 UDES: 0  
 WPGFL: 0  
 PATRN: 0  
 STATF: 0  
 RDRVF: 0  
 RCDP: 0  
 STATC: 0  
 SKAT: 0  
 PCNTR: 0  
 DCHKFL: .WORD 0  
 CRCFLG: .WORD 0  
 ENDFLG:

;PASS COUNTER  
 ;DATA CHECK FLAG 0/1 = CHECK/DO NOT CHECK  
 ;CRC CORRECTION TEST IN PROGRESS

;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  
 WCDP0: 17  
 17  
 17  
 17  
 0

826  
827  
828  
829 001066 000000  
830 001070 000000  
831 001072 002414  
832 001074 002414  
833 001076 002526  
834 001100 002526  
835 001102 002600  
836 001104 002600  
837 001106 002706  
838 001110 002706  
839 001112 002760  
840 001114 002760  
841 001116 003066  
842 001120 003066  
843 001122 003144  
844 001124 003144  
845 001126 003252  
846 001130 003252  
847 001132 003324  
848 001134 003324  
849 001136 003436  
850 001140 003436  
851 001142 003564  
852 001144 003564  
853 001146 003634  
854 001150 003634  
855 001152 003704  
856 001154 003704  
857 001156 003766  
858 001160 003766  
859 001162 004354  
860 001164 004354  
861 001166 004666  
862 001170 004666  
863 001172 002306  
864 001174 000020  
865 001176 000000

;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  
\$CNTRLS: .WORD 0

;CONTAINS # OF TESTS  
;XON/XOFF FLAG

```

866 .EVEN
867 ;PROGRAM START AND HOUSEKEEPING*****
868
869 001200 012706 000500 START: MOV #500,SP ;SET STACK POINTER
870 001204 013746 000004 MOV @#4,-(SP) ;SAVE ERROR TRAP
871 001210 013746 000006 MOV @#6,-(SP)
872 001214 012737 001240 000004 MOV #1,@#4 ;SET TIME OUT TRAP TO GO TO 1$
873 001222 005037 000006 CLR @#6
874 001226 022777 177777 177334 CMP #177777,@SWR ;USE SOFTWARE SWITCH IF SWR = 177777
875 001234 001402 BEQ 2$ ;OR TIMES OUT
876 001236 000404 BR 3$ ;OTHERWISE USE HARDWARE SWR
877 001240 022626 1$: CMP (SP)+,(SP)+ ;RESET STACK
878 001242 012737 000176 000570 2$: MOV #SWREG,SWR ;SET SWR = TO ADDRESS OF SOFTWARE SWR
879 001250 012637 000006 3$: MOV (SP)+,@#6 ;RESTORE ERROR TRAP
880 001254 012637 000004 MOV (SP)+,@#4
881 001260 005027 CLR (PC)+ ;CLEAR CHAIN INDICATOR
882 001262 000000 CHNFLG: .WORD 0 ;CHAIN MODE INDICATOR
883 ;1/0 = CHAIN/NOT CHAIN MODE
884 001264 005737 000042 TST @#42 ;BRANCH IF IN DUMP MODE
885 001270 001407 BEQ 50$
886 001272 012737 000176 000570 MOV #SWREG,SWR ;INVOKE SOFTWARE SWR
887 001300 005237 001362 INC CHNFLG ;SET CHNFLG = CHAIN MODE
888 001304 000137 001310 JMP SCHN ;GO TO CHAIN ADDRESS
889 001310 50$:
890 001310 000240 SCHN: NOP
891 001312 122737 000006 000041 4$: CMPB #6,@#41 ;BRANCH IF NOT LOADED VIA TMDP
892 001320 001005 BNE 5$
893 001322 012704 014771 MOV #MSG62,R4 ;ADVISE USER TO REMOVE MEDIA FROM UUT
894 001326 004737 013154 JSR PC,TTOUT
895 001332 000000 HALT
896 001334 012704 014162 5$: MOV #MSG1,R4
897 001340 004737 013154 JSR PC,TTOUT ;PRINT TITLE
898 001344 005737 001262 TST CHNFLG ;SEE IF IN CHAIN MODE
899 001350 001402 BEQ 53$ ;IF NOT: BR
900 001352 000137 001756 JMP TSCD ;ELSE GO START TEST
901 001356 112737 000043 014162 53$: MOVB #'@,MSG1 ;DO NOT PRINT TITLE ON RESTART
902 001364 012704 014660 MOV #MSG44,R4
903 001370 004737 013154 JSR PC,TTOUT ;REQUEST REGISTER ADDRESS
904 001374 013703 000610 MOV REGS,R3
905 001400 004737 013366 JSR PC,OCTP ;PRINT CURRENT ADDRESS
906 001404 012705 000610 MOV #REGS,R5 ;SET ADDRESS SAVE LOC
907 001410 012701 000007 MOV #7,R1 ;SET SIZE OF RESPONSE
908 001414 012702 176400 MOV #176400,R2 ;SET UPPER LIMIT
909 001420 012703 172300 MOV #172300,R3 ;SET LOWER LIMIT
910 001424 004737 012632 JSR PC,ITR ;GO GET RESPONSE
911 001430 012704 014702 MOV #MSG45,R4
912 001434 004737 013154 JSR PC,TTOUT ;REQUEST VECTOR
913 001440 013703 000606 MOV VECT,R3
914 001444 004737 013366 JSR PC,OCTP ;PRINT CURRENT VECTOR
915 001450 012705 000606 MOV #VECT,R5 ;SET ADDRESS SAVE LOC
916 001454 012701 000004 MOV #4,R1 ;SET SIZE OF RESPONSE
917 001460 012702 000224 MOV #224,R2 ;SET UPPER LIMIT
918 001464 012703 000150 MOV #150,R3 ;SET LOWER LIMIT
919 001470 004737 012632 JSR PC,ITR ;GO GET RESPONSE
920 001474 013700 000606 MOV VECT,R0 ;GET VECTOR
921 001500 012720 012446 MOV #MTINT,(R0)+ ;LOAD INTERRUPT ADDRESS IN VECTOR

```



```

922 001504 012710 000340      MOV      #340,(R0)      ;LOAD PRIORITY
923 001510 013700 000610      MOV      REGS,R0      ;GET START OF REGS
924 001514 012701 000016      MOV      #16,R1      ;SET NUMBER OF REGS
925 001520 012702 000510      MOV      #C1,R2      ;GET START OF TABLE
926 001524 010022              6$:  MOV      R0,(R2)+     ;BUILD TABLE
927 001526 062700 000002      ADD      #2,R0        ;BUMP ADDRESS
928 001532 005301              DEC      R1           ;SEE IF DONE
929 001534 001373              BNE      6$          ;IF NOT: BR
930 001536 012702 000612      MOV      #STFLGS,R2   ;
931 001542 012700 000210      MOV      #ENDFLG-STFLGS,R0 ;GET # OF FLAGS TO CLEAR
932 001546 006200              ASR      R0           ;FORM COUNT
933 001550 005022              7$:  CLR      (R2)+       ;CLEAR FLAGS + COUNTERS
934 001552 005300              DEC      R0
935 001554 001375              BNE      7$
936 001556 012704 014724      MOV      #MSG57,R4    ;REQUEST TM03 DRIVE #
937 001562 004737 013154      JSR      PC,TTOUT
938 001566 013703 000622      MOV      DRVN,R3      ;GET CURRENT DRIVE
939 001572 004737 013366      JSR      PC,OCTP      ;AND TYPE IT
940 001576 012705 000622      MOV      #DRVN,R5     ;TTR ROUTINE RETURNS DRIVE TO (R5)
941 001602 012701 000002      MOV      #2,R1        ;LIMIT RESPONSE TO 1 CHARACTER
942 001606 012702 000007      MOV      #7,R2        ;BETWEEN 0 AND 7
943 001612 012703 000000      MOV      #0,R3
944 001616 004737 012632      JSR      PC,TTR       ;GET RESPONSE & PUT IN DRVN
945 001622 012704 014742      MOV      #MSG58,R4    ;REQUEST SLAVE #
946 001626 004737 013154      JSR      PC,TTOUT
947 001632 013703 000662      MOV      SLVN,R3      ;GET CURRENT SLAVE #
948 001636 004737 013366      JSR      PC,OCTP      ;AND TYPE IT
949 001642 012705 000662      MOV      #SLVN,R5     ;TTR ROUTINE RETURNS REPONSE TO (R5)
950 001646 012701 000002      MOV      #2,R1        ;LIMIT RESPONSE TO 1 CHARACTER
951 001652 012702 000007      MOV      #7,R2        ;BETWEEN 0-7
952 001656 012703 000000      MOV      #0,R3
953 001662 004737 012632      JSR      PC,TTR       ;GET RESPONSE & PUT IT IN SLVN
954
955 001666 012704 015065      MOV      #MSG64,R4    ;REQUEST SLAVE TYPE
956 001672 004737 013154      JSR      PC,TTOUT
957 001676 013703 000602      MOV      SLVTYP,R3    ;GET CURRENT SLAVE TYPE
958 001702 004737 013366      JSR      PC,OCTP      ;TYPE CURRENT VALUE
959 001706 012705 000602      MOV      #SLVTYP,R5   ;TTR ROUTINE STORES RESPONSE IN (R5)
960 001712 012701 000002      MOV      #2,R1        ;LIMIT RESPONSE TO 1 CHAR
961 001716 012702 000001      MOV      #1,R2        ;HI LIMIT
962 001722 012703 000000      MOV      #0,R3        ;LO LIMIT
963 001726 004737 012632      JSR      PC,TTR       ;GET RESPONSE & STORE IN SLVTYP
964
965
966 001732 012706 000500      ;START 210
967 001736 005037 001004      ST2:  MOV      #500,SP ;SET STACK PTR
968 001742 005037 001014      CLR      RDRVF        ;CLEAR READ REVERSE FLAG
969 001746 005037 001020      CLR      PCNTR        ;
970 001752 004737 014020      CLR      CRCFLG       ;SET CRC FLAG = CRC NOT IN PROGRESS
                          JSR      PC,GTSWR      ;GET SOFTWARE SWITCHES

```

```
971  
972 ;TEST SCHEDULAR*****  
973  
974 001756 052777 000100 176606 TSCD: BIS #100,@TKS ;SET KEYBOARD IE BIT  
975 001764 005037 000776 CLR WPGFL ;CLEAR WRAP PATRN FLAG  
976 001770 005037 000736 CLR STFLG ;CLEAR SINGLE TEST FLAG  
977 001774 017700 176570 MOV @SWR,RO  
978 002000 042700 177700 BIC #177700,RO ;BRANCH IF SINGLE TEST SELECTED  
979 002004 001122 BNE STSCD ;GO SELECT SINGLE TEST  
980 002006 005737 001262 TST CHNFLG ;BRANCH IF NOT IN CHAIN MODE  
981 002012 001457 BEQ TSCDA  
982 002014 012737 177777 000622 MOV #-1,DRVN ;; INITIALIZE DRIVE #  
983 002022 012737 177777 000662 NXTDRV: MOV #-1,SLVN ;; INITIALIZE SLAVE #  
984 002030 012777 000040 176462 1$: MOV #40,@CS ;; INIT CONTROLLER  
985 002036 005237 000622 INC DRVN ;; STEP DRIVE #  
986 002042 022737 000010 000622 CMP #10,DRVN ;; EXIT IF ALL DRIVES TESTED  
987 002050 001521 BEQ $DONE ;; FOR AVAILABILITY  
988 002052 013777 000622 176440 MOV DRVN,@CS ;; LOAD DRIVE #  
989 002060 005777 176424 TST @C1 ;; ACCESS DRIVE  
990 002064 032777 010000 176426 BIT #10000,@CS ;; BRANCH IF DRIVE NON EXISTANT  
991 002072 001356 BNE 1$ ;; (NED = 1)  
992 002074 005237 000662 NXTSLV: INC SLVN ;; STEP SLAVE # AND BRANCH  
993 002100 001011 BNE 1$ ;; IF NOT SLAVE 0  
994 002102 005737 000622 TST DRVN ;; BRANCH IF NOT DRIVE # 0  
995 002106 001006 BNE 1$  
996 002110 122737 000006 000041 CMPB #6,@#41 ;; BRANCH IF NOT TMDP  
997 002116 001002 BNE 1$  
998 002120 005237 000662 INC SLVN ;; STEP TO SLAVE # 1  
999 002124 022737 000010 000662 1$: CMP #10,SLVN ;; BRANCH IF ALL SLAVES TESTED  
1000 002132 001733 BEQ NXTDRV ;; FOR AVAILABILITY  
1001 002134 013777 000662 176400 MOV SLVN,@C ;LOAD SLAVE UNIT #  
1002 002142 032777 002000 176366 BIT #2000,@DT ;BRANCH IF SLAVE NOT  
1003 002150 001751 BEQ NXTSLV ;PRESENT (SPR = 0)  
1004 002152 012737 001066 000740 TSCDA: MOV #TSTTBL,LTADD  
1005 002160 062737 000004 000740 TSCD0: ADD #4,LTADD  
1006 002166 013737 000740 000712 TSCD1: MOV LTADD,ITRLP  
1007 002174 062737 000002 000712 ADD #2,ITRLP ;SET ITERATION ADDRESS  
1008 002202 005037 000616 CLR HDRFL ;CLEAR PRINT HEADER FLAG  
1009 002206 017700 176526 MOV @LTADD,RO ;SET POINTER TO TEST  
1010 002212 000110 JMP (RO) ;GO TO TEST  
1011 002214 032777 002000 176346 TSCD2: BIT #2000,@SWR ;SEE IF HALT ON TEST  
1012 002222 001403 BEQ TSCD3 ;IF NOT: BR  
1013 002224 000000 HALT  
1014 002226 005037 000776 CLR WPGFL ;CLEAR WRAP DATA GENERATOR FLAG  
1015 002232 005737 000736 TSCD3: TST STFLG ;SE IF SINGLE TEST  
1016 002236 001750 BEQ TSCD0 ;IF NOT: BR  
1017 002240 017700 176324 MOV @SWR,RO  
1018 002244 042700 177700 BIC #177700,RO ;MASK TEST NUMBER  
1019 002250 001642 BEQ TSCD ;IF SO: BR  
1020 002252 012737 000001 000736 STSCD: MOV #1,STFLG ;SET SINGLE TEST FLAG  
1021 002260 023700 001174 CMP TLAS1,RO ;SEE IF EXCEEDED TESTS  
1022 002264 002410 BLT TEND ;IF SO: BR  
1023 002266 006300 ASL RO  
1024 002270 006100 ROL RO ;SET TABLE MODIFIER  
1025 002272 012737 001066 000740 MOV #1STTBL,LTADD  
1026 002300 060037 000740 ADD PC,LTADD ;SET TEST POINTER
```



```

1027 002304 000730
1028 002306 005737 001262
1029 002312 001270
1030 002314 012704 014641
1031 002320 004737 013154
1032 002324 013703 001014
1033 002330 004737 013366
1034 002334 005000
1035 002336 005300
1036 002340 001376
1037 002342 013700 000042
1038 002346 001405
1039 002350 000005
1040 002352 004710
1041 002354 000240
1042 002356 000240
1043 002360 000240
1044 002362 000240
1045 002364 005737 001262
1046 002370 001005
1047 002372 032777 010000 176170
1048 002400 001401
1049 002402 000000
1050 002404 005237 001014
1051 002410 000137 001756
1052

TEND: BR TSCD1
TST CHNFLG ;BRANCH IF IN CHAIN MODE
BNE NXTSLV
$DONE: MOV #MSG41,R4
JSR PC,TTOUT ;PRINT END OF PASS
MOV PCNTR,R3
JSR PC,OCTP ;PRINT PASS NUMBER
CLR R0
1$: DEC R0 ;DELAY WAITING FOR
BNE 1$ ;TTY TO FINISH
MOV @#42,R0 ;GET ACT11 RETURN ADDRESS
BEQ HERE ;BRANCH IF NOT ACT11
RESET
$ENDAD: JSR PC,(R0)
NOP
NOP
NOP
HERE: NOP
TST CHNFLG ;BRANCH IF IN CHAIN MODE
BNE TENDX
BIT #10000,@SWR ;SEE IF HALT ON PASS
BEQ TENDX ;IF NOT: BR
HALT
TENDX: INC PCNTR ;BUMP PASS COUNTER
JMP TSCD ;RESTART

```





```

1101
1102
1103
1104 002706 000240
1105 002710 012737 004260 001022
1106 002716 012737 000100 001024
1107 002724 012737 010640 001026
1108 002732 012737 000000 001030
1109 002740 012737 015357 000620
1110 002746 012737 002300 000774
1111 002754 000137 002644
1112
1113
1114
1115 002760 012737 004260 001022
1116 002766 012737 000100 001024
1117 002774 012737 010600 001026
1118 003002 012737 000000 001030
1119 003010 012737 015424 000620
1120 003016 012737 001700 000774
1121 003024 005037 001000
1122 003030 012737 003036 000710
1123 003036 004737 005244
1124 003042 005237 001000
1125 003046 032737 000004 001000
1126 003054 001770
1127 003056 004737 012150
1128 003062 000137 002214
1129
1130
1131
1132 003066 000240
1133 003070 004737 011630
1134 003074 012737 004260 001022
1135 003102 012737 000100 001024
1136 003110 012737 010640 001026
1137 003116 012737 000000 001030
1138 003124 012737 015472 000620
1139 003132 012737 002300 000774
1140 003140 000137 003024

;LOGIC TEST 4: WRAP 2, PE, NORMAL, ODD*****
LT4: NOP
LT4A: MOV #4260,WCS1 ;SET EXPT CS1
MOV #100,WCS2 ;SET EXPT CS2
MOV #10640,WDS ;SET EXPT DS
MOV #0,WER ;SET EXPT WER
MOV #MSLT4,EMADDR ;SET HEADER
MOV #2300,UDES ;SET PE, NORMAL, ODD
JMP LT3A ;GO EXECUTE TEST SEQUENCES

;LOGIC TEST 5: WRAP 1, NRZ, NORMAL, ODD*****
LT5: MOV #4260,WCS1 ;SET EXPT CS1
MOV #100,WCS2 ;SET EXPT CS2
MOV #10600,WDS ;SET EXPT DS
MOV #0,WER ;SET EXPT WER
MOV #MSLT5,EMADDR ;SET HEADER
MOV #1700,UDES ;SET NRZ, NORMAL, ODD
LT5A: CLR PATRN ;POINT TO PATTERN ZERO
MOV #LT5B,SCOLP ;SET SCOPE ADDRESS
LT5B: JSR PC,WAM1 ;GO DO WRAP 1
INC PATRN ;BUMP POINTER
BIT #4,PATRN ;SEE IF DONE
BEQ LT5B ;IF NOT: BR
JSR PC,ITER ;GO SEE IF ITERATIONS
JMP TSCD2 ;RETURN TO SCHEDULAR

;LOGIC TEST 6: WRAP 1, PE, NORMAL, ODD*****
LT6: NOP
LT6A: JSR PC,PPGEN ;GO GENERATE PRE/POSTAMBLE
MOV #4260,WCS1 ;SET EXPT CS1
MOV #100,WCS2 ;SET EXPT CS2
MOV #10640,WDS ;SET EXPT DS
MOV #0,WER ;SET EXPT WER
MOV #MSLT6,EMADDR ;SET HEADER
MOV #2300,UDES ;SET PE, NORMAL, ODD
JMP LT5A ;GO EXECUTE TEST SEQUENCE

```

```

1141
1142
1143
1144 003144 012737 144260 001022 LT7:  MOV    #144260,WCS1    ;SET EXPT CS1
1145 003152 012737 000100 001024      MOV    #100,WCS2      ;SET EXPT CS2
1146 003160 012737 150600 001026      MOV    #150600,WDS    ;SET EXPT DS
1147 003166 012737 000200 001030      MOV    #200,WER       ;SET EXPT ER
1148 003174 012737 015537 000620      MOV    #MSLT7,EMADDR  ;SET HEADER
1149 003202 012737 001700 000774      MOV    #1700,UDES     ;SET NRZ, NORMAL, ODD
1150 003210 005037 001000      LT7A: CLR    PATRN      ;POINT TO PATTERN 0
1151 003214 012737 003222 000710      MOV    #LT7B,SCOLP    ;SET SCOPE ADDRESS
1152 003222 004737 005200      LT7B: JSR    PC,WAMO    ;GO DO WRAP 0
1153 003226 005237 001000      INC    PATRN          ;BUMP POINTER
1154 003232 032737 000004 001000      BIT    #4,PATRN       ;SEE IF DONE
1155 003240 001770      BEQ    LT7B          ;IF NOT: BR
1156 003242 004737 012150      JSR    PC,ITER        ;GO SEE IF ITERATIONS
1157 003246 000137 002214      JMP    TSCD2          ;RETURN TO SCHEDULAR
1158
1159
1160
1161 003252 000240      LT10: NOP
1162 003254 012737 004260 001022      LT10A: MOV    #4260,WCS1    ;SET EXPT CS1
1163 003262 012737 000100 001024      MOV    #100,WCS2     ;SET EXPT CS2
1164 003270 012737 010640 001026      MOV    #10640,WDS    ;SET EXPT DS
1165 003276 012737 000000 001030      MOV    #0,WER        ;SET EXPT ER
1166 003304 012737 015605 000620      MOV    #MSLT10,EMADDR ;SET HEADER
1167 003312 012737 002300 000774      MOV    #2300,UDES    ;SET PE, NORMAL, ODD
1168 003320 000137 003210      JMP    LT7A          ;GO EXECUTE TEST SEQUENCE
    
```



```

1169
1170
1171 003324 012737 004260 001022 LT11: MOV #4260,WCS1 ;SET EXPT CS1
1172 003332 012737 000100 001024 MOV #100,WCS2 ;SET EXPT CS2
1173 003340 012737 010600 001026 MOV #10600,WDS ;SET EXPT DS
1174 003346 012737 000000 001030 MOV #0,WER ;SET EXPT ER
1175 003354 012737 015653 000620 MOV #MSLT11,EMADDR ;SET HEADER
1176 003362 012737 001720 000774 MOV #1720,UDES ;SET NRZ, CORE DUMP, ODD
1177 003370 005037 001000 CLR PATRN ;POINT TO PATTERN 0
1178 003374 012737 003402 000710 MOV #LT11A,SCOLP ;SET SCOPE ADDRESS
1179 003402 004737 005254 001000 LT11A: JSR PC,WAM2 ;GO DO WAM 2
1180 003406 022737 000002 001000 CMP #2,PATRN ;SEE IF DONE
1181 003414 001404 BEQ LT11X ;IF SO: BR
1182 003416 012737 000002 001000 MOV #2,PATRN ;SELECT PATTERN 2
1183 003424 000766 BR LT11A ;CONTINUE
1184 003426 004737 012150 LT11X: JSR PC,ITER ;GO SEE IF ITERATIONS
1185 003432 000137 002214 JMP TSCD2 ;RETURN TO SCHEDULES
1186
1187 ;LOGIC TEST 12: CORE DUMP READ, WAM 3*****
1188
1189 003436 012737 004270 001022 LT12: MOV #4270,WCS1 ;SET EXPT CS1
1190 003444 012737 000100 001024 MOV #100,WCS2 ;SET EXPT CS2
1191 003452 012737 010600 001026 MOV #10600,WDS ;SET EXPT DS
1192 003460 012737 000000 001030 MOV #0,WER ;SET EXPT ER
1193 003466 012737 015724 000620 MOV #MSLT12,EMADDR ;SET HEADER
1194 003474 012737 001720 000774 MOV #1720,UDES ;SELECT NRZ, CORE DUMP, ODD
1195 003502 005037 001000 CLR PATRN ;SELECT PATTERN 0
1196 003506 012737 003522 000710 MOV #LT12A,SCOLP ;SET SCOPE ADDRESS
1197 003514 012737 001054 001006 MOV #WCDP0,RCDP ;POINT TO PATTERN 0
1198 003522 004737 005320 LT12A: JSR PC,WAM3 ;GO DO WAM3
1199 003526 022737 000002 001000 CMP #2,PATRN ;SEE IF DONE
1200 003534 001407 BEQ LT12X ;IF SO: BR
1201 003536 012737 000002 001000 MOV #2,PATRN ;SELECT PATTERN 2
1202 003544 012737 001042 001006 MOV #WCDP2,RCDP ;POINT TO PATTERN 2
1203 003552 000763 BR LT12A ;CONTINUE
1204 003554 004737 012150 LT12X: JSR PC,ITER ;GO SEE IF ITERATION
1205 003560 000137 002214 JMP TSCD2 ;RETURN TO SCHEDULE

```

```

1206
1207
1208
1209 003564 012737 004260 001022 LT13: MOV #4260,WCS1 ;SET EXPT CS1
1210 003572 012737 000100 001024 MOV #100,WCS2 ;SET EXPT CS2
1211 003600 012737 010600 001026 MOV #10600,WDS ;SET EXPT DS
1212 003606 012737 000000 001030 MOV #0,WER ;SET EXPT ER
1213 003614 012737 015774 000620 MOV #MSLT13,EMADDR ;SET HEADER
1214 003622 012737 001710 000774 MOV #1710,UDES ;SET NRZ, NORMAL, EVEN
1215 003630 000137 003024 JMP LT5A ;GO EXECUTE WAM 1
1216
1217
1218
1219 003634 012737 144260 001022 LT14: MOV #144260,WCS1 ;SET EXPT CS1
1220 003642 012737 000100 001024 MOV #100,WCS2 ;SET EXPT CS2
1221 003650 012737 150600 001026 MOV #150600,WDS ;SET EXPT DS
1222 003656 012737 000200 001030 MOV #200,WER ;SET EXPT ER
1223 003664 012737 016055 000620 MOV #MSLT14,EMADDR ;SET HEADER
1224 003672 012737 001710 000774 MOV #1710,UDES ;SET NRZ, NORMAL, EVEN
1225 003700 000137 003210 JMP LT7A ;GO DO WAM 0
1226
1227
1228
1229 003704 012737 004276 001022 LT15: MOV #4276,WCS1 ;SET EXPT CS1
1230 003712 012737 000100 001024 MOV #100,WCS2 ;SET EXPT CS2
1231 003720 012737 010640 001026 MOV #10640,WDS ;SET EXPT DS
1232 003726 012737 000000 001030 MOV #0,WER ;SET EXPT ER
1233 003734 012737 016134 000620 MOV #MSLT15,EMADDR ;SET HEADER
1234 003742 012737 002300 000774 MOV #2300,UDES ;SELECT PE,NORMAL,ODD
1235 003750 000240 NOP
1236 003752 000240 NOP
1237 003754 012737 000001 001004 MOV #1,RDRVF ;SET READ REVERSE FLAG
1238 003762 000137 002460 JMP LT1A ;GO DO WAM 3, REVERSE
1239

```



1240  
 1241  
 1242  
 1243  
 1244  
 1245  
 1246  
 1247  
 1248  
 1249  
 1250  
 1251  
 1252  
 1253  
 1254  
 1255  
 1256  
 1257  
 1258  
 1259  
 1260  
 1261  
 1262  
 1263  
 1264  
 1265  
 1266  
 1267  
 1268  
 1269  
 1270  
 1271  
 1272  
 1273  
 1274  
 1275  
 1276  
 1277  
 1278  
 1279  
 1280  
 1281  
 1282  
 1283  
 1284  
 1285  
 1286  
 1287  
 1288  
 1289  
 1290  
 1291  
 1292  
 1293  
 1294  
 1295

003766	005037	001000	
003772	012737	000401	004344
004000	012737	016201	000620
004006	012737	004014	000710
004014	012737	144260	001022
004022	012737	000100	001024
004030	012737	150600	001026
004036	012737	000200	001030
004044	012737	001700	000774
004052	005037	001020	
004056	004737	005200	
004062	012737	000001	001020
004070	013737	017744	004346
004076	013737	004346	004350
004104	013737	004344	004352
004112	043737	004344	004350
004120	043737	004346	004352
004126	053737	004352	004350
004134	013737	004350	004346
004142	012737	144260	001022
004150	012737	000110	001024
004156	012737	150600	001026
004164	012737	100300	001030
004172	004737	005426	
004176	000240		
004200	012737	000001	001004
004206	012737	144276	001022
004214	012737	000110	001024
004222	012737	150600	001026
004230	012737	001000	001030
004236	004737	005320	
004242	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 #MSLT16,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,ODD
                    ;PARITY & PDP11 NORMAL MODE
       CLR CRCFLG ;CLEAR CRC CORRECTION FLAG
       JSR PC,WAMO ;DO A WRAP 0 --- STEP A
       MOV #1,CRCFLG ;SET CRC ERROR CORRECTION IN PROGRESS
       MOV WBUFF,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
  
```



1296	004244	012737	004260	001022	MOV	#4260,WCS1	
1297	004252	012737	000110	001024	MOV	#110,WCS2	
1298	004260	012737	010600	001026	MOV	#10600,WDS	
1299	004266	012737	000000	001030	MOV	#0,WER	
1300	004274	005037	001020		CLR	CRCFLG	;CLEAR CRC CORRECTION IN PROGRESS FLAG
1301	004300	004737	005426		JSR	PC,WAM4	;GO TO WRAP 4 --- STEP D
1302							
1303	004304	006337	004344		ASL	CRCPAT	;SELECT NEXT TRACK TO BE ALTERED
1304	004310	103241			BCC	LT16A	;CONTINUE FOR ALL TRACKS
1305	004312	012737	000401	004344	MOV	#401,CRCPAT	;RESET BITS TO TRACK 1
1306	004320	005237	001000		INC	PATRN	;SELECT NEXT PATTERN
1307	004324	022737	000003	001000	CMP	#3,PATRN	;BRANCH IF NOT DONE
1308	004332	001230			BNE	LT16A	
1309	004334	004737	012150		JSR	PC,ITER	;ITERATION LOOP
1310	004340	000137	002214		JMP	TSCD2	;RETURN TO SCHEDULER
1311							
1312							
1313	004344	000000			CRCPAT: .WORD	0	;CONTAINS BITS TO BE ALTERED
1314	004346	000000			CRCDAT: .WORD	0	;CONTAINS DATA TO BE WRITTEN BY WRAP4
1315	004350	000000			XORDAT: .WORD	0	;TEMPRARY STORAGE FOR XOR
1316	004352	000000			XORPAT: .WORD	0	;TEMPOARY STORAGE FOR XOR
1317							

```

1318
1319
1320
1321
1322
1323
1324 004354 012737 000002 001000 LT17: MOV #2,PATRN ;SELECT PATTERN #2 (125125)
1325 004362 012737 001001 004344 MOV #1001,CRCPAT ;SELECT 2 BAD TRACKS
1326 004370 012737 016271 000620 MOV #MSLT17,EMADDR ;SET ERROR MESSAGE HEADER
1327 004376 012737 004404 000710 MOV #LT17A,SCOLP ;SET SCOPE LOOP ADDRESS
1328 004404 012737 144260 001022 LT17A: MOV #144260,WCS1 ;SET EXPECTED VALUES ON COMPLETION
1329 004412 012737 000100 001024 MOV #100,WCS2 ;OF WRAP 0 BELOW
1330 004420 012737 150600 001026 MOV #150600,WDS
1331 004426 012737 000200 001030 MOV #200,WER
1332 004434 012737 001700 000774 MOV #1700,UDES ;SET UNIT DESCRIPTION
1333 004442 005037 001020 CLR CRCFLG ;SET CRC CORRECTION NOT IN PROGRESS
1334 004446 004737 005200 JSR PC,WAM0 ;DO A WRAP 0 --- STEP A
1335 004452 000240 NOP
1336 004454 012737 000002 001020 MOV #2,CRCFLG ;SET CRC CORRECTION IN PROGRESS
1337 004462 013737 017744 004346 MOV WBUFF,CRCDAT ;GET DATA TO BE WRITTEN
1338 004470 043737 004344 004346 BIC CRCPAT,CRCDAT ;MODIFY DATA TO BE WRITTEN
1339 004476 012737 144260 001022 MOV #144260,WCS1 ;SET EXPECTED VALUES ON COMPLETION
1340 004504 012737 000110 001024 MOV #110,WCS2 ;OF WRAP 4 BELOW
1341 004512 012737 150600 001026 MOV #150600,WDS
1342 004520 012737 100300 001030 MOV #100300,WER
1343 004526 004737 005426 JSR PC,WAM4 ;DO A WRAP 4 --- STEP B
1344 004532 000240 NOP
1345 004534 012737 000001 001004 MOV #1,RDRVF ;SET READ REVERSE FLAG
1346 004542 012737 144276 001022 MOV #144276,WCS1 ;SET EXPECTED VALUES ON COMPLETION
1347 004550 012737 000110 001024 MOV #110,WCS2 ;OF WRAP 3 BELOW
1348 004556 012737 150600 001026 MOV #150600,WDS
1349 004564 012737 001000 001030 MOV #1000,WER
1350 004572 004737 005320 JSR PC,WAM3 ;DO A WRAP 3 --- STEP C
1351 004576 000240 NOP
1352 004600 012737 144260 001022 MOV #144260,WCS1 ;SET EXPECTED VALUES ON COMPLETION
1353 004606 012737 000110 001024 MOV #110,WCS2 ;OF WRAP 4 BELOW
1354 004614 012737 150600 001026 MOV #150600,WDS
1355 004622 012737 100100 001030 MOV #100100,WER
1356 004630 005037 001020 CLR CRCFLG ;CLEAR CRC IN PROGRESS FLAG
1357 004634 013701 004346 MOV CRCDAT,R1 ;GET DATA THAT WAS WRITTEN IN STEP B
1358 004640 012703 017744 MOV #WBUFF,R3 ;SET START OF WRITE BUFFER
1359 004644 004737 005144 JSR PC,DAT1A ;GO SET WRITE BUFFER
1360 004650 004737 005426 JSR PC,WAM4 ;GO DO A WRAP 4 --- STEP D
1361 004654 000240 NOP
1362 004656 004737 012150 JSR PC,ITER ;ITERATE TEST
1363 004662 000137 002214 JMP TSCD2 ;RETURN TO SCHEDULER

```

1364  
1365  
1366  
1367  
1368  
1369  
1370  
1371  
1372  
1373  
1374  
1375  
1376  
1377  
1378  
1379  
1380  
1381  
1382  
1383  
1384

004666 005037 001000  
004672 012737 144276 001022  
004700 012737 000100 001024  
004706 012737 150600 001026  
004714 012737 100000 001030  
004722 012737 016355 000620  
004730 012737 001700 000774  
004736 012737 004744 000710  
004744 012737 000001 001004  
004752 012737 000001 001016  
004760 004737 005320  
004764 005037 001004  
004770 005037 001016  
004774 004737 012150  
005000 000137 002214

```

;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,WAM3
      CLR      RDRVF        ;CLEAR READ REVERSE FLAG
      CLR      DCHKFL      ;CLEAR DATA CHECK FLAG
      JSR      PC,ITER
      JMP      TSCD2       ;RETURN TO SCHEDULER
```





1441 005170 000765

1442

1443

1444

1445 005172 012701 177377

1446 005176 000762

1447

BR DAT1A ;LOAD BUFFER

;ALTERNATING PARITY CHARACTERS\*\*\*\*\*

DATA: MOV #177377,R1 ;R1=ALTERNATING PARITY DATA  
BR DAT1A ;GO LOAD BUFFER

```

1448
1449 ;WRAP AROUND MODE 0 GLOBAL*****
1450
1451 005200 012737 000006 000746 WAM0: MOV #6,WAM ;SET WAM NUMBER
1452 005206 012737 000060 000750 WAM01: MOV #60,FUN
1453 005214 005037 000752 CLR DATC
1454 005220 012737 017744 000756 MOV #WBUFF,DATAD ;SET BUFFER ADDRESS
1455 005226 012737 020356 000760 MOV #RBUFF,RDAD ;SET POINTER TO READ BUFFER
1456 005234 004737 005472 JSR PC,SETUP ;GO SET UP
1457 005240 000137 006052 JMP EXEC
1458
1459 ;WRAP AROUND MODE 1 WRITE BUFFER*****
1460
1461 005244 012737 000010 000746 WAM1: MOV #10,WAM
1462 005252 000755 BR WAM01
1463
1464 ;WRAP AROUND MODE 2 BIT FIDDLER WRITE*****
1465
1466 005254 012737 000012 000746 WAM2: MOV #12,WAM
1467 005262 012737 000060 000750 MOV #60,FUN
1468 005270 005037 000752 CLR DATC
1469 005274 012737 017744 000756 MOV #WBUFF,DATAD
1470 005302 012737 020356 000760 MOV #RBUFF,RDAD
1471 005310 004737 005472 WAM2A: JSR PC,SETUP
1472 005314 000137 006052 JMP EXEC
1473
1474 ;WRAP AROUND MODE 3 BIT FIDDLER READ*****
1475
1476 005320 012737 000014 000746 WAM3: MOV #14,WAM ;SET WAM NUMBER
1477 005326 012737 000070 000750 MOV #70,FUN ;SET FUNCTION
1478 005334 012737 020356 000756 MOV #RBUFF,DATAD ;SET BUFFER ADDRESS
1479 005342 012737 017744 000754 MOV #WBUFF,WTAD ;SET POINTER TO WRITE BUFFER
1480 005350 005737 001004 TST RDRVF
1481 005354 001411 BEQ WAM3A
1482 005356 062737 000376 000756 ADD #376,DATAD
1483 005364 062737 000377 000754 ADD #377,WTAD
1484 005372 012737 000076 000750 MOV #76,FUN ;SET READ REVERSE CODE
1485 005400 032737 000020 000774 WAM3A: BIT #20,UNDES
1486 005406 001403 BEQ WAM3B
1487 005410 013737 001006 000754 MOV RCDP,WTAD
1488 005416 004737 005472 WAM3B: JSR PC,SETUP ;GO SET UP
1489 005422 000137 006052 JMP EXEC ;GO EXECUTE
1490
1491 ;WRAP AROUND MODE 4 CRC CORRECTION
1492 ;CALL: JSR PC,WAM4
1493
1494 005426 012737 000030 000746 WAM4: MOV #30,WAM ;SET MAINTENANCE MODE FUNCTION = WRAP 4
1495 005434 012737 000060 000750 MOV #60,FUN ;SET TAPE FUNCTION = WRITE FWD
1496 005442 005037 000752 CLR DATC
1497 005446 012737 004346 000756 MOV #CRCDAT,DATAD ;SET ADRS OF WRITE BUFFER
1498 005454 012737 020356 000760 MOV #RBUFF,RDAD ;SET READ BUFFER ADDRESS
1499 005462 004737 005472 JSR PC,SETUP ;GO SETUP REGISTERS
1500 005466 000137 006052 JMP EXEC ;GO EXECUTE

```



```

;REGISTER SETUP ROUTINE*****
1501
1502
1503 005472 005737 001020          SETUP: TST   CRCFLG      ;DO NOT DO DATA SETUP NOR INIT
1504 005476 001004                    BNE   1$          ;IF CRC CORRECTION IS IN PROGRESS
1505 005500 022737 000030 000746    CMP   #30,WAM    ;DO NOT INIT IF DOING WAM 4 --- STEP D
1506 005506 001004                    BNE   2$
1507 005510 012777 000011 172772  1$:  MOV   #11,@C1   ;DO DRIVE CLEAR IF WAM4---STEP D
1508 005516 000412                    BR    SET1A
1509 005520 005737 000736          2$:  TST   STFLG      ;SEE IF SINGLE TEST
1510 005524 001403                    BEQ   SET0       ;IF NOT: BR
1511 005526 005737 000776          TST   WPGFL      ;SEE IF HAVE SELECTED PATTERN
1512 005532 001002                    BNE   SET1       ;IF SO: BR
1513 005534 004737 005004          SET0: JSR   PC,DSUP ;GO DO DATA SETUP
1514 005540 004737 012220          SET1: JSR   PC,INIT ;INIT CONTROLLER,SELECT UNIT & DRIVE
1515
1516 005544 012777 177400 172744    SET1A: MOV   #-400,@FC ;SET FC=WCX2
1517 005552 032737 000020 000774    BIT   #20,UDES   ;SEE IF CORE DUMP
1518 005560 001403                    BEQ   SET2       ;IF NOT: BR
1519 005562 012777 177000 172726    MOV   #-1000,@FC ;SET FC=WCX4
1520 005570 012777 177600 172714    SET2: MOV   #-200,@WC ;SET WC
1521 005576 013777 000756 172710    MOV   DATAD,@BA  ;SET BUS ADDRESS
1522 005604 032777 010000 172706    BIT   #10000,@CS ;ASSURE DRIVE THERE
1523 005612 001417                    BEQ   SP1        ;IF SO: BR
1524 005614 032777 020000 172746    BIT   #20000,@SWR ;SEE IF PRINT ERRORS
1525 005622 001004                    BNE   SP01       ;IF NOT: BR
1526 005624 012704 017105          MOV   #WMSG4,R4
1527 005630 004737 013154          JSR   PC,TTOUT   ;PRINT NON-EXISTANT DRIVE
1528 005634 032777 100000 172726    SP01: BIT   #100000,@SWR ;SEE IF HALT ON ERROR
1529 005642 001401                    BEQ   SP0        ;IF NOT: BR
1530 005644 000000                    HALT
1531 005646 000137 005540          SP0:  JMP   SET1       ;RESETUP
1532 005652 022737 000014 000746    SP1:  CMP   #14,WAM ;SEE IF WAM 3
1533 005660 001031                    BNE   SP1B       ;IF NOT: BR
1534 005662 117737 173066 000752    MOVB  @WTAD,DATC ;GET FIRST CHAR
1535 005670 042737 177400 000752    BIC   #177400,DATC
1536 005676 000337 000752          SWAB  DATC
1537 005702 052737 000200 000752    BIS   #200,DATC  ;SET PARITY
1538 005710 005737 001004          TST   RDRVF      ;SEE IF READ REVERSE
1539 005714 001403                    BEQ   SP1A       ;IF NOT: BR
1540 005716 005337 000754          DEC   WTAD       ;DECREMENT POINTER
1541 005722 000410                    BR    SP1B
1542 005724 005237 000754          SP1A: INC   WTAD     ;BUMP POINTER
1543 005730 032737 000020 000774    BIT   #20,UDES   ;SEE IF CORE DUMP
1544 005736 001402                    BEQ   SP1B       ;IF NOT: BR
1545 005740 005237 000754          INC   WTAD     ;BUMP POINTER AGAIN
1546 005744 053777 000774 172570    SP1B: BIS   UDES,@TC ;SET UNIT DESCRIPTION (DEN,PAR,FMT)
1547 005752 052777 000001 172554    BIS   #1,@MR    ;SET MAINT MODE
1548 005760 053777 000746 172546    BIS   WAM,@MR   ;SET WAM
1549 005766 053777 000752 172540    BIS   DATC,@MR  ;SET DATA
1550 005774 013777 000750 172506    MOV   FUN,@C1   ;SET FUNCTION
1551 006002 032777 040000 172512    BIT   #40000,@DS ;ASSURE NO ERROR
1552 006010 001002                    BNE   SP3        ;IF ERROR: BR
1553 006012 000240                    NOP
1554 006014 000207                    RTS   PC        ;RETURN
1555 006016 032777 020000 172544    SP3:  BIT   #20000,@SWR ;SEE IF PRINT ERRORS
1556 006024 001004                    BNE   SP4        ;IF NOT: BR
  
```

1557	006026	012704	017046			MOV	#WMSG2,R4	
1558	006032	004737	013154			JSR	PC,TTOUT	;PRINT SETUP ERROR
1559	006036	032777	100000	172524	SP4:	BIT	#100000,@SWR	;SEE IF HALT ON ERROR
1560	006044	001401				BEQ	SP5	;IF NOT: BR
1561	006046	000000				HALT		
1562	006050	000207			SP5:	RTS	PC	;RETURN



```

;EXECUTE WAM ROUTINE*****
1563
1564
1565 006052 000240 EXEC: NOP
1566 006054 000240 NOP
1567 006056 032777 000040 172450 BIT #40,@MR
1568 006064 001403 BEQ EXO ;ASSURE MAINT CLOCK IS ZERO
1569 006066 042777 000040 172440 BIC #40,@MR ;IF NOT: CLEAR IT
1570 006074 022737 000010 000746 EX0: CMP #10,WAM ;BRANCH IF NOT WAM 0
1571 006102 003402 BLE 2$
1572 006104 000137 006530 1$: JMP EXW2 ;GO DO WAM 0
1573 006110 022737 000030 000746 2$: CMP #30,WAM ;BRANCH IF WRAP AROUND MODE 4
1574 006116 001772 BEQ 1$
1575 006120 052777 000001 172362 EX1: BIS #1,@C1 ;SET GO BIT
1576 006126 005000 1$: CLR RO
1577 006130 012701 000002 MOV #2,R1 ;SET DELAY
1578 006134 032777 100000 172400 EX1A: BIT #100000,@TC ;SEE IF ALPHA
1579 006142 001404 BEQ 1$ ;IF SO: BR
1580 006144 005300 DEC RO
1581 006146 001372 BNE EX1A ;AWAIT ALPHA
1582 006150 005301 DEC R1
1583 006152 001370 BNE EX1A
1584 006154 005737 000602 1$: TST SLVTYP ;BRANCH IF TE16
1585 006160 001407 BEQ EX2
1586 006162 022737 000014 000746 CMP #14,WAM ;BRANCH IF NOT WAM3
1587 006170 001003 BNE EX2
1588 006172 053777 000752 172334 BIS DATC,@MR ;LOAD DATA AND DO CLOCK UP
1589 006200 005077 172362 EX2: CLR @PSW
1590 006204 012701 000400 MOV #400,R1 ;SET NUMBER OF CLKS
1591 006210 032737 000020 000774 BIT #20,UDES ;SEE IF CORE DUMP
1592 006216 001402 BEQ EX3 ;IF NOT: BR
1593 006220 012701 001000 MOV #1000,R1 ;SET # OF CLOCKS = WORD COUNT X 4
1594 006224 022737 000014 000746 EX3: CMP #14,WAM ;BRANCH IF WAM 3
1595 006232 001412 BEQ EX5A
1596 006234 032737 002000 000774 BIT #2000,UDES ;BRANCH IF NRZ
1597 006242 001404 BEQ EX5
1598 006244 006301 ASL R1
1599 006246 062701 000246 ADD #246,R1 ;SET TO ALLOW FOR PRE/POSTAMBLE
1600 006252 000402 BR EX5A
1601 006254 062701 000010 EX5: ADD #10,R1 ;ADD CLOCKS FOR CRC AND LRC
1602 006260 022737 000014 000746 EX5A: CMP #14,WAM ;BRANCH IF NOT WAM 3
1603 006266 001065 BNE EX5C
1604 006270 117700 172460 MOVB @WTAD,RO ;GET DATA CHARACTER
1605 006274 042700 177400 BIC #177400,RO ;CLEAR UPPER BYTE
1606 006300 005737 001004 TST RDRVF ;BRANCH IF READ FORWARD
1607 006304 001403 BEQ EX5A1
1608 006306 005337 000754 DEC WTAD ;DEC POINTER
1609 006312 000416 BR EX5B
1610 006314 005237 000754 EX5A1: INC WTAD ;INCREMENT DATA PTR
1611 006320 032737 000020 000774 BIT #20,UDES ;BRANCH IF NOT CORE DUMP MODE
1612 006326 001410 BEQ EX5B
1613 006330 005237 000754 INC WTAD ;BUMP POINTER
1614 006334 005777 172414 TST @WTAD ;SEE IF END
1615 006340 001003 BNE EX5B ;IF NOT: BR
1616 006342 162737 000010 000754 SUB #10,WTAD ;RESTORE POINTER
1617 006350 052777 000040 172156 EX5B: BIS #40,@MR ;CLOCK UP (DOWN IF TU??)
1618 006356 017702 172152 MOV @MR,R2 ;READ MR

```



1619	006362	042702	177600			BIC	#177600,R2	;MASK OUT DATA
1620	006366	000300				SWAB	R0	;POSITION DATA
1621	006370	022700	177000			CMP	#177000,R0	;SEE IF PATTERN 3
1622	006374	001404				BEQ	2#	;IF 50: BR
1623	006376	000240				NOP		
1624	006400	000240				NOP		
1625	006402	052700	000200		1#:	BIS	#200,R0	;SET ODD PARITY
1626	006406	050002			2#:	BIS	R0,R2	;LOAD NEW DATA
1627	006410	005737	000602			TST	SLVTYP	;BRANCH IF TE16
1628	006414	001407				BEQ	4#	
1629	006416	005301				DEC	R1	;DECREMENT CLOCK COUNT
1630	006420	001403				BEQ	3#	
1631	006422	010277	172106			MOV	R2,#MR	;CLOCK DOWN (UP IF TU77)
1632	006426	000714				BR	EX5A	;CONTINUE
1633	006430	000137	011706		3#:	JMP	EORP	;GO TO END OF RECORD
1634								
1635	006434	010277	172074		4#:	MOV	R2,#MR	;DO CLOCK DOWN
1636	006440	000426				BR	EX5D	
1637								
1638	006442	052777	000040	172064	EX5C:	BIS	#40,#MR	;CLOCK UP (DOWN)
1639	006450	042777	000040	172056		BIC	#40,#MR	;CLOCK DOWN (UP)
1640	006456	017700	172052			MOV	#MR,R0	;GET MR
1641	006462	000300				SWAB	R0	
1642	006464	032737	000010	000774		BIT	#10,UDES	;SEE IF EVEN PAR
1643	006472	001405				BEQ	EX5C0	;IF NOT: BR
1644	006474	010077	172260			MOV	R0,#RDAD	
1645	006500	005237	000760			INC	RDAD	
1646	006504	000402				BR	EX5C1	
1647	006506	110077	172246		EX5C0:	MOVB	R0,#RDAD	;PUT CHAR IN CORE
1648	006512	005237	000760		EX5C1:	INC	RDAD	
1649	006516	000240			EX5D:	NOP		
1650	006520	005301				DEC	R1	;SEE IF DONE CLKS
1651	006522	001256				BNE	EX5A	;IF NOT: BR
1652	006524	000137	011706			JMP	EORP	;GO DO EOR

```

1653                                     ;EXECUTE WAM 0*****
1654
1655 006530 012737 006714 000666 EXW2:  MOV    @EXW2H,RTRN    ;SET INTERRUPT RETURN ADDRESS
1656 006536 012701 000200          MOV    @200,R1      ;SET NUMBER OF CLOCKS = FC/2
1657 006542 032737 002000 000774          BIT    @2000,UDES   ;SEE IF PE
1658 006550 001402          BEQ    EXW2A        ;IF NOT: BR
1659 006552 012701 000100          MOV    @100,R1     ;ELSE SET CLKS = FC/4
1660 006556 012702 020356          EXW2A: MOV    @RBUF,R2    ;SET BUFFER ADDRESS
1661 006562 022737 000030 000746          CMP    @30,WAM     ;BRANCH IF NOT WRAP 4
1662 006570 001003          BNE    1$
1663 006572 052777 000010 171720          BIS    @10,@CS     ;SET INHIBIT BUS ADDRESS INCREMENT
1664 006600 012777 000161 171702 1$:      MOV    @161,@C1    ;SET WRITE COMMAND AND GO
1665 006606 005077 171754          CLR    @PSW        ;ALLOW INTERRUPT
1666 006612 032777 000040 171714 EXW2B:  BIT    @40,@MR     ;AWAIT CLOCK UP
1667 006620 001774          BEQ    EXW2B        ;GET DATA
1668 006622 017722 171706          MOV    @MR,(R2)
1669 006626 032777 000040 171700 EXW2C:  BIT    @40,@MR     ;AWAIT CLOCK DOWN
1670 006634 001374          BNE    EXW2C        ;GET DATA
1671 006636 017722 171672          MOV    @MR,(R2)
1672 006642 005301          DEC    R1           ;SEE IF DONE ALL
1673 006644 001362          BNE    EXW2B        ;IF NOT: BR
1674 006646 012701 000003          EXW2E: MOV    @3,R1
1675 006652 005000          CLR    R0           ;SET DELAY
1676 006654 005300          EXW2F: DEC    R0
1677 006656 001376          BNE    EXW2F
1678 006660 005301          DEC    R1
1679 006662 001374          BNE    EXW2F        ;DELAY
1680 006664 032777 020000 171676          BIT    @20000,@SWR ;SEE IF ERROR PRINT
1681 006672 001004          BNE    EXW2G        ;IF NOT: BR
1682 006674 012704 017310          MOV    @WMSG24,R4
1683 006700 004737 013154          JSR    PC,TTOUT
1684 006704 005777 171660          EXW2G: TST    @SWR
1685 006710 100001          BPL    EXW2H
1686 006712 000000          HALT
1687 006714 000240          EXW2H: NOP
1688 006716 012701 020356          MOV    @RBUF,R1    ;GET START OF READ BUFFER
1689 006722 012700 000400          MOV    @400,R0     ;SET SIZE
1690 006726 010102          MOV    R1,R2
1691 006730 012203          EXW2J: MOV    (R2),R3
1692 006732 000303          SWAB   R3
1693 006734 032737 000010 000774          BIT    @10,UDES    ;SEE IF EVEN PAR
1694 006742 001402          BEQ    EXW2J0       ;IF NOT: BR
1695 006744 010321          MOV    R3,(R1)     ;SAVE PAR * DATA
1696 006746 000401          BR     EXW2J1
1697 006750 110321          EXW2J0: MOV    R3,(R1) ;ASSEMBLE DATA IN BYTES
1698 006752 005300          EXW2J1: DEC    R0
1699 006754 001365          BNE    EXW2J
1700 006756 032777 000200 171604          BIT    @200,@SWR   ;CONTINUE FOR ALL
1701 006764 001002          BNE    EXW2K        ;SEE IF STATUS CHECK
1702 006766 004737 007012          JSR    PC,WSTCK    ;IF NOT: BR
1703 006772 000240          EXW2K: NOP         ;ELSE GO CHECK STATUS
1704 006774 032777 000400 171566          BIT    @400,@SWR   ;SEE IF DATA CHECK
1705 007002 001002          BNE    EXW2X        ;IF NOT: BR
1706 007004 004737 007412          JSR    PC,DCHK     ;ELSE GO CHECK DATA
1707 007010 000207          EXW2X: RTS         ;EXIT

```



```

1708
1709                                     ;WRAP AROUND STATUS CHECK ROUTINE*****
1710
1711 007012 000240          WSTCK:  NOP
1712 007014 005037 000770      CLR      SERFL      ;CLEAR ERROR FLAG
1713 007020 005037 000616      CLR      HDRFL      ;CLEAR HEADER FLAG
1714 007024 005737 000602      TST      SLVTYP     ;BRANCH IF TU77
1715 007030 001004          BNE      10$
1716 007032 022737 015244 000620  CMP      @MSLT2,EMADDR ;SEE IF TEST 2
1717 007040 001404          BEQ      2$
1718 007042 022737 016134 000620 10$:  CMP      @MSLT15,EMADDR ;SEE IF TEST 15
1719 007050 001012          BNE      5$
1720 007052 022737 000003 001000 2$:  CMP      @3,PATRN     ;SEE IF PATTERN 3
1721 007060 001006          BNE      5$
1722 007062 052737 140000 001026 4$:  BIS      @140000,WDS   ;SET EXPT PARITY ERROR
1723 007070 052737 000100 001030 5$:  BIS      @100,WER     ;SET CODE-CS1
1724 007076 012737 017132 000670  MOV      @WMSG6,ERADD ;SET CODE-CS1
1725 007104 017702 171400      MOV      @C1,R2      ;GET RCVD CS1
1726 007110 042702 140000      BIC      @140000,R2   ;MASK OUT SC & TRE BITS
1727 007114 013705 001022      MOV      WCS1,R5     ;GET EXPT CS1
1728 007120 042705 140000      BIC      @140000,R5   ;MASK OUT SC & TRE BITS
1729 007124 004737 007266      JSR      PC,WSTG     ;GO CHK
1730 007130 012737 017157 000670  MOV      @WMSG6D,ERADD ;SET CODE-CS2
1731 007136 017702 171356      MOV      @CS,R2      ;SET RCVD CS2
1732 007142 042702 001000      BIC      @1000,R2    ;CLEAR MXF BIT (DON'T CARE BIT)
1733 007146 013705 001024      MOV      WCS2,R5     ;GET EXPT CS2
1734 007152 053705 000622      BIS      DRVN,R5     ;SET DRIVE NUMBER IN EXPT CS2
1735 007156 004737 007266      JSR      PC,WSTG     ;GO CHK
1736 007162 012737 017165 000670  MOV      @WMSG6E,ERADD ;SET CODE-DS
1737 007170 017702 171326      MOV      @DS,R2      ;SET RCVD DS
1738 007174 013705 001026      MOV      WDS,R5     ;GET EXPT DS
1739 007200 004737 007266      JSR      PC,WSTG     ;GO CHK
1740 007204 012737 017172 000670 1$:  MOV      @WMSG6F,ERADD ;SET CODE-ER
1741 007212 017702 171306      MOV      @ER,R2     ;GET RCVD ER
1742 007216 000240      NOP
1743 007220 000240      NOP
1744 007222 022737 016355 000620  CMP      @MSLT20,EMADDR ;SEE IF TEST 20
1745 007230 001002          BNE      12$
1746 007232 042702 001000      BIC      @1000,R2    ;MASK FCE
1747 007236 013705 001030      MOV      WER,R5     ;GET EXPT ER
1748 007242 004737 007266      JSR      PC,WSTG     ;GO CHK
1749 007246 005737 000770      TST      SERFL      ;SEE IF ANY ERRORS
1750 007252 001456          BEQ      WSTX
1751 007254 005777 171310      TST      @SWR
1752 007260 100053          BPL      WSTX
1753 007262 000000          HALT
1754 007264 000451          BR      WSTX        ;CONTINUE
1755 007266 000240          WSTG:  NOP
1756 007270 020205      CMP      R2,R5      ;SEE IF EXPT=RCVD
1757 007272 001446          BEQ      WSTX
1758 007274 005237 000770      INC      SERFL      ;SET ERROR FLAG
1759 007300 032777 020000 171262  BIT      @20000,@SWR  ;SEE IF PRINT ERRORS
1760 007306 001040          BNE      WSTX
1761 007310 005737 000616      TST      HDRFL      ;SEE IF DONE HEADER
1762 007314 001010          BNE      WSTG
1763 007316 013704 000620      MOV      EMADDR,R4

```



1764	007322	004737	013154		JSR	PC,TTOUT	;PRINT TEST HEADER	
1765	007326	012704	017274		MOV	#WMSG23,R4		
1766	007332	004737	013154		JSR	PC,TTOUT	;PRINT STATUS TAG	
1767	007336	012737	000001	000616	WSTGO:	MOV	#1,HDRFL	;SET HEADER FLAG
1768	007344	013704	000670		MOV	ERADD,R4		
1769	007350	004737	013154		JSR	PC,TTOUT	;PRINT CODE	
1770	007354	012704	014516		MOV	#MSG12,R4		
1771	007360	004737	013154		JSR	PC,TTOUT	;PRINT EXPT TAG	
1772	007364	010503			MOV	R5,R3		
1773	007366	004737	013366		JSR	PC,OCTP	;PRINT EXPT STATUS	
1774	007372	012704	014525		MOV	#MSG13,R4		
1775	007376	004737	013154		JSR	PC,TTOUT	;PRINT RCVD TAG	
1776	007402	010203			MOV	R2,R3		
1777	007404	004737	013366		JSR	PC,OCTP	;PRINT RCVD STATUS	
1778	007410	000207			WSTX:	RTS	;RETURN	
1779								

```

1780
1781 ;DATA CHECK ROUTINE*****
1782
1783 007412 000240          DCHK:  NOP
1784 007414 005737 001020  TST      CRCFLG      ;DO NOT DO A DATA CHACK
1785 007420 001402          BEQ      2$          ;IF CRC CORRECTION IN PROGRESS
1786 007422 000137 010252  1$:    JMP      DCHKX1
1787 007426 005737 001016  2$:    TST      DCHKFL      ;BRANCH IF DATA IS NOT TO BE CHECKED
1788 007432 001373          BNE      1$
1789 007434 005037 000616  CLR      HDRFL      ;CLEAR HEADER FLAG
1790 007440 005037 000764  CLR      DERFL      ;CLEAR DATA ERROR FLAG
1791 007444 005037 000772  CLR      CRCNT      ;CLEAR CHAR CNTR
1792 007450 032737 000010 000774 BIT      @10,UDES    ;SEE IF EVEN PARITY
1793 007456 001402          BEQ      DCHKA0      ;IF NOT: BR
1794 007460 000137 010274  JMP      DCHKE      ;ELSE GO CHECK EVEN
1795 007464 022737 000010 000746 DCHKA0: CMP      @10,WAM    ;SEE IF WAM 1
1796 007472 001006          BNE      DCHKA      ;IF NOT: BR
1797 007474 032737 002000 000774 BIT      @2000,UDES  ;SEE IF PE
1798 007502 001402          BEQ      DCHKA      ;IF NOT: BR
1799 007504 000137 010776  JMP      PRCHK      ;GO CHK DATA
1800 007510 012700 177400  DCHKA:  MOV      @-400,R0 ;SET NUMBER OF CHARACTERS
1801 007514 022737 000012 000746  CMP      @12,WAM
1802 007522 001006          BNE      DCHKA1      ;IF NOT WRAP 2: BR
1803 007524 032737 000020 000774  BIT      @20,UDES
1804 007532 001402          BEQ      DCHKA1      ;IF NOT CORE DUMP: BR
1805 007534 012700 177000  MOV      @-1000,R0
1806 007540 022737 000030 000746 DCHKA1: CMP      @30,WAM    ;BRANCH IF WRAP 4
1807 007546 001404          BEQ      1$
1808 007550 022737 000006 000746  CMP      @6,WAM      ;SEE IF WRAP 0
1809 007556 001007          BNE      DCHKA2      ;IF NOT: BR
1810 007560 012700 177744  1$:    MOV      @-34,R0  ;SET NUMBER OF CHARACTERS READ
1811 007564 012701 017754  MOV      @WBUF+10,R1 ;SET POINTER
1812 007570 005037 000764  CLR      DERFL      ;CLEAR DATA ERROR FLAG
1813 007574 000431          BR       DCHKB0
1814 007576 022737 000012 000746 DCHKA2: CMP      @12,WAM    ;SEE IF WRAP 2
1815 007604 001021          BNE      DCHKB      ;IF NOT: BR
1816 007606 032737 002000 000774  BIT      @2000,UDES  ;SEE IF PE
1817 007614 001415          BEQ      DCHKB      ;IF NOT: BR
1818 007616 012700 177653  MOV      @-125,R0   ;POINT TO START OF DATA
1819 007622 012737 000001 000762  MOV      @1,W2FLG    ;SET WRAP 2 FLAG
1820 007630 004737 007650  JSR      PC,DCHKB    ;GO CHECK DATA
1821 007634 004737 011276  JSR      PC,WIDCHK   ;GO CHECK WRAP 1 DATA
1822 007640 005037 000762  CLR      W2FLG
1823 007644 000137 010234  JMP      DCHKX
1824 007650 005037 000764  DCHKB:  CLR      DERFL
1825 007654 012701 017744  MOV      @WBUF,R1   ;SET GOOD POINTER
1826 007660 012702 020356  DCHKB0: MOV      @RBUF,R2  ;SET READ POINTER
1827 007664 032737 000020 000774  BIT      @20,UDES    ;SEE IF CORE DUMP
1828 007672 001416          BEQ      DCHKO      ;IF NOT: BR
1829 007674 022737 000012 000746  CMP      @12,WAM    ;SEE IF WAM 2
1830 007702 001011          BNE      DCHKD      ;IF NOT: BR
1831 007704 005737 001000  TST      PATRN      ;SEE IF PATTERN 0
1832 007710 001003          BNE      DCHKC      ;IF NOT: BR
1833 007712 012701 001054  MOV      @WCDPO,R1  ;SET CORE DUMP PATTERN 0
1834 007716 000404          BR       DCKO
1835 007720 012701 001042  DCHKC:  MOV      @WCDP2,R1 ;SET CORE DUMP WRITE PATTERN 2

```



1836	007724	000401				BR	DCHKO		;GO CHECK DATA
1837	007726	000240				DCHKD: NOP			
1838	007730	121112				DCHKO: CMPB	(R1),(R2)		;SEE IF DATA OK
1839	007732	001466				BEQ	DCHK2		;IF SO: BR
1840	007734	032777	020000	170626		BIT	#20000,@SWR		;SEE IF PRINT ERRORS
1841	007742	001062				BNE	DCHK2		;IF NOT: BR
1842	007744	005737	000616			TST	HDRFL		;SEE IF DONE HEADER
1843	007750	001004				BNE	DCHK1		;IF SO: BR
1844	007752	013704	000620			MOV	EMADDR,R4		
1845	007756	004737	013154			JSR	PC,TTOUT		;PRINT HEADER
1846	007762	005737	000764			DCHK1: TST	DERFL		;SEE IF FIRST ERROR
1847	007766	001014				BNE	DCHK1A		;IF NOT: BR
1848	007770	012704	017242			MOV	#WMSG16,R4		
1849	007774	004737	013154			JSR	PC,TTOUT		;PRINT DATA ERROR TAG
1850	010000	012704	017467			MOV	#WMSG32,R4		
1851	010004	004737	013154			JSR	PC,TTOUT		;PRINT PATRN TAG
1852	010010	013703	001000			MOV	PATRN,R3		
1853	010014	004737	013366			JSR	PC,OCTP		;PRINT PATTERN NUMBER
1854	010020	012737	000001	000616		DCHK1A: MOV	#1,HDRFL		;SET HEADER FLAG
1855	010026	012737	000001	000764		MOV	#1,DERFL		;SET DATA ERROR FLAG
1856	010034	012704	017266			MOV	#WMSG21,R4		
1857	010040	004737	013154			JSR	PC,TTOUT		;PRINT CHARACTER NUMBER TAG
1858	010044	013703	000772			MOV	CRCNT,R3		
1859	010050	004737	013366			JSR	PC,OCTP		;PRINT CHARACTER NUMBER
1860	010054	012704	017254			MOV	#WMSG17,R4		
1861	010060	004737	013154			JSR	PC,TTOUT		;PRINT GOOD TAG
1862	010064	111103				MOVB	(R1),R3		
1863	010066	004737	013614			JSR	PC,DOUT		;PRINT GOOD DATA
1864	010072	012704	017261			MOV	#WMSG20,R4		
1865	010076	004737	013154			JSR	PC,TTOUT		;PRINT BAD TAG
1866	010102	111203				MOVB	(R2),R3		
1867	010104	004737	013614			JSR	PC,DOUT		;PRINT BAD DATA
1868	010110	005737	000762			DCHK2: TST	W2FLG		;SEE IF WRAP 2 NRZ
1869	010114	001020				BNE	DCHK2B		;IF SO: BR
1870	010116	005201				INC	R1		;BUMP POINTER
1871	010120	032737	000020	000774		BIT	#20,UDES		;SEE IF CORE DUMP
1872	010126	001413				BEQ	DCHK2B		;IF NOT: BR
1873	010130	022737	000012	000746		CMP	#12,WAM		;SEE IF WAM 2
1874	010136	001006				BNE	DCHK2A		;IF NOT: BR
1875	010140	005201				INC	R1		;BUMP POINTER
1876	010142	005711				TST	(R1)		;SEE IF END OF PATTERN
1877	010144	001004				BNE	DCHK2B		;IF NOT: BR
1878	010146	162701	000010			SUB	#10,R1		;RESET POINTER TO START OF PATTERN
1879	010152	000401				BR	DCHK2B		;CONTINUE CHECK
1880	010154	000240				DCHK2A: NOP			
1881	010156	005202				DCHK2B: INC	R2		
1882	010160	022737	000030	000746		CMP	#30,WAM		;BRANCH IF WAM 4
1883	010166	001404				BEQ	1\$		
1884	010170	022737	000006	000746		CMP	#6,WAM		;SEE IF WAM 0
1885	010176	001002				BNE	DCHK3		;IF NOT: BR
1886	010200	062701	000010			ADD	#10,R1		;BUMP POINTER
1887	010204	005237	000772			DCHK3: INC	CRCNT		;BUMP CHAR CNTR
1888	010210	032777	000400	170352		BIT	#400,@SWR		;SEE IF CONT DATA CHK
1889	010216	001006				BNE	DCHKX		;IF NOT: BR
1890	010220	005200				INC	RO		;SEE IF DONE
1891	010222	001242				BNE	DCHKO		;IF NOT: BR



1892	010224	005737	000762
1893	010230	001401	
1894	010232	000207	
1895	010234	005777	170330
1896	010240	100004	
1897	010242	005737	000764
1898	010246	001401	
1899	010250	000000	
1900	010252	005037	000772
1901	010256	005037	000616
1902	010262	005037	000764
1903	010266	005037	000766
1904	010272	000207	

	TST	W2FLG
	BEQ	DCHKX
	RTS	PC
DCHKX:	TST	@SWR
	BPL	DCHKX1
	TST	DERFL
	BEQ	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

```

```

1905
1906
1907
1908 010274 000240
1909 010276 022737 000006 000746
1910 010304 001005
1911 010306 012700 177744
1912 010312 012701 017754
1913 010316 000404
1914 010320 012700 177400
1915 010324 012701 017744
1916 010330 012702 020356
1917 010334 111105
1918 010336 005003
1919 010340 012704 000010
1920 010344 032705 000001
1921 010350 001401
1922 010352 005203
1923 010354 005304
1924 010356 001402
1925 010360 006005
1926 010362 000770
1927 010364 000240
1928 010366 111105
1929 010370 042705 177400
1930 010374 005703
1931 010376 001003
1932 010400 012705 100020
1933 010404 000405
1934 010406 032703 000001
1935 010412 001402
1936 010414 052705 100000
1937 010420 042712 077400
1938 010424 020512
1939 010426 001477
1940 010430 032777 020000 170132
1941 010436 001073
1942 010440 005737 000616
1943 010444 001004
1944 010446 013704 000620
1945 010452 004737 013154
1946 010456 005737 000764
1947 010462 001014
1948 010464 012704 017242
1949 010470 004737 013154
1950 010474 012704 017467
1951 010500 004737 013154
1952 010504 013703 001000
1953 010510 004737 013366
1954 010514 000240
1955 010516 012737 000001 000764
1956 010524 012737 000001 000616
1957 010532 012704 017266
1958 010536 004737 013154
1959 010542 013703 000772
1960 010546 004737 013366

;EVEN PARITY DATA CHECK*****
DCHKE: NOP
      CMP #6,WAM ;SEE IF WRAP 0
      BNE 1$ ;IF NOT: BR
      MOV #34,R0 ;SET NUMBER OF CHARACTERS READ
      MOV #WBUF+10,R1 ;SET POINTER
      BR 2$
1$: MOV #-400,R0 ;SET NUMBER OF CHARACTERS
   MOV #WBUF,R1 ;R1=START OF WRITE BUFFER
   MOV #RBUF,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 HDRFL ;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 #WMSG16,R4
      JSR PC,TTOUT ;PRINT BAD DATA TAG
      MOV #WMSG32,R4
      JSR PC,TTOUT ;PRINT PATTERN TAG
      MOV PATRN,R3
      JSR PC,OCIP ;PRINT PATTERN NUMBER
DCKE7: NOP
      MOV #1,DERFL ;SET DATA ERROR FLAG
      MOV #1,HDRFL ;SET HEADER FLAG
      MOV #WMSG21,R4
      JSR PC,TTOUT ;PRINT CHAR NUMBER TAG
      MOV CRCNT,R3
      JSR PC,OCIP ;ORINT CHAR NUMBER
  
```

1961	010552	012704	017254		MOV	#WMSG17,R4	
1962	010556	004737	013154		JSR	PC,TTOUT	;PRINT GOOD DATA TAG
1963	010562	110503			MOVB	R5,R3	
1964	010564	004737	013614		JSR	PC,DOUT	;PRINT EXPT DATA
1965	010570	010503			MOV	R5,R3	
1966	010572	004737	010702		JSR	PC,DCKEP	;GO PRINT PARITY BIT
1967	010576	000240			NOP		
1968	010600	012704	017261		MOV	#WMSG20,R4	
1969	010604	004737	013154		JSR	PC,TTOUT	;PRINT BAD TAG
1970	010610	111203			MOVB	(R2),R3	
1971	010612	004737	013614		JSR	PC,DOUT	;PRINT BAD DATA
1972	010616	011203			MOV	(R2),R3	
1973	010620	004737	010702		JSR	PC,DCKEP	;GO PRINT PARITY BIT
1974	010624	000240			NOP		
1975	010626	005201			DCKE10: INC	R1	
1976	010630	022737	000006	000746	CMP	#6,WAM	;SEE IF WRAP 0
1977	010636	001002			BNE	1\$	;IF NOT: BR
1978	010640	062701	000010		ADD	#10,R1	;BUMP POINTER
1979	010644	005722			1\$: TST	(R2)+	;BUMP POINTERS
1980	010646	005237	000772		INC	CRCNT	;BUMP CHAR CNTR
1981	010652	032777	000400	167710	BIT	#400,BSWR	;SEE IF CONTINUE DATA CHECK
1982	010660	001402			BEQ	DCKE11	;IF SO: BR
1983	010662	000137	010234		JMP	DCHKX	;GO TO END OF DATA CHECK
1984	010666	005200			DCKE11: INC	R0	;SEE IF DONE
1985	010670	001402			BEQ	DCKE12	;IF SO: BR
1986	010672	000137	010334		JMP	DCKE0	;ELSE CONTINUE
1987	010676	000137	010234		DCKE12: JMP	DCHKX	;GO TO END OF DATA CHECK
1988	010702	000240			DCKEP: NOP		
1989	010704	012737	000240	000612	MOV	#240,TOB	
1990	010712	004737	013254		JSR	PC,TOG	;SPACE
1991	010716	012737	000260	000612	MOV	#260,TOB	;SET PAR=0
1992	010724	005703			TST	R3	;SEE IF PARITY REALLY=0
1993	010726	100002			BPL	DCKEPO	;IF SO: BR
1994	010730	005237	000612		INC	TOB	;ELSE SET TO 1
1995	010734	004737	013254		DCKEPO: JSR	PC,TOG	;PRINT PARITY BIT
1996	010740	000207			RTS	PC	;RETURN
1997							



```

1998
1999
2000
2001 010742 012700 000051          PSCHK: MOV #51,R0          ;SET SIZE OF POSTAMBLE
2002 010746 012701 017622          MOV #POST,R1          ;SET POINTER TO POSTAMBLE
2003 010752 005037 000616          CLR HDRFL             ;CLEAR HEADER FLAG
2004 010756 005037 000772          CLR CRCNT            ;CLEAR CHAR CNTR
2005 010762 005037 000764          CLR DERFL            ;CLEAR DATA ERROR FLAG
2006 010766 000240
2007 010770 000240
2008 010772 000137 011014          JMP PDO              ;GO CHECK POSTAMBLE
2009
2010 010776 012700 000051          PRCHK: MOV #51,R0          ;SET SIZE OF PREAMBLE
2011 011002 012701 017500          MOV #PRE,R1          ;SET POINTER TO PREAMBLE
2012 011006 012702 020356          MOV #RBUF,R2          ;SET POINTER TO START OF READ BUFFER
2013 011012 022122          CMP (R1), (R2)        ;BUMP ADDRESS POINTERS
2014 011014 121112          PDO:  CMPB (R1), (R2)    ;CHECK DATA
2015 011016 001004          BNE PD1              ;IF NOT GOOD: BR
2016 011020 126162 000001 000001  CMPB 1(R1), 1(R2)     ;COMPARE COMPLIMENT BYTE
2017 011026 001477          BEQ PD5              ;IF GOOD: BR
2018 011030 032777 020000 167532 PD1:  BIT #20000, @SWR      ;SEE IF PRINT INHIBIT
2019 011036 001073          BNE PD5              ;IF SO: BR
2020 011040 005737 000616          TST HDRFL            ;SEE IF DONE HEADER
2021 011044 001020          BNE PD4              ;IF SO: BR
2022 011046 013704 000620          MOV EMADDR, R4
2023 011052 004737 013154          JSR PC, TTOUT        ;PRINT TEST HEADER
2024 011056 005737 000766          TST PREFL            ;SEE IF PREAMBLE CHECK
2025 011062 001403          BEQ PD2              ;IF NOT: BR
2026 011064 012704 017413          MOV #WMSG29, R4      ;SET POSTAMBLE HEADER
2027 011070 000402          BR PD3
2028 011072 012704 017375          PD2:  MOV #WMSG28, R4    ;SET PREAMBLE HEADER
2029 011076 004737 013154          PD3:  JSR PC, TTOUT    ;PRINT HEADER
2030 011102 005237 000616          INC HDRFL
2031 011106 012704 017266          PD4:  MOV #WMSG21, R4
2032 011112 004737 013154          JSR PC, TTOUT        ;PRINT CHAR NUMBER TAG
2033 011116 013703 000772          MOV CRCNT, R3
2034 011122 004737 013366          JSR PC, OCTP         ;PRINT CHAR NUMBER
2035 011126 012704 017254          MOV #WMSG17, R4
2036 011132 004737 013154          JSR PC, TTOUT        ;PRINT GOOD TAG
2037 011136 116103 000001          MOVB 1(R1), R3
2038 011142 004737 013614          JSR PC, DOUT         ;PRINT GOOD CHAR
2039 011146 012737 000240 000612  MOV #240, TOB
2040 011154 004737 013254          JSR PC, TOG
2041 011160 111103          MOVB (R1), R3
2042 011162 004737 013614          JSR PC, DOUT         ;PRINT COMPLEMENT
2043 011166 012704 017261          MOV #WMSG20, R4
2044 011172 004737 013154          JSR PC, TTOUT        ;PRINT BAD TAG
2045 011176 116203 000001          MOVB 1(R2), R3
2046 011202 004737 013614          JSR PC, DOUT         ;PRINT BAD CHAR
2047 011206 012737 000240 000612  MOV #240, TOB
2048 011214 004737 013254          JSR PC, TOG
2049 011220 111203          MOVB (R2), R3
2050 011222 004737 013614          JSR PC, DOUT         ;PRINT COMPLEMENT
2051 011226 022122          PD5:  CMP (R1), (R2)        ;BUMP ADDRESS POINTERS
2052 011230 005237 000772          INC CRCNT            ;BUMP CHAR NUMBER
2053 011234 005300          DEC R0               ;SEE IF DONE

```

2054	011236	001266	
2055	011240	005737	000766
2056	011244	001402	
2057	011246	000137	010234
2058	011252	005237	000766
2059	011256	005037	000616
2060	011262	005037	000772
2061	011266	005037	000764
2062	011272	000137	011276
2063			

PD6:

BNE	PDO
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

```

```

2064
2065
2066
2067 011276 012700 177400          W1DCHK: MOV    #-400,R0          ;SET NUMBER OF CHAR TO CHECK
2068 011302 012701 017744          MOV    #WBUF,R1          ;SET WRITE DATA POINTER
2069 011306 012702 020356          MOV    #RBUF,R2          ;SET READ DATA POINTER
2070 011312 062702 000124          ADD    #124,R2           ;POINT TO START OF DATA
2071 011316 005737 000762          TST    W2FLG             ;SEE IF WRAP 2
2072 011322 001401                   BEQ    W1D0              ;IF NOT WAM 2: BR
2073 011324 005302                   DEC    R2                ;RESET POINTER
2074 011326 111105          W1D0:  MOVB   (R1),R5
2075 011330 120512          CMPB   R5,(R2)           ;CHECK DATA
2076 011332 001007          BNE    W1D1              ;IF NOT GOOD:BR
2077 011334 005737 000762          TST    W2FLG             ;SEE IF WRAP 2
2078 011340 001001          BNE    W1D0A            ;IF SO: BR
2079 011342 105105          COMB   R5                ;COMPLIMENT EXPT DATA
2080 011344 120562 000001          W1D0A: CMPB   R5,1(R2)    ;CHECK COMPLIMENT DATA
2081 011350 001510          BEQ    W1D3              ;IF GOOD: BR
2082 011352 032777 020000 167210 W1D1:  BIT    #20000,@SWR      ;SEE IF PRINT INHIBIT
2083 011360 001104          BNE    W1D3              ;IF SO: BR
2084 011362 005737 000616          TST    HDRFL            ;SEE IF DONE HEADER
2085 011366 001020          BNE    W1D2              ;IF SO: BR
2086 011370 013704 000620          MOV    EMADDR,R4
2087 011374 004737 013154          JSR    PC,TTOUT          ;PRINT TEST HEADER
2088 011400 012704 017242          MOV    #WMSG16,R4
2089 011404 004737 013154          JSR    PC,TTOUT          ;PRINT BAD DATA TAG
2090 011410 012704 017467          MOV    #WMSG32,R4
2091 011414 004737 013154          JSR    PC,TTOUT          ;PRINT PATRN TAG
2092 011420 013703 001000          MOV    PATRN,R3
2093 011424 004737 013366          JSR    PC,OCTP           ;PRINT PATTERN NUMBER
2094 011430 012737 000001 000616 W1D2:  MOV    #1,HDRFL         ;SET HEADER FLAG
2095 011436 012704 017266          MOV    #WMSG21,R4
2096 011442 004737 013154          JSR    PC,TTOUT          ;PRINT CHAR NUMBER TAG
2097 011446 013703 000772          MOV    CRCNT,R3
2098 011452 004737 013366          JSR    PC,OCTP           ;PRINT CHAR NUMBER
2099 011456 012704 017254          MOV    #WMSG17,R4
2100 011462 004737 013154          JSR    PC,TTOUT          ;PRNT GOOD TAG
2101 011466 111105          MOVB   (R1),R5
2102 011470 110503          MOVB   R5,R3             ;GET GOOD CHAR
2103 011472 005737 000762          TST    W2FLG             ;SEE IF WRAP 2
2104 011476 001001          BNE    W1D2A            ;IF SO: BR
2105 011500 105103          COMB   R3                ;ELSE COMPLIMENT CHAR
2106 011502 004737 013614          W1D2A: JSR    PC,DOUT          ;PRINT CHARACTER
2107 011506 012737 000240 000612  MOV    #240,TOB
2108 011514 004737 013254          JSR    PC,TOG            ;SPACE
2109 011520 110503          MOVB   R5,R3
2110 011522 004737 013614          JSR    PC,DOUT          ;PRINT CHAR
2111 011526 012704 017261          MOV    #WMSG20,R4
2112 011532 004737 013154          JSR    PC,TTOUT          ;PRINT BAD TAG
2113 011536 116203 000001          MOVB   1(R2),R3
2114 011542 004737 013614          JSR    PC,DOUT          ;PRINT BAD CHAR
2115 011546 012737 000240 000612  MOV    #240,TOB
2116 011554 004737 013254          JSR    PC,TOG            ;SPACE
2117 011560 111203          MOVB   (R2),R3
2118 011562 004737 013614          JSR    PC,DOUT          ;PRINT CHAR
2119 011566 005237 000764          INC    DERFL            ;SET DATA ERROR FLAG
  
```



2120	011572	122122		W1D3:	CMPB	(R1)+,(R2)+		;BUMP ADDRESS
2121	011574	105722			TSTB	(R2)+		;BUMP ADDRESS
2122	011576	005237	000772		INC	CRCNT		;BUMP CHAR CNTR
2123	011602	000406			BR	W1D5		
2124	011604	005737	000762	W1D4:	TST	W2FLG		;SEE IF WRAP 2
2125	011610	001401			BEQ	W1D4A		;IF NOT: BR
2126	011612	000207			RTS	PC		;ELSE RETURN
2127	011614	000137	010742	W1D4A:	JMP	PSCHK		;GO CHECK POSTAMBLE
2128	011620	005200		W1D5:	INC	RO		
2129	011622	001770			BEQ	W1D4		
2130	011624	000137	011326		JMP	W1D0		
2131								
2132								
2133								
2134	011630	000240		PPGEN:	NOP			
2135	011632	012700	000050		MOV	#50,RO		;SET SIZE OF PREAMBLE
2136	011636	012701	017500		MOV	#PRE,R1		
2137	011642	005721			TST	(R1)+		;SET ADDRESS OF PRE
2138	011644	012721	177400	1\$:	MOV	#177400,(R1)+		;FILL TABLE
2139	011650	005300			DEC	RO		;SEE IF DONE
2140	011652	001374			BNE	1\$		;IF NOT: BR
2141	011654	012701	017622		MOV	#POST,R1		;SET ADDRESS OF POST
2142	011660	012700	000050		MOV	#50,RO		;SET SIZE OF POST
2143	011664	012721	000377		MOV	#377,(R1)+		;SET SYNC CHAR
2144	011670	012721	177400	2\$:	MOV	#177400,(R1)+		;FILL TABLE
2145	011674	005300			DEC	RO		;SEE IF DONE
2146	011676	001374			BNE	2\$		;IF NOT: BR
2147	011700	000207			RTS	PC		;RETURN

;PREAMBLE/POSTAMBLE GENERATE SUBROUTINE\*\*\*\*\*

```

2148
2149
2150
2151 011702 005237 000674      EORPA: INC      TEMP2      ;SET WRAP FLAG
2152 011706 017700 166622      EORP:  MOV      @MR,R0      ;GET MAINT REG
2153 011712 042700 000036      BIC      @36,R0      ;CLEAR CURRENT OP CODE
2154 011716 052700 000024      BIS      @24,R0      ;SET EOR CLEAR OP CODE
2155 011722 010077 166606      MOV      R0,@MR      ;DO EOR
2156 011726 042777 000037 166600  BIC      @37,@MR      ;CLEAR EOR AND MM
2157 011734 005000      CLR      R0
2158 011736 012701 000002      MOV      @2,R1
2159 011742 032777 000001 166540  EORP1: BIT      @1,@C1      ;SEE IF GO GONE
2160 011750 001427      BEQ      EORP2      ;IF SO: BR
2161 011752 005300      DEC      R0
2162 011754 001372      BNE      EORP1      ;AWAIT GO RESET
2163 011756 005301      DEC      R1
2164 011760 001370      BNE      EORP1
2165 011762 032777 020000 166600  BIT      @20000,@SWR      ;SEE IF ERROR PRINT INHIBIT
2166 011770 001017      BNE      EORP2      ;IF SO: BR
2167 011772 005737 000616      TST      HDRFL      ;SEE IF DONE HEADER
2168 011776 001004      BNE      EORP1A      ;IF SO: BR
2169 012000 013704 000620      MOV      EMADDR,R4
2170 012004 004737 013154      JSR      PC,TTOUT      ;PRINT HEADER
2171 012010 012704 017432      EORP1A: MOV      @MSG31,R4
2172 012014 004737 013154      JSR      PC,TTOUT      ;PRINT EOR GO BIT ERROR
2173 012020 005777 166544      TST      @SWR      ;SEE IF HALT ON ERROR
2174 012024 100001      BPL      EORP2      ;IF NOT: BR
2175 012026 000000      HALT
2176 012030 000240      EORP2: NOP
2177 012032 005737 000674      TST      TEMP2      ;SEE IF WAM
2178 012036 001024      BNE      EORPX      ;IF NOT: BR
2179 012040 022737 000014 000746  CMP      @14,WAM      ;BRANCH IF NOT WRAP 3
2180 012046 001003      BNE      2:
2181 012050 105777 166434      1: TSTB      @C1      ;WAIT FOR READY
2182 012054 100375      BPL      1:
2183 012056 032777 000200 166504  2: BIT      @200,@SWR      ;SEE IF STATUS CHECK
2184 012064 001002      BNE      EORP3      ;IF NOT: BR
2185 012066 004737 007012      JSR      PC,WSTCK      ;ELSE GO CHECK STATUS
2186 012072 000240      EORP3: NOP
2187 012074 032777 000400 166466  BIT      @400,@SWR      ;SEE IF DATA CHECK
2188 012102 001002      BNE      EORPX      ;IF NOT: BR
2189 012104 004737 007412      JSR      PC,DCHK      ;ELSE GO CHECK DATA
2190 012110 000240      EORPX: NOP
2191 012112 005037 000674      CLR      TEMP2      ;CLEAR FLAG
2192 012116 000207      RTS      PC      ;RETURN
2193

```

```

2194
2195 ;SCOPE LOOP ON ERROR SUBROUTINE*****
2196
2197 012120 000240 SCOPE: NOP
2198 012122 032777 040000 166440 BIT @40000,@SWR ;SEE IF LOOP ON ERROR
2199 012130 001001 BNE 1$ ;IF SO: BR
2200 012132 000207 RTS PC ;ELSE EXIT
2201 012134 000240 1$: NOP
2202 012136 005726 TST (SP), ;RESET STACK
2203 012140 000240 NOP
2204 012142 000240 NOP
2205 012144 000177 166540 JMP @SCOLP ;LOOP ON ERROR
2206
2207 ;TEST ITERATION SUBROUTINE*****
2208
2209 012150 032777 004000 166412 ITER: BIT @4000,@SWR ;SEE IF ITERATIONS
2210 012156 001403 BEQ 2$ ;IF SO: BR
2211 012160 005037 000700 1$: CLR ITCNT ;CLEAR ITERATION COUNTER
2212 012164 000207 RTS PC ;ELSE EXIT
2213 012166 005737 001014 2$: TST @@PCNTR ;DO SINGLE SUBTEST ITERATION
2214 012172 001772 BEQ 1$ ;ON FIRST PASS
2215 012174 005237 000700 INC ITCNT ;BUMP COUNTER
2216 012200 023737 000700 000604 CMP ITCNT,ITAMT ;SEE IF DONE ALL
2217 012206 001764 BEQ 1$ ;IF SO: BR
2218 012210 005726 TST (SP), ;RESET STACK
2219 012212 017700 166474 MOV @ITRLP,R0 ;SET ITERATION POINTER
2220 012216 000110 JMP (R0) ;GO ITERATE
2221

```



```

2222
2223
2224
2225 012220 012777 000040 166272 INIT: MOV #40, @CS ;INIT
2226 012226 013777 000622 166264 MOV DRVN, @CS ;SELECT DRIVE
2227 012234 013777 000662 166300 MOV SLVN, @TC ;SELECT SLAVE
2228 012242 013746 000774 MOV UDES, -(SP) ;GET TEST'S UNIT DESCRIPTION
2229 012246 042716 174377 BIC #174377, (SP) ;CLEAR ALL BUT DENSITY SELECT BITS
2230 012252 022726 001400 CMP #1400, (SP) ;BRANCH IF NOT NRZ (800 BPI)
2231 012256 001005 BNE 1$
2232 012260 032777 000040 166234 BIT #40, @DS ;BRANCH IF SLAVE IS IN NRZ MODE
2233 012266 001420 BEQ 4$ ;(PES = 0)
2234 012270 000404 BR 2$ ;GO CHANGE DENSITY
2235 012272 032777 000040 166222 1$: BIT #40, @DS ;BRANCH IF SLAVE IS IN PE MODE
2236 012300 001013 BNE 4$ ;(PES = 1)
2237 012302 012777 000007 166200 2$: MOV #7, @C1 ;REWIND SLAVE
2238 012310 032777 000200 166204 20$: BIT #200, @DS ;WAIT FOR READY
2239 012316 001774 BEQ 20$
2240 012320 032777 020000 166174 3$: BIT #20000, @DS ;LOOP UNTIL REWIND IS COMPLETE
2241 012326 001374 BNE 3$ ;(PIP = 0)
2242 012330 053777 000774 166204 4$: BIS UDES, @TC ;LOAD UNIT DESCRIPTION
2243 012336 032777 000002 166156 BIT #2, @DS ;BRANCH IF NOT AT BOT
2244 012344 001407 BEQ 6$
2245 012346 012777 000025 166134 MOV #25, @C1 ;ERASE TO GET OFF BOT
2246 012354 032777 000200 166140 5$: BIT #200, @DS ;LOOP UNTIL DONE
2247 012362 001774 BEQ 5$
2248 012364 032777 000020 166130 6$: BIT #20, @DS ;WAIT FOR SETTLEDOWN BIT TO CLEAR
2249 012372 001374 BNE 6$
2250 012374 012777 000011 166106 MOV #11, @C1 ;DO A DRIVE CLEAR
2251 012402 032777 000004 166126 BIT #4, @DT ;BRANCH IF TE16
2252 012410 001404 BEQ 7$
2253 012412 005737 000602 TST SLVTYP ;BRANCH IF TU77 (SLVTYP = 1)
2254 012416 001012 BNE 100$
2255 012420 000403 BR 99$ ;TAKE ERROR EXIT
2256 012422 005737 000602 7$: TST SLVTYP ;BRANCH IF OPERATOR SELECTED TE16
2257 012426 001406 BEQ 100$
2258 012430 012704 015123 99$: MOV #MSG65, R4 ;TYPE 'INCORRECT SLAVE TYPE!!! PROGRAM ABORTED'
2259 012434 004737 013154 JSR PC, TTOUT
2260 012440 012716 002306 MOV #TEND, (SP) ;GO TO END OF PROGRAM
2261 012444 000207 100$: RTS PC ;RETURN TO CALLER
2262
2263 ;MAG TAPE INTERRUPT HANDLER*****
2264
2265 012446 000240 MTINT: NOP
2266 012450 013716 000666 MOV RTRN, (SP) ;SET RETURN TO (RTRN)
2267 012454 000002 RTI ;RETURN
2268
2269 ;TTY INTERRUPT HANDLER*****
2270
2271 012456 017746 166112 TTINT: MOV @TKB, -(SP) ;GET CHARACTER
2272 012462 042716 000200 BIC #200, (SP) ;CLEAR PARITY BIT
2273 012466 122716 000003 CMPB #3, (SP) ;BRANCH IF NOT CONTROL C
2274 012472 001006 BNE 1$
2275 012474 005737 001262 TST CHNFLG ;INHIBIT TC IF CHAIN MODE
2276 012500 001003 BNE 1$
2277 012502 000005 RESET

```

2278	012504	000137	000200		JMP	@#200	;RESTART PROGRAM
2279	012510	122716	000001	1\$:	CMPB	#1,(SP)	;BRANCH IF NOT #A
2280	012514	001017			BNE	2\$	
2281	012516	022737	000176	000570	CMP	#SWREG,SWR	;BRANCH IF HARDWARE SWR INVOKED
2282	012524	001016			BNE	3\$	
2283	012526	012737	177570	000570	MOV	#177570,SWR	;INVOKE HARDWARE SWR
2284	012534	004737	014116		JSR	PC,.SAVE	;SAVE REGISTERS ON THE STACK
2285	012540	012704	015037		MOV	#MSG63,R4	;TYPE 'HARDWARE SWR IN USE'
2286	012544	004737	013154		JSR	PC,TTOUT	
2287	012550	004737	014140		JSR	PC,.RESTORE	;RESTORE REGISTERS
2288	012554	022716	000007	2\$:	CMP	#7,(SP)	;BRANCH IF NOT #G
2289	012560	001006			BNE	4\$	

```

2290 012562 012737 000176 000570 3$: MOV #SWREG,SWR ;INVOKE SOFTWARE SWR
2291 012570 004737 014020 JSR PC,GTSWR ;GET SWITCHES
2292 012574 000414 BR 6$
2293 012576 122716 000023 4$: CMPB #23,(SP) ;SEE IF +S
2294 012602 001004 BNE 5$ ;BRANCH IF NOT
2295 012604 112737 000377 001176 MOVB #377,$CNTRLS ;SET XOFF FLAG
2296 012612 000405 BR 6$
2297 012614 122716 000021 5$: CMPB #21,(SP) ;SEE IF +Q
2298 012620 001002 BNE 6$ ;BRANCH IF NOT
2299 012622 105037 001176 CLRB $CNTRLS
2300 012626 005726 6$: TST (SP)+ ;POP CHARACTER OFF STACK
2301 012630 000002 RTI ;RETURN
2302

```



2303  
2304  
2305  
2306  
2307  
2308  
2309  
2310  
2311  
2312  
2313  
2314  
2315  
2316  
2317  
2318  
2319  
2320  
2321  
2322  
2323  
2324  
2325  
2326  
2327  
2328  
2329  
2330  
2331  
2332  
2333  
2334  
2335  
2336  
2337  
2338  
2339  
2340  
2341  
2342  
2343  
2344  
2345  
2346  
2347  
2348  
2349  
2350  
2351  
2352  
2353  
2354  
2355  
2356  
2357  
2358

012632 010146  
012634 011601  
012636 005037 000672  
012642 005000  
012644 004737 013112  
012650 122737 000003 000614  
012656 001003  
012660 000005  
012662 000137 000200  
012666 122737 000015 000614  
012674 001004  
012676 005737 000672  
012702 001471  
012704 000457  
012706 122737 000025 000614  
012714 001005  
012716 012704 014765  
012722 004737 013154  
012726 000742  
012730 122737 000177 000614  
012736 001012  
012740 000241  
012742 006000  
012744 006200  
012746 006200  
012750 012704 014767  
012754 004737 013154  
012760 005201  
012762 000730  
012764 122737 000060 000614  
012772 101402  
012774 000137 013072  
013000 122737 000070 000614  
013006 101002  
013010 000137 013072  
013014 005237 000672  
013020 006300  
013022 006300  
013024 006300

TTR:  
10\$:  
  
1\$:  
  
  
11\$:  
  
  
2\$:  
  
  
21\$:  
  
  
  
  
3\$:  
  
4\$:  
  
5\$:

```
*****  
; 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  
*****  
MOV R1, -(SP) ;SAVE CHAR COUNT  
MOV (SP), R1 ;RESET CHAR COUNT (FOR +U)  
CLR TEMP1 ;CLEAR FIRST CHARACTER FLAG  
CLR R0  
JSR PC, TTIN ;GO READ CHARACTER  
CMPB #3, TIB ;BRANCH IF NOT +C  
BNE 11$  
RESET  
JMP @#200 ;RESTART AT 200  
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  
CMPB #25, TIB ;BRANCH IF NOT CONTROL U  
BNE 21$  
MOV #MSG59, R4 ;TYPE <CR><LF>  
JSR PC, TTOUT  
BR 10$  
CMPB #177, TIB ;BRANCH IF NOT 'RUBOUT'  
BNE 3$ ;REMOVE LAST TYPED CHAR  
ROR R0  
ASR R0  
ASR R0  
MOV #MSG60, R4 ;TYPE '\'  
JSR PC, TTOUT  
INC R1 ;DECREMENT CHAR RECEIVED COUNT  
BR 1$ ;GET NEXT CHAR  
CMPB #60, TIB ;SEE IF CHAR IS LESS THAN 0  
BLOS 4$ ;IF NOT: BR  
JMP TIB ;ELSE GO TO ERROR  
CMPB #70, TIB ;SEE IF CHAR IS GREATER THAN 7  
BHI 5$ ;IF NOT: BR  
JMP TIB ;ELSE GO TO ERROR  
INC TEMP1 ;SET FIRST CHARACTER FLAG  
ASL R0  
ASL R0 ;SHIFT 3 LEFT  
ASL R0
```

2359	013026	042737	177770	000614		BIC	#177770,TIB	;STRIP ASCII
2360	013034	053700	000614			BIS	TIB,R0	;LOAD CHARACTER
2361	013040	005301				DEC	R1	;SEE IF DONE
2362	013042	001300				BNE	1\$	;IF NOT: BR
2363	013044	020002			6\$:	CMP	R0,R2	;SEE IF EXCEEDED MAXIMUM LIMIT
2364	013046	101402				BLOS	7\$	;IF NOT: BR
2365	013050	000137	013072			JMP	TINER	;ELSE GO TO ERROR
2366	013054	020300			7\$:	CMP	R3,R0	;SEE IF BELOW MINIMUM LIMIT
2367	013056	101402				BLOS	8\$	;IF NOT: BR
2368	013060	000137	013072			JMP	TINER	;ELSE GO TO ERROR
2369	013064	010015			8\$:	MOV	R0,(R5)	;LOAD VALUE
2370	013066	005726			9\$:	TST	(SP)+	;POP CHAR COUNT OFF STACK
2371	013070	000207				RTS	PC	;EXIT

```

2372
2373
2374
2375 013072 012704 014635
2376 013076 004737 013154
2377 013102 005726
2378 013104 162716 000020
2379 013110 000207
2380
2381
2382
2383 013112 005277 165454
2384 013116 105777 165450
2385 013122 100375
2386 013124 017737 165444 000614
2387 013132 042737 000200 000614
2388 013140 013737 000614 000612
2389 013146 004737 013254
2390 013152 000207
2391
2392
2393
2394 013154 112437 000612
2395 013160 122737 000043 000612
2396 013166 001472
2397 013170 122737 000045 000612
2398 013176 001403
2399 013200 004737 013254
2400 013204 000763
2401 013206 112737 000015 000612
2402 013214 004737 013254
2403 013220 012703 000004
2404 013224 005037 000612
2405 013230 004737 013254
2406 013234 005303
2407 013236 001372
2408 013240 112737 000012 000612
2409 013246 004737 013254
2410 013252 000740

;TTY ENTRY ERROR SUBROUTINE*****
TINER: MOV #MSG40,R4
JSR PC,TTOUT ;PRINT?
TST (SP); ;POP CHAR COUNT OFF STACK
SUB #20,(SP) ;RESET SP TO START OF VALUE ROUTINE
RTS PC ;REDO VALUE ENTRY

;TTY READ SUBROUTINE*****
TTIN: INC @TKS
1$: TSTB @TKS
BPL 1$
MOV @TKB,TIB
BIC #200,TIB
MOV TIB,TOB ;MOVE CHAR TO TTY OUPUT BFR
JSR PC,TOG ;ECHO CHARACTER
RTS PC

;TTY OUTPUT SUBROUTINE*****
TTOUT: MOVB (R4),TOB
CMPB #43,TOB
BEQ TEX
CMPB #45,TOB
BEQ 1$
JSR PC,TOG
BR TTOUT
1$: MOVB #15,TOB
JSR PC,TOG
MOV #4,R3
2$: CLR TOB
JSR PC,TOG
DEC R3
BNE 2$ ;DO FILLERS
MOVB #12,TOB
JSR PC,TOG
BR TTOUT

```



2411	013254	105777	165312		TOG:	TSTB	@TKS		;SEE IF INPUT AT KEYBOARD
2412	013260	100024				BPL	3\$		;IF SO, THEN
2413	013262	117737	165306	001177		MOVB	@TKB,%CNTRLS+1		;MOVE CHARACTER AND
2414	013270	142737	000200	001177		BICB	@200,%CNTRLS+1		;MASK OFF PARITY BIT.
2415	013276	122737	000023	001177		CMPB	@23,%CNTRLS+1		;SEE IF CHARACTER IS XOFF
2416	013304	001004				BNE	2\$		;IF XOFF, THEN
2417	013306	112737	000377	001176		MOVB	@377,%CNTRLS		;SET XOFF FLAG
2418	013314	000757				BR	TOG		
2419	013316	122737	000021	001177	2\$:	CMPB	@21,%CNTRLS+1		;SEE IF CHARACTER IS XON
2420	013324	001002				BNE	3\$		;IF SO THEN
2421	013326	105037	001176			CLRB	%CNTRLS		;CLEAR XOFF FLAG
2422	013332	105737	001176		3\$:	TSTB	%CNTRLS		;SEE IF IN XOFF MODE
2423	013336	100746				BMI	TOG		;IF NOT THEN
2424	013340	105777	165232			TSTB	@TPS		;CHECK IF PRINTER READY
2425	013344	100343				BPL	TOG		;
2426	013346	113777	000612	165224		MOVB	TOB,@TPB		
2427	013354	000207			TEX:	RTS	PC		;RETURN

```

2428
2429
2430
2431 013356 012737 000001 013612 OCTPE: MOV #1,OFL
2432 013364 000402 BR OCTPE1
2433 013366 005037 013612 OCTP: CLR OFL ;CLEAR FLAG FOR LEADING ZERO
2434 013372 010304 OCTPE1: MOV R3,R4 ;SEE IF NUMBER IS ZERO
2435 013374 001007 BNE OCTP0 ;IF NOT ZERO: BR
2436 013376 005737 013612 TST OFL ;SEE IF PRINT ALL 0
2437 013402 001004 BNE OCTP0 ;IF SO: BR
2438 013404 004737 013572 JSR PC,OCTPG1 ;ELSE PRINT ZERO
2439 013410 000137 013534 JMP OCTP3 ;SPACE AND EXIT
2440 013414 032704 100000 OCTP0: BIT #100000,R4 ;SEE IF MSD = 1
2441 013420 001406 BEQ OCTP1 ;IF NOT: BR
2442 013422 012704 000001 MOV #1,R4
2443 013426 004737 013550 JSR PC,OCTPG ;PRINT 1
2444 013432 000137 013444 JMP OCTP2
2445 013436 005004 OCTP1: CLR R4
2446 013440 004737 013550 JSR PC,OCTPG ;PRINT 0
2447 013444 010304 OCTP2: MOV R3,R4
2448 013446 006004 ROR R4
2449 013450 006004 ROR R4
2450 013452 006004 ROR R4 ;POSITION DIGIT
2451 013454 006004 ROR R4
2452 013456 000304 SWAB R4
2453 013460 004737 013550 JSR PC,OCTPG ;PRINT DIGIT 2
2454 013464 010304 MOV R3,R4
2455 013466 006004 ROR R4
2456 013470 000304 SWAB R4
2457 013472 004737 013550 JSR PC,OCTPG ;PRINT DIGIT 3
2458 013476 010304 MOV R3,R4
2459 013500 006104 ROL R4
2460 013502 006104 ROL R4
2461 013504 000304 SWAB R4
2462 013506 004737 013550 JSR PC,OCTPG ;PRINT DIGIT 4
2463 013512 010304 MOV R3,R4
2464 013514 006004 ROR R4
2465 013516 006004 ROR R4
2466 013520 006004 ROR R4
2467 013522 004737 013550 JSR PC,OCTPG
2468 013526 010304 MOV R3,R4
2469 013530 004737 013550 JSR PC,OCTPG ;PRINT DIGIT 5

```

2470	013534	012737	000240	000612	OCTP3:	MOV	#240,TOB	
2471	013542	004737	013254			JSR	PC,TOG	;PRINT SPACE
2472	013546	000207				RTS	PC	;EXIT
2473	013550	042704	177770		OCTPG:	BIC	#177770,R4	
2474	013554	001004				BNE	OCTPGO	
2475	013556	005737	013612			TST	OFL	
2476	013562	001001				BNE	OCTPGO	
2477	013564	000207				RTS	PC	
2478								
2479	013566	005237	013612		OCTPGO:	INC	OFL	
2480	013572	052704	000260		OCTPG1:	BIS	#260,R4	
2481	013576	010437	000612			MOV	R4,TOB	
2482	013602	004737	013254			JSR	PC,TOG	
2483	013606	010304				MOV	R3,R4	
2484	013610	000207				RTS	PC	
2485	013612	000000			OFL:	0		;FIRST CHAR FLAG
2486								



```

2487
2488
2489
2490 013614 012704 000010      DOUT:  MOV    #10,R4      ;SET NUMBER TO PRINT
2491 013620 110337 000612      MOVB   R3,TOB
2492 013624 105777 164746      1$:   TSTB   @TPS
2493 013630 100375                BPL    1$
2494 013632 132737 000200 000612  BITB   #200,TOB
2495 013640 001404                BEQ    2$
2496 013642 012777 000061 164730  MOV    #061,@TPB
2497 013650 000403                BR     3$
2498 013652 012777 000060 164720  2$:   MOV    #060,@TPB
2499 013660 006337 000612      3$:   ASL    TOB
2500 013664 005304                DEC    R4
2501 013666 001356                BNE   1$
2502 013670 000207                RTS    PC
2503
2504 013672 013703 000676      DOUTD: MOV    TEMP3,R3
2505 013676 000303                SWAB  R3
2506 013700 004737 013614      JSR    PC,DOUT
2507 013704 013703 000676      MOV    TEMP3,R3
2508 013710 004737 013614      JSR    PC,DOUT
2509 013714 000207                RTS    PC
2510
2511
2512
2513 013716 010304      SNPT:  MOV    R3,R4
2514 013720 000304                SWAB  R4
2515 013722 006004                ROR   R4
2516 013724 006004                ROR   R4
2517 013726 006004                ROR   R4
2518 013730 006004                ROR   R4      ;GET FIRST DIGIT
2519 013732 004737 013774      JSR    PC,SNPG  ;PRINT
2520 013736 010304                MOV   R3,R4
2521 013740 000304                SWAB  R4      ;GET SECOND DIGIT
2522 013742 004737 013774      JSR    PC,SNPG  ;PRINT
2523 013746 010304                MOV   R3,R4
2524 013750 006004                ROR   R4
2525 013752 006004                ROR   R4
2526 013754 006004                ROR   R4
2527 013756 006004                ROR   R4
2528 013760 004737 013774      JSR    PC,SNPG  ;PRINT THIRD DIGIT
2529 013764 010304                MOV   R3,R4
2530 013766 004737 013774      JSR    PC,SNPG  ;PRINT FOURTH DIGIT
2531 013772 000207                RTS    PC      ;EXIT
2532 013774 012737 000260 000612  SNPG:  MOV    #260,TOB  ;SET BASE = 0
2533 014002 042704 177760      BIC   #177760,R4 ;MASK DIGIT
2534 014006 050437 000612      BIS   R4,TOB    ;SET ASCII
2535 014012 004737 013254      JSR   PC,TOG    ;TYPE DIGIT
2536 014016 000207                RTS    PC      ;RETURN
  
```

```

2537
2538           ;THIS ROUTINE GETS THE NEW VALUE FOR THE SOFTWARE SWITCH REG
2539
2540 014020 022737 000176 000570 GTSWR:  CMP    #SWREG,SWR    ;BRANCH IF SOFTWARE SWR NOT
2541 014026 001032                BNE    1$          ;INVOKED
2542 014030 004737 014116                JSR    PC,.SAVE    ;SAVE REGISTERS ON THE STACK
2543 014034 012704 016426                MOV    #MSWR,R4    ;TYPE 'SWR = '
2544 014040 004737 013154                JSR    PC,TTOUT
2545 014044 017703 164520                MOV    @SWR,R3     ;GET CURRENT SETTING
2546 014050 004737 013356                JSR    PC,OCTPE    ;AND TYPE THEM
2547 014054 012704 016436                MOV    #MNEW,R4    ;TYPE 'NEW = '
2548 014060 004737 013154                JSR    PC,TTOUT
2549 014064 013705 000570                MOV    SWR,R5     ;TTR ROUTINE RETURN NEW VALUE TO (R5)
2550 014070 012701 000007                MOV    #7,R1      ;LIMIT RESPONSE TO 7 CHARS
2551 014074 012702 177777                MOV    #177777,R2 ;BETWEEN 0 AND 177777
2552 014100 012703 000000                MOV    #0,R3
2553 014104 004737 012632                JSR    PC,TTR      ;GET RESPONSE
2554 014110 004737 014140                JSR    PC,.RESTORE ;RESTORE REGISTERS
2555 014114 000207                1$:   RTS    PC    ;RETURN TO CALLER
2556
2557           ;;ROUTINE TO SAVE REGISTERS ON THE STACK
2558 014116 010546                .SAVE: MOV    #5,-(SP) ;;R5 IS SAVED AT 12(SP)
2559 014120 010446                MOV    #4,-(SP) ;;R4 IS SAVED AT 10(SP)
2560 014122 010346                MOV    #3,-(SP) ;;R3 IS SAVED AT 6(SP)
2561 014124 010246                MOV    #2,-(SP) ;;R2 IS SAVED AT 4(SP)
2562 014126 010146                MOV    #1,-(SP) ;;R1 IS SAVED AT 2(SP)
2563 014130 010046                MOV    #0,-(SP) ;;R0 IS SAVED AT (SP)
2564 014132 016646 000014                MOV    14(SP),-(SP) ;;PUSH RETURN PC ON THE STACK
2565 014136 000207                RTS    PC          ;;RETURN TO CALLER
2566
2567           ;;ROUTINE TO RESTORE REGISTERS SAVED ON THE STACK
2568 014140 012666 000014                .RESTORE:MOV (SP)+,14(SP) ;;STORE RETURN PC ON STACK
2569 014144 012600                MOV    (SP)+,#0
2570 014146 012601                MOV    (SP)+,#1
2571 014150 012602                MOV    (SP)+,#2
2572 014152 012603                MOV    (SP)+,#3
2573 014154 012604                MOV    (SP)+,#4
2574 014156 012605                MOV    (SP)+,#5
2575 014160 000207                RTS    PC          ;;RETURN
2576

```





2633	014577	045	041524	041040	MSG18:	.ASCII	/MTC BITS 12-0 #/
2634	014604	052111	020123	031061			
2635	014612	030055	021440				
2636	014616	043045	020103	044502	MSG19:	.ASCII	/MFC BITS 15-0 #/
2637	014624	051524	030440	026465			
2638	014632	020060	043				
2639	014635	040	020077	043	MSG40:	.ASCII	/ ? #/
2640	014641	045	042445	042116	MSG41:	.ASCII	/MEND OF PASS #/
2641	014646	047440	020106	040520			
2642	014654	051523	021440				
2643	014660	051045	043505	051511	MSG44:	.ASCII	/MREGISTER START: #/
2644	014666	042524	020122	052123			
2645	014674	051101	035124	021440			
2646	014702	053045	041505	047524	MSG45:	.ASCII	/MVECTOR ADDRESS: #/
2647	014710	020122	042101	051104			
2648	014716	051505	035123	021440			
2649	014724	052045	030115	020063	MSG57:	.ASCII	/MTM03 DRIVE: #/
2650	014732	051104	053111	035105			
2651	014740	021440					
2652	014742	052045	030505	027466	MSG58:	.ASCII	'MTE16/TU77 SLAVE: #';..B
2653	014750	052524	033467	051440			
2654	014756	040514	042526	020072			
2655	014764	043					
2656	014765	045	043		MSG59:	.ASCII	/M#/
2657	014767	134	043		MSG60:	.ASCII	/\#/
2658	014771	045	042522	047515	MSG62:	.ASCII	/MREMOVE TMDP FROM SLAVE TO BE TESTED#/
2659	014776	042526	052040	042115			
2660	015004	020120	051106	046517			
2661	015012	051440	040514	042526			
2662	015020	052040	020117	042502			
2663	015026	052040	051505	042524			
2664	015034	022504	043				
2665	015037	045	040510	042122	MSG63:	.ASCII	/MHARDWARE SWR IN USE#/
2666	015044	040527	042522	051440			
2667	015052	051127	044440	020116			
2668	015060	051525	022505	043			
2669	015065	045	046123	053101	MSG64:	.ASCII	/MSLAVE TYPE (0-TE16,1-TU77): #/
2670	015072	020105	054524	042520			
2671	015100	024040	036460	042524			
2672	015106	033061	030454	052075			
2673	015114	033525	024467	020072			
2674	015122	043					
2675	015123	045	044445	041516	MSG65:	.ASCII	/MINCORRECT SLAVE TYPE!!! PROGRAM ABORTED#/
2676	015130	051117	042522	052103			
2677	015136	051440	040514	042526			
2678	015144	052040	050131	020505			
2679	015152	020441	050040	047522			
2680	015160	051107	046501	040440			
2681	015166	047502	052122	042105			
2682	015174	021445					

```

2683                                     ;TEST HEADER*****
2684
2685 015176 022445 047514 044507 MSLT1: .ASCII /LOGIC TEST 1: WRAP 3,NRZ,NORMAL,ODD#/
2686 015204 020103 042524 052123
2687 015212 030440 020072 051127
2688 015220 050101 031440 047054
2689 015226 055122 047054 051117
2690 015234 040515 026114 042117
2691 015242 021504
2692 015244 022445 047514 044507 MSLT2: .ASCII /LOGIC TEST 2: WRAP 3,PE,NORMAL,ODD#/
2693 015252 020103 042524 052123
2694 015260 031040 020072 051127
2695 015266 050101 031440 050054
2696 015274 026105 047516 046522
2697 015302 046101 047454 042104
2698 015310 043
2699 015311 045 046045 043517 MSLT3: .ASCII /LOGIC TEST 3: WRAP 2,NRZ,NORMAL,ODD#/
2700 015316 041511 052040 051505
2701 015324 020124 035063 053440
2702 015332 040522 020120 026062
2703 015340 051116 026132 047516
2704 015346 046522 046101 047454
2705 015354 042104 043
2706 015357 045 046045 043517 MSLT4: .ASCII /LOGIC TEST 4: WRAP 2,PE,NORMAL,ODD#/
2707 015364 041511 052040 051505
2708 015372 020124 035064 053440
2709 015400 040522 020120 026062
2710 015406 042520 047054 051117
2711 015414 040515 026114 042117
2712 015422 021504
2713 015424 022445 047514 044507 MSLT5: .ASCII /LOGIC TEST 5: WRAP 1,NRZ,NORMAL,ODD#/
2714 015432 020103 042524 052123
2715 015440 032440 020072 051127
2716 015446 050101 030440 047054
2717 015454 055122 047054 051117
2718 015462 040515 026114 042117
2719 015470 021504
2720 015472 022445 047514 044507 MSLT6: .ASCII /LOGIC TEST 6: WRAP 1,PE,NORMAL,ODD#/
2721 015500 020103 042524 052123
2722 015506 033040 020072 051127
2723 015514 050101 030440 050054
2724 015522 026105 047516 046522
2725 015530 046101 047454 042104
2726 015536 043
2727 015537 045 046045 043517 MSLT7: .ASCII /LOGIC TEST 7: WRAP 0,NRZ,NORMAL,ODD#/
2728 015544 041511 052040 051505
2729 015552 020124 035067 053440
2730 015560 040522 020120 026060
2731 015566 051116 026132 047516
2732 015574 046522 046101 047454
2733 015602 042104 043
2734 015605 045 046045 043517 MSLT10: .ASCII /LOGIC TEST 10: WRAP 0,PE,NORMAL,ODD#/
2735 015612 041511 052040 051505
2736 015620 020124 030061 020072
2737 015626 051127 050101 030040
2738 015634 050054 026105 047516

```



2739	015642	046522	046101	047454	
2740	015650	042104	043		
2741	015653	045	046045	043517	MSLT11: .ASCII /LOGIC TEST 11: CORE DUMP WRITE (M8906)*/
2742	015660	041511	052040	051505	
2743	015666	020124	030461	020072	
2744	015674	047503	042522	042040	
2745	015702	046525	020120	051127	
2746	015710	052111	020105	046450	
2747	015716	034470	033060	021451	
2748	015724	022445	047514	044507	MSLT12: .ASCII /LOGIC TEST 12: CORE DUMP READ (M8906)*/
2749	015732	020103	042524	052123	
2750	015740	030440	035062	041440	
2751	015746	051117	020105	052504	
2752	015754	050115	051040	040505	
2753	015762	020104	046450	034470	
2754	015770	033060	021451		
2755	015774	022445	047514	044507	MSLT13: .ASCII /LOGIC TEST 13: EVEN PARITY WRITE (M8933 M8934)*/
2756	016002	020103	042524	052123	
2757	016010	030440	035063	042440	
2758	016016	042526	020116	040520	
2759	016024	044522	054524	053440	
2760	016032	044522	042524	024040	
2761	016040	034115	031471	020063	
2762	016046	034115	031471	024464	
2763	016054	043			
2764	016055	045	046045	043517	MSLT14: .ASCII /LOGIC TEST 14: EVEN PARITY READ(M8933 M8934)*/
2765	016062	041511	052040	051505	
2766	016070	020124	032061	020072	
2767	016076	053105	047105	050040	
2768	016104	051101	052111	020131	
2769	016112	042522	042101	046450	
2770	016120	034470	031463	046440	
2771	016126	034470	032063	021451	
2772	016134	022445	047514	044507	MSLT15: .ASCII /LOGIC TEST 15: READ REVERSE(M8906)*/
2773	016142	020103	042524	052123	
2774	016150	030440	035065	051040	
2775	016156	040505	020104	042522	
2776	016164	042526	051522	024105	
2777	016172	034115	030071	024466	
2778	016200	043			
2779	016201	045	046045	043517	MSLT16: .ASCII /LOGIC TEST 16: CRC CORRECTION SINGLE TRACK,ALL FRAMES*/
2780	016206	041511	052040	051505	
2781	016214	020124	033061	020072	
2782	016222	051103	020103	047503	
2783	016230	051122	041505	044524	
2784	016236	047117	051440	047111	
2785	016244	046107	020105	051124	
2786	016252	041501	026113	046101	
2787	016260	020114	051106	046501	
2788	016266	051505	043		
2789	016271	045	046045	043517	MSLT17: .ASCII /LOGIC TEST 17: CRC CORRECTION MULTIPLE BAD TRACKS*/
2790	016276	041511	052040	051505	
2791	016304	020124	033461	020072	
2792	016312	051103	020103	047503	
2793	016320	051122	041505	044524	
2794	016326	047117	046440	046125	



2795	016334	044524	046120	020105
2796	016342	040502	020104	051124
2797	016350	041501	051513	043
2798	016355	045	046045	043517
2799	016362	041511	052040	051505
2800	016370	020124	030062	020072
2801	016376	042522	042101	051040
2802	016404	053105	051105	042523
2803	016412	047054	055122	053454
2804	016420	040522	020120	021463

MSLT20: .ASCII /LOGIC TEST 20: READ REVERSE,NRZ,WRAP 30/

```

2805
2806 ;TAG MESSAGE
2807
2808 016426 051445 051127 036440 $MSWR: .ASCII /#SWR - #/
2809 016434 021440
2810 016436 047040 053505 036440 $MNEW: .ASCII / NEW = #/
2811 016444 021440
2812
2813 .EVEN
2814 ;WRITE BUFFER
2815
2816 016446 000100 WDATA:
2817 016446 177777 -1
2818 016450 177777 -1
2819 016452 177777 -1
2820 016454 177777 -1
2821 016456 177777 -1
2822 016460 177777 -1
2823 016462 177777 -1
2824 016464 177777 -1
2825 016466 177777 -1
2826 016470 177777 -1
2827 016472 177777 -1
2828 016474 177777 -1
2829 016476 177777 -1
2830 016500 177777 -1
2831 016502 177777 -1
2832 016504 177777 -1
2833 016506 177777 -1
2834 016510 177777 -1
2835 016512 177777 -1
2836 016514 177777 -1
2837 016516 177777 -1
2838 016520 177777 -1
2839 016522 177777 -1
2840 016524 177777 -1
2841 016526 177777 -1
2842 016530 177777 -1
2843 016532 177777 -1
2844 016534 177777 -1
2845 016536 177777 -1
2846 016540 177777 -1
2847 016542 177777 -1
2848 016544 177777 -1
2849 016546 177777 -1
2850 016550 177777 -1
2851 016552 177777 -1
2852 016554 177777 -1
2853 016556 177777 -1
2854 016560 177777 -1
2855 016562 177777 -1
2856 016564 177777 -1
2857 016566 177777 -1
2858 016570 177777 -1
2859 016572 177777 -1
2860 016574 177777 -1

```

2861	016576	177777	-1
2862	016600	177777	-1
2863	016602	177777	-1
2864	016604	177777	-1
2865	016606	177777	-1
2866	016610	177777	-1
2867	016612	177777	-1
2868	016614	177777	-1
2869	016616	177777	-1
2870	016620	177777	-1
2871	016622	177777	-1
2872	016624	177777	-1
2873	016626	177777	-1
2874	016630	177777	-1
2875	016632	177777	-1
2876	016634	177777	-1
2877	016636	177777	-1
2878	016640	177777	-1
2879	016642	177777	-1
2880	016644	177777	-1

2881			
2882			
2883			:READ BUFFER
2884			

2885	016646	000100	RDATA:	0
2886	016646	000000		0
2887	016650	000000		0
2888	016652	000000		0
2889	016654	000000		0
2890	016656	000000		0
2891	016660	000000		0
2892	016662	000000		0
2893	016664	000000		0
2894	016666	000000		0
2895	016670	000000		0
2896	016672	000000		0
2897	016674	000000		0
2898	016676	000000		0
2899	016700	000000		0
2900	016702	000000		0
2901	016704	000000		0
2902	016706	000000		0
2903	016710	000000		0
2904	016712	000000		0
2905	016714	000000		0
2906	016716	000000		0
2907	016720	000000		0
2908	016722	000000		0
2909	016724	000000		0
2910	016726	000000		0
2911	016730	000000		0
2912	016732	000000		0
2913	016734	000000		0
2914	016736	000000		0
2915	016740	000000		0
2916	016742	000000		0



2917	016744	000000	0
2918	016746	000000	0
2919	016750	000000	0
2920	016752	000000	0
2921	016754	000000	0
2922	016756	000000	0
2923	016760	000000	0
2924	016762	000000	0
2925	016764	000000	0
2926	016766	000000	0
2927	016770	000000	0
2928	016772	000000	0
2929	016774	000000	0
2930	016776	000000	0
2931	017000	000000	0
2932	017002	000000	0
2933	017004	000000	0
2934	017006	000000	0
2935	017010	000000	0
2936	017012	000000	0
2937	017014	000000	0
2938	017016	000000	0
2939	017020	000000	0
2940	017022	000000	0
2941	017024	000000	0
2942	017026	000000	0
2943	017030	000000	0
2944	017032	000000	0
2945	017034	000000	0
2946	017036	000000	0
2947	017040	000000	0
2948	017042	000000	0
2949	017044	000000	0

;WRAP AROUND MESSAGES\*\*\*\*\*

2952									
2953	017046	051445	052105	050125	WMSG2:	.ASCII	/#SETUP ERROR#/#/		
2954	017054	042440	051122	051117					
2955	017062	021445							
2956	017064	050045	052101	047122	WMSG3:	.ASCII	/#PATRN NUMBER = #/		
2957	017072	047040	046525	042502					
2958	017100	020122	020075	043					
2959	017105	045	047516	026516	WMSG4:	.ASCII	/#NON-EXISTANT DRIVE#/#/		
2960	017112	054105	051511	040524					
2961	017120	052116	042040	044522					
2962	017126	042526	021445						
2963	017132	041445	030523	021440	WMSG6:	.ASCII	/#CS1 #/		
2964	017140	053445	020103	043	WMSG6A:	.ASCII	/#WC #/		
2965	017145	045	040502	021440	WMSG6B:	.ASCII	/#BA #/		
2966	017152	043045	020103	043	WMSG6C:	.ASCII	/#FC #/		
2967	017157	045	051503	020062	WMSG6D:	.ASCII	/#CS2 #/		
2968	017164	043							
2969	017165	045	051504	021440	WMSG6E:	.ASCII	/#DS #/		
2970	017172	042445	020122	043	WMSG6F:	.ASCII	/#ER #/		
2971	017177	045	051501	021440	WMSG6G:	.ASCII	/#AS #/		
2972	017204	041445	020103	043	WMSG6H:	.ASCII	/#CC #/		

2973	017211	045	041104	021440	WMSG6I:	.ASCII	/DB #/
2974	017216	046445	020122	043	WMSG6J:	.ASCII	/MR #/
2975	017223	045	052104	021440	WMSG6K:	.ASCII	/DT #/
2976	017230	052045	020103	043	WMSG6L:	.ASCII	/TC #/
2977	017235	045	047123	021440	WMSG6M:	.ASCII	/SN #/
2978	017242	041045	042101	042040	WMSG16:	.ASCII	/BAD DATA#/
2979	017250	052101	021501				
2980	017254	043445	020072	043	WMSG17:	.ASCII	/G: #/
2981	017261	045	035102	021440	WMSG20:	.ASCII	/B: #/
2982	017266	041445	035116	021440	WMSG21:	.ASCII	/CN: #/
2983	017274	041045	042101	051440	WMSG23:	.ASCII	/BAD STATUS#/
2984	017302	040524	052524	021523			
2985	017310	047045	020117	047111	WMSG24:	.ASCII	/NO INTERRUPT#/
2986	017316	042524	051122	050125			
2987	017324	021524					
2988	017326	047045	020117	046103	WMSG25:	.ASCII	/NO CLOCK UP#/
2989	017334	041517	020113	050125			
2990	017342	043					
2991	017343	045	047516	041440	WMSG26:	.ASCII	/NO CLOCK DOWN#/
2992	017350	047514	045503	042040			
2993	017356	053517	021516				
2994	017362	042045	052101	020101	WMSG27:	.ASCII	/DATA PAT: #/
2995	017370	040520	035124	043			
2996	017375	045	040502	020104	WMSG28:	.ASCII	/BAD PREAMBLE#/
2997	017402	051120	040505	041115			
2998	017410	042514	043				
2999	017413	045	040502	020104	WMSG29:	.ASCII	/BAD POSTAMBLE#/
3000	017420	047520	052123	046501			
3001	017426	046102	021505				
3002	017432	042445	051117	041440	WMSG31:	.ASCII	/EOR CLEAR DID NOT CLEAR GO#/
3003	017440	042514	051101	042040			
3004	017446	042111	047040	052117			
3005	017454	041440	042514	051101			
3006	017462	043440	022517	043			
3007	017467	040	040520	051124	WMSG32:	.ASCII	/ PATRN #/
3008	017474	020116	043				
3009							
3010		017500					
3011	017500	000000			PRE:	.EVEN	0
3012	017502	000000					0
3013	017504	000000					0
3014	017506	000000					0
3015	017510	000000					0
3016	017512	000000					0
3017	017514	000000					0
3018	017516	000000					0
3019	017520	000000					0
3020	017522	000000					0
3021	017524	000000					0
3022	017526	000000					0
3023	017530	000000					0
3024	017532	000000					0
3025	017534	000000					0
3026	017536	000000					0
3027	017540	000000					0
3028	017542	000000					0

3029	017544	000000	0
3030	017546	000000	0
3031	017550	000000	0
3032	017552	000000	0
3033	017554	000000	0
3034	017556	000000	0
3035	017560	000000	0
3036	017562	000000	0
3037	017564	000000	0
3038	017566	000000	0
3039	017570	000000	0
3040	017572	000000	0
3041	017574	000000	0
3042	017576	000000	0
3043	017600	000000	0
3044	017602	000000	0
3045	017604	000000	0
3046	017606	000000	0
3047	017610	000000	0
3048	017612	000000	0
3049	017614	000000	0
3050	017616	000000	0
3051	017620	000000	0
3052	017622	000000	0
3053	017624	000000	0
3054	017626	000000	0
3055	017630	000000	0
3056	017632	000000	0
3057	017634	000000	0
3058	017636	000000	0
3059	017640	000000	0
3060	017642	000000	0
3061	017644	000000	0
3062	017646	000000	0
3063	017650	000000	0
3064	017652	000000	0
3065	017654	000000	0
3066	017656	000000	0
3067	017660	000000	0
3068	017662	000000	0
3069	017664	000000	0
3070	017666	000000	0
3071	017670	000000	0
3072	017672	000000	0
3073	017674	000000	0
3074	017676	000000	0
3075	017700	000000	0
3076	017702	000000	0
3077	017704	000000	0
3078	017706	000000	0
3079	017710	000000	0
3080	017712	000000	0
3081	017714	000000	0
3082	017716	000000	0
3083	017720	000000	0
3084	017722	000000	0

POST:



3085	017724	000000	0
3086	017726	000000	0
3087	017730	000000	0
3088	017732	000000	0
3089	017734	000000	0
3090	017736	000000	0
3091	017740	000000	0
3092	017742	000000	0
3093	017744	000000	WBUFF: 0
3094		020356	. = . + 410
3095	020356	000000	RBUFF: 0
3096			
3097		000001	.END















ST2	001732	672	966#											
SWR	000570	714#	874	878*	886*	977	1011	1017	1047	1400	1524	1528	1555	1559
		1680	1684	1700	1704	1751	1759	1840	1888	1895	1940	1981	2018	2082
		2165	2173	2183	2187	2198	2209	2281	2283*	2290*	2540	2545	2519	
SWREG	000176	663#	878	886	2281	2290	2540							
TADX	001172	863#												
TC	000542	697#	1001*	1546*	1578	2227*	2242*							
TEMP1	000672	753#	2322*	2331	2355*									
TEMP2	000674	754#	2151*	2177	2191*									
TEMP3	000676	755#	2504	2507										
TEND	002306	863	1022	1028#	2260									
TENDX	002404	1046	1048	1050#										
TEX	013354	2396	2427#											
TIB	000614	730#	2325	2329	2334	2339	2349	2352	2359*	2360	2386*	2387*	2388	
TINER	013072	2351	2354	2365	2368	2375#								
TKB	000574	716#	2271	2386	2413									
TKS	000572	715#	974*	2383*	2384	2411								
TLAST	001174	864#	1021											
TOB	000612	729#	1989*	1991*	1994*	2039*	2047*	2107*	2115*	2388*	2394*	2395	2397	2401*
		2404*	2408*	2426	2470*	2481*	2491*	2494	2499*	2532*	2534*			
TOG	013254	1990	1995	2040	2048	2108	2116	2389	2399	2402	2405	2409	2411#	2418
		2423	2425	2471	2482	2535								
TPB	000600	718#	2426*	2496*	2498*									
TPS	000576	717#	2424	2492										
TREF	000732	769#												
TR00	000624	734#												
TR01	000626	735#												
TR02	000630	736#												
TR03	000632	737#												
TR04	000634	738#												
TR05	000636	739#												
TR06	000640	740#												
TR07	000642	741#												
TR10	000644	742#												
TR11	000646	743#												
TR12	000650	744#												
TR13	000652	745#												
TR14	000654	746#												
TR15	000656	747#												
TSCD	001756	900	974#	1019	1051									
TSCDA	002152	981	1004#											
TSCD0	002160	1005#	1016											
TSCD1	002166	1006#	1027											
TSCD2	002214	1011#	1072	1100	1128	1157	1185	1205	1310	1363	1383			
TSCD3	002232	1012	1015#											
TSTTBL	001066	829#	1004	1025										
TTIN	013112	2324	2383#											
TTINT	012456	657	2271#											
TTOUT	013154	894	897	903	912	937	946	956	1031	1403	1527	1558	1683	1764
		1766	1769	1771	1775	1845	1849	1851	1857	1861	1865	1945	1949	1951
		1958	1962	1969	2023	2029	2032	2036	2044	2087	2089	2091	2096	2100
		2112	2170	2172	2259	2286	2337	2346	2376	2394#	2400	2410	2544	2548
		910	919	944	953	963	1410	2320#	2553					
TTR	012632	773#												
T24FL	000742	786#	1063*	1082*	1092*	1110*	1120*	1139*	1149*	1167*	1176*	1194*	1214*	1224*
UDES	000774	1234*	1269*	1332*	1375*	1485	1517	1543	1546	1591	1596	1611	1642	1657







\$CATCH	557#	642
\$CHAIN	557#	881
\$CHNMO	557#	980
\$RESTO	557#	2567
\$SAVE	557#	2557
.\$ACT1	557#	643
.\$EOP	557#	1037

. ABS. 020360 000

ERRORS DETECTED: 0

CZTEBC,CZTEBC/SOL/CRF=CZTEAE.SML/ML,CZTEBC.P11  
RUN-TIME: 3 5 .8 SECONDS  
RUN-TIME RATIO: 13/9=1.3  
CORE USED: 11K (22 PAGES)