

TM03/TE16

CONTROL LOGIC TEST 2
CZTEBB0

AH-A795B-MC
COPYRIGHT © 77-78
FICHE 1 OF 1

JAN 1979
digital
MADE IN USA

This microfiche card contains a grid of frames. The leftmost column consists of 16 frames, each containing a small table with multiple columns and rows of data. The remaining frames in the grid are mostly blank or contain very faint, illegible text. The card is oriented vertically on the page.

.REM 2

IDENTIFICATION

PRODUCT CODE: AC-A794B-MC
PRODUCT NAME: CZTEBBO TMO3-TE16/TU77 CONTROL LOGIC TEST PART II
DATE CREATED: 15 NOV 78
MAINTAINER: DIAGNOSTIC ENGINEERING
AUTHOR: J. G. ADAMS

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

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

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

COPYRIGHT (©) 1977, 1978 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 TMO3 FORMATTER. EACH TEST WILL ATTEMPT TO ISOLATE FAILURES TO THE MODULE LEVEL AND PROVIDE PRINTOUT INFORMATION WHICH WILL IDENTIFY THE FAILING MODULE. THE LEVEL OF FAULT ISOLATION IS POSSIBLE BECAUSE OF TMO3 THE STRUCTURE AND ITS MAINTAINENCE MODES.

2. REQUIREMENTS (HARDWARE)

- A. ANY PDP-11 PROCESSOR
- B. 8K OF CORE
- C. CONSOLE TTY
- D. TMO3 MAGTAPE CONTROLLER
- E. MASSBUS CONTROLLER (RH)
- 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(B) AND 210(B).

A. 200(B): STARTING AT THIS ADDRESS WILL CAUSE A PROGRAM IDENTIFICATION HEADER TO BE PRINTED BEFORE TESTING IS BEGUN.

B. 210(B): 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 SFC 5. CONSOLE SWITCH SETTINGS
** TYPE ^C TO RESTART PROGRAM (@200)

4.1 AUTOMATIC MODE OPERATION

1: 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(SLVTP:) 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 BO
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)	[SLVTYP:]
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=DO CONTINUOUS CYCLE
0=HALT AT END OF PASS
SW11: 1=INHIBIT ITERATIONS
0=ITERATE EACH TEST ITS ASSIGNED AMOUNT
SW10: 1=HALT AT END OF CURRENT TEST
0=CONTINUE TO NEXT TEST
SW8: 1=INHIBIT WRAP AROUND DATA CHECK
0=DO DATA CHECKS
SW7: 1=INHIBIT WRAP AROUND STATUS CHECK
0=DO STATUS CHECK
SW6: 1-SELECTABLE WRAP DATA PATTERN (IN SINGLE TEST)
0-AUTO PATTERN
SW5-0: SELECT INDIVIDUAL TEST ** 00=DO ALL TESTS

6. ERROR PRINTOUTS

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

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

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

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

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

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

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

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

7. OPERATION

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

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

ALL CONSOLE SWITCHES ARE DYNAMIC AND MAY BE CHANGED AT ANY TIME. THE NORMAL OPERATING SEQUENCE IS ALL SWITCHES DOWN (0). THE TEST WILL TAKE APPROXIMATELY 3 MINUTES TO RUN; HOWEVER, IF ITERATIONS ARE INHIBITED (SW11=1) THE TEST WILL RUN IN ABOUT 30 SECONDS. THE END OF PASS IS NOTED BY A PRINTOUT STAT.ING 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 4: WRP2, PE, NORMAL, ODD

THE TEST IS EXACTLY LIKE TEST 43 EXCEPT THAT PE WRT CLK ENBL L MUST BE ASSERTED BY M8932 TO ENABLE WR TO STROBE. THIS DOES NOT HAPPEN UNTIL THE PE WRITE CONTROL CIRCUIT HAS CLOCKED THROUGH THE PREAMBLE.

CIRCUITS

PRINT REFERENCES

(IN ADDITION TO TEST 44)
PE WRITE CONTROL

TCPE3

LOGIC TEST 5: WRP1, NRZ, NORMAL, ODD

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

LIKELY FAULT LOCATIONS: M8933, M8934 (CRC GENERATOR)

CIRCUITS

PRINT REFERENCES

WRITE BUFFER
CRC GENERATOR

TCCM2
CNRZ2

LOGIC TEST 6: WRAP1, PE, NORMAL, ODD

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

LIKELY FAULT LOCATIONS: M8932 (WRITE CONTROL STATES), M8933

CIRCUITS

PRINT REFERENCE

WRITE BUFFER
WRITE CONTROL

TCCM2
TCPE3

LOGIC TEST 7: WRAP0, NORMAL, ODD

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

LIKELY FAULT LOCATIONS: 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: WRP0, 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: WAM0,

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 %

.LIST BIN,LOC,SEQ

546
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

```
.TITLE CZTEBB0 TMO3-TE16/TU77 CLT II  
:CONTROL LOGIC TEST PART II  
:AC-A794B-MC  
:FEB 77  
:J.G. ADAMS  
:REVISED MAY 1978 BY J.G.ADAMS ;++B CHANGED MODULE TYPEOUTS TO  
;++B REFLECT TMO3 REGISTER SET  
;++B ADDED TU77 TEST CAPABILITY  
  
.MCALL .SACT11,.$EOF,$CATCH,$SAVE,$RESTORE,$CHAIN,$CHMODE  
.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
```



```
628 ;REGISTER EQUIVS*****
629
630 000000 R0=%0
631 000001 R1=%1
632 000002 R2=%2
633 000003 R3=%3
634 000004 R4=%4
635 000005 R5=%5
636 000006 SP=%6
637 000007 PC=%7
638
639
640
641 ;ACT11 HOOK *****
642 000764 $SVPC= ;SAVE CURRENT LOCATION CTR
643 000046 .=46
644 000046 002350 .WORD SENDAD ;SET LOCATION 46
645 000052 000052 .=52
646 000052 000000 .WORD 0 ;SET LOCATION 52 - 0
647 000764 .-$SVPC ;RESTORE LOCATION CTR
648
649 ;TTY INTERRUPT VECTOR*****
650
651 000060 .-60
652 000060 012454 .WORD TTINT ;TTY INTERRUPT HANDLER ADDRESS
653 000062 0003-0 .WORD 340 ;PRIORITY LEVEL 7
654
655 ;SOFTWARE SWITCH REGISTER*****
656 ;USED IF HARDWARE SWR = 177777, OR NOT AVAIL.
657 000176 000176 .-176
658 000176 000000 SWREG: .WORD 0
659
660 ;START ADDRESS*****
661
662 000200 000200 .-200
663 000200 000137 001176 JMP START ;PROGRAM START
664
665 ;RESTART ADDRESS*****
666 000210 000210 .-210
667 000210 000137 001730 JMP ST2
668
669 ;TMO3 INTERRUPT VECTOR*****
670
671 000224 000224 .-224
672 000224 012444 MTINT ;TAPE INTERRUPT HANDLER ADDRESS
673 000226 000340 340
674
```

```
675
676          000510          .=510
677          :MASS BUS REGISTER EQUIVS*****
678
679 000510 172440          C1: 172440
680 000512 172442          WC: 172442
681 000514 172444          BA: 172444
682 000516 172446          FC: 172446
683 000520 172450          CS: 172450
684 000522 172452          DS: 172452
685 000524 172454          ER: 172454
686 000526 172456          AS: 172456
687 000530 172460          CC: 172460
688 000532 172462          DB: 172462
689 000534 172464          MR: 172464
690 000536 172466          DT: 172466
691 000540 172470          JN: 172470
692 000542 172472          TC: 172472
693
694          :ILLEGAL FUNCTION CODES
695
696 000544 005405          ILFT: 5405
697 000546 007415          7415
698 000550 016423          16423
699 000552 020437          20437
700 000554 022443          22443
701 000556 025447          25447
702 000560 031455          31455
703 000562 033465          33465
704 000564 036473          36473
705
706          :CONSTANTS*****
707
708 000566 177776          PSW: 177776          :PROCESSOR STATUS
709 000570 177570          SWR: 177570          :SWITCH REGISTER
710 000572 177560          TKS: 177560          :TTY READER STATUS
711 000574 177562          TKB: 177562          :TTY READ BUFFER
712 000576 177564          TPS: 177564          :TTY PUNCH STATUS
713 000600 177566          TPB: 177566          :TTY PUNCH BUFFER
714 000602 000000          SLVTYP: .WORD 0          :INDICATES SLAVE TYPE (0/1 = TE16/TU77)
715 000604 000020          ITAMT: 20          :ITERATION AMOUNT
716 000606 000224          VECT: 224          :INTERRUPT VECTOR(RH)
717 000610 172440          REGS: 172440          :STARTING REGISTER ADDRESS
```

718
719
720
721
722
723 000612
724 000612 000000
725 000614 000000
726 000616 000000
727 000620 000000
728 000622 000000
729 000624 000000
730 000626 000000
731 000630 000000
732 000632 000000
733 000634 000000
734 000636 000000
735 000640 000000
736 000642 000000
737 000644 000000
738 000646 000000
739 000650 000000
740 000652 000000
741 000654 000000
742 000656 000000
743 000660 000000
744 000662 000000
745 000664 000000
746 000666 000000
747 000670 000000
748 000672 000000
749 000674 000000
750 000676 000000
751 000700 000000
752 000702 000000
753 000704 000000
754 000706 000000
755 000710 000000
756 000712 000000
757 000714 000000
758 000716 000000
759 000720 000000
760 000722 000000
761 000724 000000
762 000726 000000
763 000730 000000
764 000732 000000
765 000734 000000
766 000736 000000
767 000740 000000
768 000742 000000
769 000744 000000
770 000746 000000
771 000750 000000
772 000752 000000
773 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
EPRF: 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

774	000756	000000	DATA0:	0	
775	000760	000000	RDAD:	0	
776	000762	000000	W2FLG:	0	
777	000764	000000	DERFL:	0	
778	000766	000000	PREFL:	0	
779	000770	000000	SERFL:	0	
780	000772	000000	CRCNT:	0	
781	000774	000000	UDES:	0	
782	000776	000000	WPGFL:	0	
783	001000	000000	PATRN:	0	
784	001002	000000	STATF:	0	
785	001004	000000	RDRVF:	0	
786	001006	000000	RCDP:	0	
787	001010	000000	STATC:	0	
788	001012	000000	SKAT:	0	
789	001014	000000	PCNTR:	0	;PASS COUNTER
790	001016	000000	DCHKFL:	.WORD 0	;DATA CHECK FLAG 0/1 - CHECK/DO NOT CHECK
791	001020	000000	CRCFLG:	.WORD 0	;CRC CORRECTION TEST IN PROGRESS
792	001022		ENDFLG:		
793					
794					
795					;EXPT WRAP STATUS*****
796	001022	000000	WCS1:	0	
797	001024	000000	WCS2:	0	
798	001026	000000	WDS:	0	
799	001030	000000	WER:	0	
800					
801					;DATA PATTERN GENERATORS*****
802					
803	001032		DATBL:		
804	001032	005136	DATA0:	DAT1 ;ALL ONE BITS	
805	001034	005156	DATA1:	DAT2 ;ALL ZERO BITS	
806	001036	005162	DATA2:	DAT3 ;ALTERNATING ONE/ZERO BITS	
807	001040	005170	DATA3:	DAT4 ;ALTERNATING PARITY CHARACTERS	
808					
809					;CORE DUMP PATTERNS*****
810					
811	001042	000005	WCDP2:	5	
812	001044	000005		5	
813	001046	000012		12	
814	001050	000012		12	
815	001052	000000		0	
816	001054	000017	WCDPO:	17	
817	001056	000017		17	
818	001060	000017		17	
819	001062	000017		17	
820	001064	000000		0	

821
822 ;LOGIC TEST ENTRY TABLE*****
823

824	001066	000000	TSTTBL: 0
825	001070	000000	0
826	001072	002412	LT1
827	001074	002412	LT1
828	001076	002524	LT2
829	001100	002524	LT2
830	001102	002576	LT3
831	001104	002576	LT3
832	001106	002704	LT4
833	001110	002704	LT4
834	001112	002756	LT5
835	001114	002756	LT5
835	001116	003064	LT6
837	001120	003064	LT6
838	001122	003142	LT7
839	001124	003142	LT7
840	001126	003250	LT10
841	001130	003250	LT10
842	001132	003322	LT11
843	001134	003322	LT11
844	001136	003434	LT12
845	001140	003434	LT12
846	001142	003562	LT13
847	001144	003562	LT13
848	001146	003632	LT14
849	001150	003632	LT14
850	001152	003702	LT15
851	001154	003702	LT15
852	001156	003764	LT16
853	001160	003764	LT16
854	001162	004352	LT17
855	001164	004352	LT17
856	001166	004664	LT20
857	001170	004664	LT20
858	001172	002304	
859	001174	000020	

TADX: .WORD TEND
TLAST: .WORD 20

;CONTAINS # OF TESTS

```

860                                     .EVEN
861                                     ;PROGRAM START AND HOUSEKEEPING*****
862
863 001176 012706 000500                START: MOV    #500,SP          ;SET STACK POINTER
864 001202 013746 000004                MOV    @#4,-(SP)        ;SAVE ERROR TRAP
865 001206 013746 000006                MOV    @#6,-(SP)
866 001212 012737 001236 000004        MOV    #1$,@#4          ;SET TIME OUT TRAP TO GO TO 1$
867 001220 005037 000006                CLR    @#6
868 001224 022777 177777 177336        CMP    #177777,@SWR     ;USE SOFTWARE SWITCH IF SWR - 177777
869 001232 001402                        BEQ    2$                ;OR TIMES OUT
870 001234 000404                        BR     3$                ;OTHERWISE USE HARDWARE SWR
871 001236 022626                        1$:  CMP    (SP)+,(SP)+    ;RESET STACK
872 001240 012737 000176 000570        2$:  MOV    #SWREG,SWR    ;SET SWR = TO ADDRESS OF SOFTWARE SWR
873 001246 012637 000006                3$:  MOV    (SP)+,@#6     ;RESTORE ERROR TRAP
874 001252 012637 000004                MOV    (SP)+,@#4
875 001256 005027                        CLR    (PC)+
876 001260 000000                (HNFLG: .WORD 0        ;:CLEAR CHAIN INDICATOR
                                     ;:CHAIN MODE INDICATOR
                                     ;:1/0 = CHAIN/NOT CHAIN MODE
877
878 001262 005737 000042                TST    @#42             ;:BRANCH IF IN DUMP MODE
879 001266 001407                        BEQ    50$
880 001270 012737 000176 000570        MOV    #SWREG,SWR      ;:INVOKE SOFTWARE SWR
881 001276 005237 001260                INC    CHNFLG          ;:SET CHNFLG = CHAIN MODE
882 001302 000137 001306                JMP    SCHN            ;:GO TO CHAIN ADDRESS
883 001306
884 001306 000240                50$: NOP
885 001310 122737 000006 00004'        SCHN: 4$:  CMPB   #6,@#41      ;:BRANCH IF NOT LOADED VIA TMDP
886 001316 001005                        BNE    5$
887 001320 012704 014651                MOV    #MSG62,R4       ;:ADVISE USER TO REMOVE MEDIA FROM UUT
888 001324 004737 013120                JSR    PC,TTOUT
889 001330 000000                HALT
890 001332 012704 014042                5$:  MOV    #MSG1,R4
891 001336 004737 013120                JSR    PC,TTOUT        ;:PRINT TITLE
892 001342 005737 001260                TST    CHNFLG          ;:SEE IF IN CHAIN MODE
893 001346 001402                        BEQ    53$             ;:IF NOT: BR
894 001350 000137 001754                JMP    TSCD            ;:ELSE GO START TEST
895 001354 112737 000043 014042        53$: MOVB   #'#,MSG1     ;:DO NOT PRINT TITLE ON RESTART
896 001362 012704 014540                MOV    #MSG44,R4
897 001366 004737 013120                JSR    PC,TTOUT        ;:REQUEST REGISTER ADDRESS
898 001372 013703 000610                MOV    REGS,R3
899 001376 004737 013246                JSR    PC,OCTP         ;:PRINT CURRENT ADDRESS
900 001402 012705 000610                MOV    #REGS,R5       ;:SET ADDRESS SAVE LOC
901 001406 012701 000007                MOV    #7,R1          ;:SET SIZE OF RESPONSE
902 001412 012702 176400                MOV    #176400,R2     ;:SET UPPER LIMIT
903 001416 012703 172300                MOV    #172300,R3     ;:SET LOWER LIMIT
904 001422 004737 012576                JSR    PC,TTR         ;:GO GET RESPONSE
905 001426 012704 014562                MOV    #MSG45,R4
906 001432 004737 013120                JSR    PC,T'OUT       ;:REQUEST VECTOR
907 001436 013703 000606                MOV    VECT,R3
908 001442 004737 013246                JSR    PC,OCTP        ;:PRINT CURRENT VECTOR
909 001446 012705 000606                MOV    #VECT,R5       ;:SET ADDRESS SAVE LOC
910 001452 012701 000004                MOV    #4,R1          ;:SET SIZE OF RESPONSE
911 001456 012702 000224                MOV    #224,R2        ;:SET UPPER LIMIT
912 001462 012703 000150                MOV    #150,R3        ;:SET LOWER LIMIT
913 001466 004737 012576                JSR    PC,TTR         ;:GO GET RESPONSE
914 001472 013700 000606                MOV    VECT,R0        ;:GET VECTOR
915 001476 012720 012444                MOV    #MTINT,(R0)+   ;:LOAD INTERRUPT ADDRESS IN VECTOR

```

```

916 001502 012710 000340      MOV      #340,(R0)      ;LOAD PRIORITY
917 001506 013700 000610      MOV      REGS,R0       ;GET START OF REGS
918 001512 012701 000016      MOV      #16,R1       ;SET NUMBER OF REGS
919 001516 012702 000510      MOV      #C1,R2       ;GET START OF TABLE
920 001522 010022          6$: MOV      R0,(R2)+     ;BUILD TABLE
921 001524 062700 000002      ADD      #2,R0        ;BUMP ADDRESS
922 001530 005301          DEC      R1           ;SEE IF DONE
923 001532 001373          BNE      6$          ;IF NOT: BR
924 001534 012702 000612      MOV      #STFLGS,R2   ;
925 001540 012700 000210      MOV      #ENDFLG-STFLGS,R0 ;GET # OF FLAGS TO CLEAR
926 001544 006200          ASR      R0           ;FORM COUNT
927 001546 005022          7$: CLR      (R2)+     ;CLEAR FLAGS + COUNTERS
928 001550 005300          DEC      R0
929 001552 001375          BNE      7$
930 001554 012704 014604      MOV      #MSG57,R4    ;REQUEST TMO3 DRIVE #
931 001560 004737 013120      JSR      PC,TTOUT
932 001564 013703 000622      MOV      DRVN,R3     ;GET CURRENT DRIVE
933 001570 004737 013246      JSR      PC,OCTP    ;AND TYPE IT
934 001574 012705 000622      MOV      #DRVN,R5    ;TTR ROUTINE RETURNS DRIVE TO (R5)
935 001600 012701 000002      MOV      #2,R1       ;LIMIT RESPONSE TO 1 CHARACTER
936 001604 012702 000007      MOV      #7,R2       ;BETWEEN 0 AND 7
937 001610 012703 000000      MOV      #0,R3
938 001614 004737 012576      JSR      PC,TTR     ;GET RESPONSE & PUT IN DRVN
939 001620 012704 014622      MOV      #MSG58,R4    ;REQUEST SLAVE #
940 001624 004737 013120      JSR      PC,TTOUT
941 001630 013703 000662      MOV      SLVN,R3     ;GET CURRENT SLAVE #
942 001634 004737 013246      JSR      PC,OCTP    ;AND TYPE IT
943 001640 012705 000662      MOV      #SLVN,R5    ;TTR ROUTINE RETURNS REPONSE TO (R5)
944 001644 012701 000002      MOV      #2,R1       ;LIMIT RESPONSE TO 1 CHARACTER
945 001650 012702 000007      MOV      #7,R2       ;BETWEEN 0-7
946 001654 012703 000000      MOV      #0,R3
947 001660 004737 012576      JSR      PC,TTR     ;GET RESPONSE & PUT IT IN SLVN
948
949 001664 012704 014745      MOV      #MSG64,R4    ;REQUEST SLAVE TYPE
950 001670 004737 013120      JSR      PC,TTOUT
951 001674 013703 000602      MOV      SLVTYP,R3   ;GET CURRENT SLAVE TYPE
952 001700 004737 013246      JSR      PC,OCTP    ;TYPE CURRENT VALUE
953 001704 012705 000602      MOV      #SLVTYP,R5  ;TTR ROUTINE STORES RESPONSE IN (R5)
954 001710 012701 000002      MOV      #2,R1       ;LIMIT RESPONSE TO 1 CHAR
955 001714 012702 000001      MOV      #1,R2       ;HI LIMIT
956 001720 012703 000000      MOV      #0,R3       ;LO LIMIT
957 001724 004737 012576      JSR      PC,TTR     ;GET RESPONSE & STORE IN SLVTYP
958
959          ;START 210
960 001730 012706 000500      ST2: MOV      #500,SP  ;SET STACK PTR
961 001734 005037 001004      CLR      RDRVF      ;CLEAR READ REVERSE FLAG
962 001740 005037 001014      CLR      PCNTR
963 001744 005037 001020      CLR      CRCFLG    ;SET CRC FLAG = CRC NOT IN PROGRESS
964 001750 004737 013700      JSR      PC,GTSWR  ;GET SOFTWARE SWITCHES

```

```

965
966 ;TEST SCHEDULAR*****
967
968 001754 052777 000100 176610 TSCD: BIS #100,@TKS ;SET KEYBOARD IE BIT
969 001762 005037 000776 CLR WPGFL ;CLEAR WRAP PATRN FLAG
970 001766 005037 000736 CLR STFLG ;CLEAR SINGLE TEST FLAG
971 001772 017700 176572 MOV @SWR,RO
972 001776 042700 177700 BIC #177700,RO ;BRANCH IF SINGLE TEST SELECTED
973 002002 001122 BNE STSCD ;GO SELECT SINGLE TEST
974 002004 005737 001260 TST CHNFLG ;:BRANCH IF NOT IN CHAIN MODE
975 002010 001457 BEQ TSCDA
976 002012 012737 177777 000622 MOV #-1,DRVN ;:INITIALIZE DRIVE #
977 002020 012737 177777 000662 NXTDRV: MOV #-1,SLVN ;:INITIALIZE SLAVE #
978 002026 012777 000040 176464 1$: MOV #40,@CS ;:INIT CONTROLLER
979 002034 005237 000622 INC DRVN ;:STEP DRIVE #
980 002040 022737 000010 000622 CMP #10,DRVN ;:EXIT IF ALL DRIVES TESTED
981 002046 001521 BEQ $DONE ;:FOR AVAILABILITY
982 002050 013777 000622 176442 MOV DRVN,@CS ;:LOAD DRIVE #
983 002056 005777 176426 TST @C1 ;:ACCESS DRIVE
984 002062 032777 010000 176430 BIT #10000,@CS ;:BRANCH IF DRIVE NON EXISTANT
985 002070 001356 BNE 1$ ;:(NED = 1)
986 002072 005237 000662 NXTSLV: INC SLVN ;:STEP SLAVE # AND BRANCH
987 002076 001011 BNE 1$ ;:IF NOT SLAVE 0
988 002100 005737 000622 TST DRVN ;:BRANCH IF NOT DRIVE # 0
989 002104 001006 BNE 1$
990 002106 122737 000006 000041 CMPB #6,@#41 ;:BRANCH IF NOT TMDP
991 002114 001002 BNE 1$
992 002116 005237 000662 INC SLVN ;:STEP TO SLAVE # 1
993 002122 022737 000010 000662 1$: CMP #10,SLVN ;:BRANCH IF ALL SLAVES TESTED
994 002130 001733 BEQ NXTDRV ;:FOR AVAILABILITY
995 002132 013777 000662 176402 MOV SLVN,@TC ;:LOAD SLAVE UNIT #
996 002140 032777 002000 176370 BIT #2000,@DT ;:BRANCH IF SLAVE NOT
997 002146 001751 BEQ NXTSLV ;:PRESENT (SPR = 0)
998 002150 012737 001066 000740 TSCDA: MOV #TSTTBL,LTADD
999 002156 062737 000004 000740 TSCD0: ADD #4,LTADD
1000 002164 013737 000740 000712 TSCD1: MOV LTADD,ITRLP
1001 002172 062737 000002 000712 ADD #2,ITRLP ;:SET ITERATION ADDRESS
1002 002200 005037 000616 CLR HDRFL ;:CLEAR PRINT HEADER FLAG
1003 002204 017700 176530 MOV @LTADD,RO ;:SET POINTER TO TEST
1004 002210 000110 JMP (RO) ;:GO TO TEST
1005 002212 032777 002000 176350 TSCD2: BIT #2000,@SWR ;:SEE IF HALT ON TEST
1006 002220 001403 BEQ TSCD3 ;:IF NOT: BR
1007 002222 000000 HALT
1008 002224 005037 000776 CLR WPGFL ;:CLEAR WRAP DATA GENERATOR FLAG
1009 002230 005737 000736 TSCD3: TST STFLG ;:SE IF SINGLE TEST
1010 002234 001750 BEQ TSCD0 ;:IF NOT: BR
1011 002236 017700 176326 MOV @SWR,RO
1012 002242 042700 177700 BIC #177700,RO ;:MASK TEST NUMBER
1013 002246 001642 BEQ TSCD ;:IF SO: BR
1014 002250 012737 000001 000736 STSCD: MOV #1,STFLG ;:SET SINGLE TEST FLAG
1015 002256 023700 001174 CMP TLAST,RO ;:SEE IF EXCEEDED TESTS
1016 002262 002410 BLT TEND ;:IF SO: BR
1017 002264 006300 ASL RO
1018 002266 006100 ROL RO ;:SET TABLE MODIFIER
1019 002270 012737 001066 000740 MOV #*STTBL,LTADD
1020 002276 060037 000740 ADD RO,LTADD ;:SET TEST POINTER
    
```


1021	002302	000730			BR	TSCD1		
1022	002304	005737	001260	TEND:	TST	CHNFLG		;BRANCH IF IN CHAIN MODE
1023	002310	001270			BNE	NXTSLV		
1024	002312	012704	014521	\$DONE:	MOV	#MSG41,R4		
1025	002316	004737	013120		JSR	PC,TTOUT		;PRINT END OF PASS
1026	002322	013703	001014		MOV	PCNTR,R3		
1027	002326	004737	013246		JSR	PC,OCTP		;PRINT PASS NUMBER
1028	002332	005000			CLR	RO		
1029	002334	005300		1\$:	DEC	RO		;DELAY WAITING FOR
1030	002336	001376			BNE	1\$;TTY TO FINISH
1031	002340	013700	000042		MOV	@#42,RO		;GET ACT11 RETURN ADDRESS
1032	002344	001405			BEQ	HERE		;BRANCH IF NOT ACT11
1033	002346	000005			RESET			
1034	002350	004710		\$ENDAD:	JSR	PC,(RO)		
1035	002352	000240			NOP			
1036	002354	000240			NOP			
1037	002356	000240			NOP			
1038	002360	000240		HERE:	NOP			
1039	002362	005737	001260		TST	CHNFLG		;BRANCH IF IN CHAIN MODE
1040	002366	001005			BNE	TENDX		
1041	002370	032777	010000 176172		BIT	#10000,@SWR		;SEE IF HALT ON PASS
1042	002376	001001			BNE	TENDX		;IF NOT: BR
1043	002400	000000			HALT			
1044	002402	005237	001014	TENDX:	INC	PCNTR		;BUMP PASS COUNTER
1045	002406	000137	001754		JMP	TSCD		;RESTART
1046								

```

1047                                     ;THESE TESTS CHECK DATA FORMATTING
1048                                     ;AND TRANSFER THROUGH THE TM03 WRAP AROUND MODES
1049
1050                                     ;LOGIC TEST 1: WRAP 3, NRZ, NORMAL ODD *****
1051
1052 002412 012737 004270 001022 LT1:  MOV    #4270,WCS1      ;SET EXPT CS1
1053 002420 012737 000100 001024      MOV    #100,WCS2      ;SET EXPT CS2
1054 002426 012737 010600 001026      MOV    #10600,WDS     ;SET EXPT DS
1055 002434 012737 000000 001030      MOV    #0,WER         ;SET EXPT ER
1056 002442 012737 015056 000620      MOV    #MSLT1,EMADDR  ;SET HEADER
1057 002450 012737 001700 000774      MOV    #1700,UDES     ;SET NRZ,NORMAL, ODD
1058 002456 005037 001000                LT1A: CLR    PATRN      ;POINT TO PATTERN 0
1059 002462 012737 002470 000710      MOV    #LT1B,SCOLP    ;SET SCOPE ADDRESS
1060 002470 004737 005316                LT1B: JSR    PC,WAM3    ;GO DO WRAP 3
1061 002474 005237 001000                INC    PATRN          ;BUMP PATTERN POINTER
1062 002500 032737 000004 001000      BIT    #4,PATRN       ;SEE IF DONE
1063 002506 001770                BEQ    LT1B           ;IF NOT: BR
1064 002510 004737 012146                JSR    PC,ITER        ;GO SEE IF ITERATIONS
1065 002514 005037 001004                CLR    RDRVF          ;CLEAR READ REVERSE FLAG
1066 002520 000137 002212                JMP    TSCD2          ;RETURN TO SCHEDULAR
1067
1068                                     ;LOGIC TEST 2: WRAP 3, PE, ORMAL, ODD*****
1069
1070 002524 000240                LT2:  NOP
1071 002526 012737 004270 001022 LT2A:  MOV    #4270,WCS1      ;SET EXPT CS1
1072 002534 012737 000100 001024      MOV    #100,WCS2      ;SET EXPT CS2
1073 002542 012737 010640 001026      MOV    #10640,WDS     ;SET EXPT DS
1074 002550 012737 000000 001030      MOV    #0,WER         ;SET EXPT WER
1075 002556 012737 015124 000620      MOV    #MSLT2,EMADDR  ;SET HEADER
1076 002564 012737 002300 000774      MOV    #2300,UDES     ;SET PE, NORMAL, ODD
1077 002572 000137 002456                JMP    LT1A           ;EXECUTE TEST SEQUENCE
1078
1079                                     ;LOGIC TEST 3: WRAP 2, NRZ, NORMAL, ODD*****
1080
1081 002576 012737 004260 001022 LT3:  MOV    #4260,WCS1      ;SET EXPT CS1
1082 002604 012737 000100 001024      MOV    #100,WCS2      ;SET EXPT CS2
1083 002612 012737 010600 001026      MOV    #10600,WDS     ;SET EXPT DS
1084 002620 012737 000000 001030      MOV    #0,WER         ;SET EXPT WER
1085 002626 012737 015171 000620      MOV    #MSLT3,EMADDR  ;SET HEADER
1086 002634 012737 001700 000774      MOV    #1700,UDES     ;SET TO NRZ,NORMAL, ODD
1087 002642 005037 001000                LT3A: CLR    PATRN      ;POINT TO PATTERN 0
1088 002646 012737 002654 000710      MOV    #LT3B,SCOLP    ;SET SCOPE ADDRESS
1089 002654 004737 005252                LT3B: JSR    PC,W M2    ;GO DO WRAP 2
1090 002660 005237 001000                INC    PATRN          ;BUMP POINTER
1091 002664 032737 000004 001000      BIT    #4,PATRN       ;SEE IF DONE
1092 002672 001770                BEQ    LT3B           ;IF NOT: BR
1093 002674 004737 012146                JSR    PC,ITER        ;GO SEE IF ITERATIONS
1094 002700 000137 002212                JMP    TSCD2          ;RETURN TO SCHEDULAR

```

```

1095
1096
1097
1098 002704 000240
1099 002706 012737 004260 001022
1100 002714 012737 000100 001024
1101 002722 012737 010640 001026
1102 002730 012737 000000 001030
1103 002736 012737 015237 000620
1104 002744 012737 002300 000774
1105 002752 000137 002642
1106
1107
1108
1109 002756 012737 004260 001022
1110 002764 012737 000100 001024
1111 002772 012737 010600 001026
1112 003000 012737 000000 001030
1113 003006 012737 015304 000620
1114 003014 012737 001700 000774
1115 003022 005037 001000
1116 003026 012737 003034 000710
1117 003034 004737 005242
1118 003040 005237 001000
1119 003044 032737 000004 001000
1120 003052 001770
1121 003054 004737 012146
1122 003060 000137 002212
1123
1124
1125
1126 003064 000240
1127 003066 004737 011626
1128 003072 012737 004260 001022
1129 003100 012737 000100 001024
1130 003106 012737 010640 001026
1131 003114 012737 000000 001030
1132 003122 012737 015352 000620
1133 003130 012737 002300 000774
1134 003136 000137 003022

```

```

;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

```

```

1135
1136
1137
1138 003142 012737 144260 001022 LT7: MOV #144260,WCS1 :SET EXPT CS1
1139 003150 012737 000100 001024 MOV #100,WCS2 :SET EXPT CS2
1140 003156 012737 150600 001026 MOV #150600,WDS :SET EXPT DS
1141 003164 012737 000200 001030 MOV #200,WER :SET EXPT ER
1142 003172 012737 015417 000620 MOV #MSLT7,EMADDR :SET HEADER
1143 003200 012737 001700 001774 MOV #1700,UDES :SET NRZ, NORMAL, ODD
1144 003206 005037 001000 LT7A: CLR PATRN :POINT TO PATTERN 0
1145 003212 012737 003220 001710 MOV #LT7B,SCOLP :SET SCOPE ADDRESS
1146 003220 004737 005176 LT7B: JSR PC,WAMO :GO DO WRAP 0
1147 003224 005237 001000 INC PATRN :BUMP POINTER
1148 003230 032737 000004 001000 BIT #4,PATRN :SEE IF DONE
1149 003236 001770 BEQ LT7B :IF NOT: BR
1150 003240 004737 012146 JSR PC,ITER :GO SEE IF ITERATIONS
1151 003244 000137 002212 JMP TSCD2 :RETURN TO SCHEDULAR
1152
1153
1154
1155 003250 000240 L 10: NOP
1156 003252 012737 004260 001022 L*10A: MOV #4260,WCS1 :SET EXPT CS1
1157 003260 012737 000100 001024 MOV #100,WCS2 :SET EXPT CS2
1158 003266 012737 010640 001026 MOV #10640,WDS :SET EXPT DS
1159 003274 012737 000000 001030 MOV #0,WER :SET EXPT ER
1160 003302 012737 015465 000620 MOV #MSLT10,EMADDR :SET HEADER
1161 003310 012737 002300 000774 MOV #2300,UDES :SET PE, NORMAL, ODD
1162 003316 000137 003206 JMP LT7A :GO EXECUTE TEST SEQUENCE

```

```
1163 ;LOGIC TEST 11: CORE DUMP WRITE, WAM2*****
1164
1165 003322 012737 004260 001022 LT11: MOV #4260,WCS1 :SET EXPT CS1
1166 003330 012737 000100 001024 MOV #100,WCS2 :SET EXPT CS2
1167 003336 012737 010600 001026 MOV #10600,WDS :SET EXPT DS
1168 003344 012737 000000 001030 MOV #0,WER :SET EXPT ER
1169 003352 012737 015533 000620 MOV #MSLT11,EMADDR :SET HEADER
1170 003360 012737 001720 000774 MOV #1720,UDES :SET NRZ, CORE DUMP, ODD
1171 003366 005037 001000 CLR PATRN :POINT TO PATTERN 0
1172 003372 012737 003400 000710 MOV #LT11A,SCOLP :SET SCOPE ADDRESS
1173 003400 004737 005252 LT11A: JSR PC,WAM2 :GO DO WAM 2
1174 003404 022737 000002 001000 CMP #2,PATRN :SEE IF DONE
1175 003412 001404 BEQ LT11X :IF SO: BR
1176 003414 012737 000002 001000 MOV #2,PATRN :SELECT PATTERN 2
1177 003422 000766 BR LT11A :CONTINUE
1178 003424 004737 012146 LT11X: JSR PC,ITER :GO SEE IF ITERATIONS
1179 003430 000137 002212 JMP TSCD2 :RETURN TO SCHEDULES
```

```
1180
1181 ;LOGIC TEST 12: CORE DUMP READ, WAM 3*****
1182
1183 003434 012737 004270 001022 LT12: MOV #4270,WCS1 :SET EXPT CS1
1184 003442 012737 000100 001024 MOV #100,WCS2 :SET EXPT CS2
1185 003450 012737 010600 001026 MOV #10600,WDS :SET EXPT DS
1186 003456 012737 000000 001030 MOV #0,WER :SET EXPT ER
1187 003464 012737 015604 000620 MOV #MSLT12,EMADDR :SET HEADER
1188 003472 012737 001720 000774 MOV #1720,UDES :SELECT NRZ, CORE DUMP, ODD
1189 003500 005037 001000 CLR PATRN :SELECT PATTERN 0
1190 003504 012737 003520 000710 MOV #LT12A,SCOLP :SET SCOPE ADDRESS
1191 003512 012737 001054 001006 MOV #WCDP0,RCDP :POINT TO PATTERN 0
1192 003520 004737 005316 LT12A: JSR PC,WAM3 :GO DO WAM3
1193 003524 022737 000002 001000 CMP #2,PATRN :SEE IF DONE
1194 003532 001407 BEQ LT12X :IF SO: BR
1195 003534 012737 000002 001000 MOV #2,PATRN :SELECT PATTERN 2
1196 003542 012737 001042 001006 MOV #WCDP2,RCDP :POINT TO PATTERN 2
1197 003550 000763 BR LT12A :CONTINUE
1198 003552 004737 012146 LT12X: JSR PC,ITER :GO SEE IF ITERATION
1199 003556 000137 002212 JMP TSCD2 :RETURN TO SCHEDULE
```

```

1200
1201                ;LOGIC TEST 13: EVEN PARITY WRITE: WAM 1(M8933)*****
1202
1203 003562 012737 004260 001022 LT13: MOV #4260,WCS1 ;SET EXPT CS1
1204 003570 012737 000100 001024 MOV #100,WCS2 ;SET EXPT CS2
1205 003576 012737 010600 001026 MOV #10600,WDS ;SET EXPT DS
1206 003604 012737 000000 001030 MOV #0,WER ;SET EXPT ER
1207 003612 012737 015654 000620 MOV #MSLT13,EMADDR ;SET HEADER
1208 003620 012737 001710 000774 MOV #1710,UDES ;SET NRZ, NORMAL, EVEN
1209 003626 000137 003022 JMP LT5A ;GO EXECUTE WAM 1
1210
1211                ;LOGIC TEST 14: EVEN PARITY READ: WAM 0(M8933 M8934)*****
1212
1213 003632 012737 144260 001022 LT14: MOV #144260,WCS1 ;SET EXPT CS1
1214 003640 012737 000100 001024 MOV #100,WCS2 ;SET EXPT CS2
1215 003646 012737 150600 001026 MOV #150600,WDS ;SET EXPT DS
1216 003654 012737 000200 001030 MOV #200,WER ;SET EXPT ER
1217 003662 012737 015735 000620 MOV #MSLT14,EMADDR ;SET HEADER
1218 003670 012737 001710 000774 MOV #1710,UDES ;SET NRZ, NORMAL, EVEN
1219 003676 000137 003206 JMP LT7A ;GO DO WAM 0
1220
1221                ;LOGIC TEST 15: READ REVERSE: WAM 3(M8906)*****
1222
1223 003702 012737 004276 001022 LT15: MOV #4276,WCS1 ;SET EXPT CS1
1224 003710 012737 000100 001024 MOV #100,WCS2 ;SET EXPT CS2
1225 003716 012737 010640 001026 MOV #10640,WDS ;SET EXPT DS
1226 003724 012737 000000 001030 MOV #0,WER ;SET EXPT ER
1227 003732 012737 016014 000620 MOV #MSLT15,EMADDR ;SET HEADER
1228 003740 012737 002300 000774 MOV #2300,UDES ;SELECT PE,NORMAL,ODD
1229 003746 000240 NOP
1230 003750 000240 NOP
1231 003752 012737 000001 001004 MOV #1,RDRVF ;SET READ REVERSE FLAG
1232 003760 000137 002456 JMP LT1A ;GO DO WAM 3, REVERSE
1233
    
```

```
1234  
1235  
1236  
1237  
1238  
1239  
1240  
1241  
1242  
1243  
1244  
1245  
1246  
1247  
1248  
1249  
1250  
1251  
1252  
1253  
1254 003764 005037 001000  
1255 003770 012737 000401 004342  
1256 003776 012737 016061 000620  
1257 004004 012737 004012 000710  
1258  
1259 004012 012737 144260 001022  
1260 004020 012737 000100 001024  
1261 004026 012737 150600 001026  
1262 004034 012737 000200 001030  
1263 004042 012737 001700 000774  
1264  
1265 004050 005037 001020  
1266 004054 004737 00517  
1267 004060 012737 000001 001020  
1268 004066 013737 017624 004344  
1269  
1270 004074 013737 004344 004346  
1271 004102 013737 004342 004350  
1272 004110 043737 004342 004346  
1273 004116 043737 004344 004350  
1274 004124 053737 004350 004346  
1275 004132 013737 004346 004344  
1276  
1277 004140 012737 144260 001022  
1278 004146 012737 000110 001024  
1279 004154 012737 150600 001026  
1280 004162 012737 100300 001030  
1281 004170 004737 005424  
1282 004174 000240  
1283 004176 012737 000001 001004  
1284 004204 012737 144276 001022  
1285 004212 012737 000110 001024  
1286 004220 012737 150600 001026  
1287 004226 012737 001000 001030  
1288 004234 004737 005316  
1289 004240 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.

L116: 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,800BP1,ODD
;PARITY & PDP11 NORMAL MODE
CLR CRCFLG ;CLEAR CRC CORRECTION FLAG
JSR PC,WAM0 ;DO A WRAP 0 --- STEP A
MOV #1,CRCLC ;SET CRC ERROR CORRECTION IN PROGRESS
MOV WBUF,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

1290	004242	012737	004260	001022	MOV	#4260,WCS1	
1291	004250	012737	000110	001024	MOV	#110,WCS2	
1292	004256	012737	010600	001026	MOV	#10600,WDS	
1293	004264	012737	000000	001030	MOV	#0,WER	
1294	004272	005037	001020		CLR	CRCFLG	:CLEAR CRC CORRECTION IN PROGRESS FLAG
1295	004276	004737	005424		JSR	PC,WAM4	:GO TO WRAP 4 --- STEP D
1296							
1297	004302	006337	004342		ASL	CRCPAT	:SELECT NEXT TRACK TO BE ALTERED
1298	004306	103241			BCC	LT16A	:CONTINUE FOR ALL TRACKS
1299	004310	012737	000401	004342	MOV	#401,CRCPAT	:RESET BITS TO TRACK 1
1300	004316	005237	001000		INC	PATRN	:SELECT NEXT PATTERN
1301	004322	022737	000003	001000	CMP	#3,PATRN	:BRANCH IF NOT DONE
1302	004330	001230			BNE	LT16A	
1303	004332	004737	012146		JSR	PC,ITER	:ITERATION LOOP
1304	004336	000137	002212		JMP	TSCD2	:RETURN TO SCHEDULER
1305							
1306							
1307	004342	000000					:CONTAINS BITS TO BE ALTERED
1308	004344	000000					:CONTAINS DATA TO BE WRITTEN BY WRAP4
1309	004346	000000					:TEMPRARY STORAGE FOR XOR
1310	004350	000000					:TEMPOARY STORAGE FOR XOR
1311							


```
1312
1313
1314      ;LOGIC TEST 17: CRC CORRECTION - MULTIPLE BAD TRACKS
1315      ;THIS TEST CHECKS THAT CRC ERROR CORRECTION IS NOT PERFORMED WHEN
1316      ;MULTIPLE BAD TRACKS ARE DETECTED.
1317
1318 004352 012757 000002 001000 LT17:  MOV      #2,PATRN      ;SELECT PATTERN #2 (125125)
1319 004360 012737 001001 004342      MOV      #1001,CRCPAT    ;SELECT 2 BAD TRACKS
1320 004366 012737 016151 000620      MOV      #MSLT17,EMADDR  ;SET ERROR MESSAGE HEADER
1321 004374 012737 004402 000710      MOV      #LT17A,SCOLP    ;SET SCOPE LOOP ADDRESS
1322 004402 012737 144260 001022 LT17A: MOV      #144260,WCS1 ;SET EXPECTED VALUES ON COMPLETION
1323 004410 012737 000100 001024      MOV      #100,WCS2      ;OF WRAP 0 BELOW
1324 004416 012737 150600 001026      MOV      #150600,WDS
1325 004424 012737 000200 001030      MOV      #200,WER
1326 004432 012737 001700 000774      MOV      #1700,UDES     ;SET UNIT DESCRIPTION
1327 004440 005037 001020      CLR      CRCFLG        ;SET CRC CORRECTION NOT IN PROGRESS
1328 004444 004737 005176      JSR      PC,WAM0       ;DO A WRAP 0 --- STEP A
1329 004450 000240      NOP
1330 004452 012737 000002 001020      MOV      #2,CRCFLG     ;SET CRC CORRECTION IN PROGRESS
1331 004460 013737 017624 004344      MOV      WBUF,CRCDAT    ;GET DATA TO BE WRITTEN
1332 004466 043737 004342 004344      BIC      CRCPAT,CRCDAT  ;MODIFY DATA TO BE WRIT'EN
1333 004474 012737 144260 001022      MOV      #144260,WCS1  ;SET EXPECTED VALUES ON COMPLETION
1334 004502 012737 000110 001024      MOV      #110,WCS2     ;OF WRAP 4 BELOW
1335 004510 012737 150600 001026      MOV      #150600,WDS
1336 004516 012737 100300 001030      MOV      #100300,WER
1337 004524 004737 005424      JSR      PC,WAM4       ;DO A WRAP 4 --- STEP B
1338 004530 000240      NOP
1339 004532 012737 000001 001004      MOV      #1,RDRVF      ;SET READ REVERSE FLAG
1340 004540 012737 144276 001022      MOV      #144276,WCS1  ;SET EXPECTED VALUES ON COMPLETION
1341 004546 012737 000110 001024      MOV      #110,WCS2     ;OF WRAP 3 BELOW
1342 004554 012737 150600 001026      MOV      #150600,WDS
1343 004562 012737 001000 001030      MOV      #1000,WER
1344 004570 004737 005316      JSR      PC,WAM3       ;DO A WRAP 3 --- STEP C
1345 004574 000240      NOP
1346 004576 012737 144260 001022      MOV      #144260,WCS1  ;SET EXPECTED VALUES ON COMPLETION
1347 004604 012737 000110 001024      MOV      #110,WCS2     ;OF WRAP 4 BELOW
1348 004612 012737 150600 001026      MOV      #150600,WDS
1349 004620 012737 100100 001030      MOV      #100100,WER
1350 004626 005037 001020      CLR      CRCFLG        ;CLEAR CRC IN PROGRESS FLAG
1351 004632 013701 004344      MOV      CRCDAT,R1     ;GET DATA THAT WAS WRITTEN IN STEP B
1352 004636 012703 017624      MOV      #WBUF,R3      ;SET START OF WRITE BUFFER
1353 004642 004737 005142      JSR      PC,DAT1A      ;GO SET WRITE BUFFER
1354 004646 004737 005424      JSR      PC,WAM4       ;GO DO A WRAP 4 --- STEP D
1355 004652 000240      NOP
1356 004654 004737 012146      JSR      PC,ITER      ;ITERATE TEST
1357 004660 000137 002212      IMP      TSCD2        ;RETURN TO SCHEDULER
```

```
1358  
1359  
1360 :LOGIC TEST 20: READ REVERSE,NRZ,WRAP 3  
1361 :THIS TEST TESTS THAT A CRC ERROR OCCURS AFTER A READ REVERSE USING  
1362 :WRAP AROUND MODE 3 IN NRZ MODE  
1363 004664 005037 001000 LT20: CLR PATRN ;SET PATTERN # 0 (ALL 1'S)  
1364 004670 012737 144276 001022 MOV #144276,WCS1 ;SET EXPECTED VALUES ON COMPLETION  
1365 004676 012737 000100 001024 MOV #100,WCS2  
1366 004704 012737 150600 001026 MOV #150600,WDS  
1367 004712 012737 100000 001030 MOV #100000,WER  
1368 004720 012737 016235 000620 MOV #MSLT20,EMADDR ;SET ERROR MESSAGE ADDRESS  
1369 004726 012737 001700 000774 MOV #1700,UDES ;SET UNIT DESCRIPTION  
1370 004734 012737 004742 000710 MOV #LT20A,SCOLP ;SET SCOPE LOOP ADDRESS  
1371 004742 012737 000001 001004 LT20A: MOV #1,RDRVF ;SET READ REVERSE FLAG  
1372 004750 012737 000001 001016 MOV #1,DCHKFL ;SET DATA CHECK FLAG TO NOT CHACK DATA  
1373 004756 004737 005316 JSR PC,WAM3  
1374 004762 005037 001004 CLR RDRVF ;CLEAR READ REVERSE FLAG  
1375 004766 005037 001016 CLR DCHKFL ;CLEAR DATA CHECK FLAG  
1376 004772 004737 012146 JSR PC,ITER  
1377 004776 000137 002212 JMP TSCD2 ;RETURN TO SCHEDULER  
1
```

1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392 005002 005737 000736
1393 005006 001434
1394 005010 032777 000100 173552
1395 005016 001430
1396 005020 012704 016744
1397 005024 004737 013120
1398 005030 013703 001000
1399 005034 004737 013246
1400 005040 012705 001000
1401 005044 012701 000002
1402 005050 012702 000003
1403 005054 012703 000000
1404 005060 004737 012576
1405 005064 112737 000001 000777
1406 005072 113737 001000 000776
1407 005100 012703 017624
1408 005104 013701 001000
1409 005110 006301
1410 005112 004771 001032
1411 005116 012702 000202
1412 005122 012701 020236
1413 005126 005021
1414 005130 005302
1415 005132 001375
1416 005134 000207
1417
1418
1419
1420 005136 012701 177777
1421 005142 012702 000202
1422 005146 010123
1423 005150 005302
1424 005152 001375
1425 005154 000207
1426
1427
1428
1429 005156 005001
1430 005160 000770
1431
1432
1433
1434 005162 012701 125125

```
*****  
:DATA SETUP ROUTINE:  
:THIS ROUTINE IS USED TO GENERATE THE DATA PATTERNS  
:THE PATTERN TO BE WRITTEN IS SPECIFIED BY:  
:PATRN =0 ALL 1'S  
:       =1 ALL 0'S  
:       =2 ALTERNATE 1'S & 0'S (125125)  
:       =3 ALTERNATING PARITY (177377)  
*****  
DSUP: 1ST STFLG ;SEE IF SINGLE TEST  
      BEQ DSO ;IF NOT: BR  
1394 BIT #100,@SWR ;SEE IF SELECT PATTERN  
      BEQ DSO ;IF NOT: BR  
1396 MOV #WMSG3,R4  
1397 JSR PC,TTOUT ;REQUEST PATTERN NUMBER  
1398 MOV PATRN,R3  
1399 JSR PC,OCTP ;PRINT PATTERN NUMBER  
1400 MOV #PATRN,R5 ;GET ADDRESS OF PATRN ENTRY  
1401 MOV #2,R1 ;SET SIZE OF ENTRY  
1402 MOV #3,R2 ;SET UPPER LIMIT  
1403 MOV #0,R3 ;SET LOWER LIMIT  
1404 JSR PC,TTT ;GO GET PATTERN NUMBER  
1405 MOVB #1,WPGFL+1 ;SET FLAG  
1406 MOVB PATRN,WPGFL ;SET PATTERN NUMBER  
1407 MOV #WBUF,R3 ;R3 = ADDR OF WRITE BUFFER  
1408 MOV PATRN,R1 ;R1 = PATTERN SELECTOR  
1409 ASL R1 ;MAKE PATTERN SELECTOR EVEN  
1410 JSR PC,@DATBL(R1) ;GO GENERATE PATTERN  
1411 DS3: MOV #202,R2 ;R2=BUFFER SIZE +2  
1412 MOV #RBUF,R1 ;R1=READ DATA START  
1413 DS4: CLR (R1)+ ;CLEAR BUFFER  
1414 DEC R2 ;SEE IF DONE ALL  
1415 BNE DS4 ;IF NOT: BR  
1416 RTS PC ;EXIT  
  
:ALL ONES*****  
1420 DAT1: MOV #-1,R1 ;R1=DATA  
1421 DAT1A: MOV #202,R2 ;R2=WORD COUNT +2  
1422 1$: MOV R1,(R3)+ ;LOAD BUFFER  
1423 DEC R2 ;SEE IF DONE  
1424 BNE 1$ ;IF NOT: BR  
1425 RTS PC ;RETURN TO CALLER  
  
:ALL ZEROS~*****  
1429 DAT2: CLR R1 ;R1=DATA  
1430 BR DAT1A ;LOAD BUFFER  
  
:ONE/ZERO IN ALTERNATING CHARACTERS*****  
1434 DAT3: MOV #125125,R1 ;R1=DATA
```

```
1435 005166 000765          BR      DAT1A          ;LOAD BUFFER
1436
1437                          ;ALTERNATING PARITY CHARACTERS*****
1438
1439 005170 012701 177377    DAT4:  MOV      #177377,R1      ;R1=ALTERNATING PARITY DATA
1440 005174 000762          BR      DAT1A          ;GO LOAD BUFFER
1441
```

```

1442
1443                                     ;WRAP AROUND MODE 0 GLOBAL*****
1444
1445 005176 012737 000006 000746 WAM0:  MOV    #6,WAM           ;SET WAM NUMBER
1446 005204 012737 000060 000750 WAM01: MOV    #60,FUN
1447 005212 005037 000752          CLR    DATC
1448 005216 012737 017624 000756          MOV    #WBUF, DATAD      ;SET BUFFER ADDRESS
1449 005224 012737 020236 000760          MOV    #RBUF, RDAD      ;SET POINTER TO READ BUFFER
1450 005232 004737 005470          JSR    PC, SETUP        ;GO SET UP
1451 005236 000137 006050          JMP    EXEC
1452
1453                                     ;WRAP AROUND MODE 1 WRITE BUFFER*****
1454
1455 005242 012737 000010 000746 WAM1:  MOV    #10,WAM
1456 005250 000755          BR     WAM01
1457
1458                                     ;WRAP AROUND MODE 2 BIT FIDDLER WRITE*****
1459
1460 005252 012737 000012 000746 WAM2:  MOV    #12,WAM
1461 005260 012737 000060 000750          MOV    #60,FUN
1462 005266 005037 000752          CLR    DATC
1463 005272 012737 017624 000756          MOV    #WBUF, DATAD
1464 005300 012737 020236 000760          MOV    #RBUF, RDAD
1465 005306 004737 005470          WAM2A: JSR    PC, SETUP
1466 005312 000137 006050          JMP    EXEC
1467
1468                                     ;WRAP AROUND MODE 3 BIT FIDDLER READ*****
1469
1470 005316 012737 000014 000746 WAM3:  MOV    #14,WAM           ;SET WAM NUMBER
1471 005324 012737 000070 000750          MOV    #70,FUN         ;SET FUNCTION
1472 005332 012737 020236 000756          MOV    #RBUF, DATAD    ;SET BUFFER ADDRESS
1473 005340 012737 017624 000754          MOV    #WBUF, WTAD     ;SET POINTER TO WRITE BUFFER
1474 005346 005737 001004          TST    RDRVF
1475 005352 001411          BEQ    WAM3A
1476 005354 062737 000376 000756          ADD    #376, DATAD
1477 005362 062737 000377 000754          ADD    #377, WTAD
1478 005370 012737 000076 000750          MOV    #76,FUN         ;SET READ REVERSE CODE
1479 005376 032737 000020 000774 WAM3A: BIT    #20, UDES
1480 005404 001403          BEQ    WAM3B
1481 005406 013737 001006 000754          MOV    RCDP, WTAD
1482 005414 004737 005470          WAM3B: JSR    PC, SETUP    ;GO SET UP
1483 005420 000137 006050          JMP    EXEC            ;GO EXECUTE
1484
1485                                     ;WRAP AROUND MODE 4 CRC CORRECTION
1486                                     ;CALL: JSR    PC, WAM4
1487
1488 005424 012737 000030 000746 WAM4:  MOV    #30,WAM           ;SET MAINTENANCE MODE FUNCTION - WRAP 4
1489 005432 012737 000060 000750          MOV    #60,FUN         ;SET TAPE FUNCTION = WRITE FWD
1490 005440 005037 000752          CLR    DATC
1491 005444 012737 004344 000756          MOV    #CRCDAT, DATAD  ;SET ADRS OF WRITE BUFFER
1492 005452 012737 020236 000760          MOV    #RBUF, RDAD     ;SET READ BUFFER ADDRESS
1493 005460 004737 005470          JSR    PC, SETUP        ;GO SETUP REGISTERS
1494 005464 000137 006050          JMP    EXEC            ;GO EXECUTE

```

```

1495 ;REGISTER SETUP ROUTINE*****
1496
1497 005470 005737 001020 SETUP: TST CRCFLG ;DO NOT DO DATA SETUP NOR INIT
1498 005474 001004 BNE 1$ ;IF CRC CORRECTION IS IN PROGRESS
1499 005476 022737 000030 000746 CMP #30,WAM ;DO NOT INIT IF DOING WAM 4 --- STEP D
1500 005504 001004 BNE 2$
1501 005506 012777 000011 172774 1$: MOV #11,@C1 ;DO DRIVE CLEAR IF WAM4---STEP D
1502 005514 000412 BR SET1A
1503 005516 005737 000736 2$: TST STFLG ;SEE IF SINGLE TEST
1504 005522 001403 BEQ SET0 ;IF NOT: BR
1505 005524 005737 000776 TST WPGFL ;SEE IF HAVE SELECTED PATTERN
1506 005530 001002 BNE SET1 ;IF SO: BR
1507 005532 004737 005002 SET0: JSR PC,DSUP ;GO DO DATA SETUP
1508 005536 004737 012216 SET1: JSR PC,INIT ;INIT CONTROLLER,SELECT UNIT & DRIVE
1509 ;LOAD SLAVE DESC AND MOVE OFF BOT
1510 005542 012777 177400 172746 SET1A: MOV #-400,@FC ;SET FC=WCX2
1511 005550 032737 000020 000774 BIT #20,UDES ;SEE IF CORE DUMP
1512 005556 001403 BEQ SET2 ;IF NOT: BR
1513 005560 012777 177000 172730 MOV #-1000,@FC ;SET FC=WCX4
1514 005566 012777 177600 172716 SET2: MOV #-200,@WC ;SET WC
1515 005574 013777 000756 172712 MOV DATAD,@BA ;SET BUS ADDRESS
1516 005602 032777 010000 172710 BIT #10000,@CS ;ASSURE DRIVE THERE
1517 005610 001417 BEQ SP1 ;IF SO: BR
1518 005612 032777 020000 172750 BIT #20000,@SWR ;SEE IF PRINT ERRORS
1519 005620 001004 BNE SP01 ;IF NOT: BR
1520 005622 012704 016765 MOV #WMSG4,R4
1521 005626 004737 013120 JSR PC,TTOUT ;PRINT NON-EXISTANT DRIVE
1522 005632 032777 100000 172730 SP01: BIT #100000,@SWR ;SEE IF HALT ON ERROR
1523 005640 001401 BEQ SP0 ;IF NOT: BR
1524 005642 000000 HALT
1525 005644 000137 005536 SP0: JMP SET1 ;RESETUP
1526 005650 022737 000014 000746 SP1: CMP #14,WAM ;SEE IF WAM 3
1527 005656 001031 BNE SP1B ;IF NOT: BR
1528 005660 117737 173070 000752 MOVB @WTAD,DATC ;GET FIRST CHAR
1529 005666 042737 177400 000752 BIC #177400,DATC
1530 005674 000337 000752 SWAB DATC
1531 005700 052737 000200 000752 BIS #200,DATC ;SET PARITY
1532 005706 005737 001004 TST RDRVF ;SEE IF READ REVERSE
1533 005712 001403 BEQ SP1A ;IF NOT: BR
1534 005714 005337 000754 DEC WTAD ;DECREMENT POINTER
1535 005720 000410 BR SP1B
1536 005722 005237 000754 SP1A: INC WTAD ;BUMP POINTER
1537 005726 032737 000020 000774 BIT #20,UDES ;SEE IF CORE DUMP
1538 005734 001402 BEQ SP1B ;IF NOT: BR
1539 005736 005237 000754 INC WTAD ;BUMP POINTER AGAIN
1540 005742 053777 000774 172572 SP1B: BIS UDES,@TC ;SET UNIT DESCRIPTION (DEN,PAR,FMT)
1541 005750 052777 000001 172556 BIS #1,@MR ;SET MAINT MODE
1542 005756 053777 000746 172550 BIS WAM,@MR ;SET WAM
1543 005764 053777 000752 172542 BIS DATC,@MR ;SET DATA
1544 005772 013777 000750 172510 MOV FUN,@C1 ;SET FUNCTION
1545 006000 032777 040000 172514 BIT #40000,@DS ;ASSURE NO ERROR
1546 006006 001002 BNE SP3 ;IF ERROR: BR
1547 006010 000240 NOP
1548 006012 000207 RTS PC ;RETURN
1549 006014 032777 020000 172546 SP3: BIT #20000,@SWR ;SEE IF PRINT ERRORS
1550 006022 001004 BNF SP4 ;IF NOT: BR

```

1551	006024	012704	016726		MOV	#WMSG2,R4	
1552	006030	004737	013120		JSR	PC,TTOUT	:PRINT SETUP ERROR
1553	006034	032777	100000	172526	BIT	#100000,@SWR	:SEE IF HALT ON ERROR
1554	006042	001401			BEQ	SP5	:IF NOT: BR
1555	006044	000000			HALT		
1556	006046	000207			RTS	PC	:RETURN

```

1557                                     ;EXECUTE WAM ROUTINE*****
1558
1559 006050 000240 EXEC: NOP
1560 006052 000240 NOP
1561 006054 032777 000040 172452 BIT #40,@MR
1562 006062 001403 BEQ EX0 EX0 ;ASSURE MAINT CLOCK IS ZERO
1563 006064 042777 000040 172442 BIC #40,@MR ;IF NOT: CLEAR IT
1564 006072 022737 000010 000746 EX0: CMP #10,WAM ;BRANCH IF NOT WAM 0
1565 006100 003402 BLE 2$
1566 006102 000137 006526 1$: JMP EXW2 ;GO DO WAM 0
1567 006106 022737 000030 000746 2$: CMP #30,WAM ;BRANCH IF WRAP AROUND MODE 4
1568 006114 001772 BEQ 1$
1569 006116 052777 000001 172364 EX1: BIS #1,@C1 ;SET GO BIT
1570 006124 005000 1$: CLR R0
1571 006126 012701 000002 MOV #2,R1 ;SET DELAY
1572 006132 032777 100000 172402 EX1A: BIT #100000,@TC ;SEE IF ALPHA
1573 006140 001404 BEQ 1$ ;IF SO: BR
1574 006142 005300 DEC R0
1575 006144 001372 BNE EX1A ;AWAIT ALPHA
1576 006146 005301 DEC R1
1577 006150 001370 BNE EX1A
1578 006152 005737 000602 1$: TST SLVTYP ;BRANCH IF TE16
1579 006156 001407 BEQ EX2
1580 006160 022737 000014 000746 CMP #14,WAM ;BRANCH IF NOT WAM3
1581 006166 001003 BNE EX2
1582 006170 053777 000752 172336 BIS DATC,@MR ;LOAD DATA AND DO CLOCK UP
1583 006176 005077 172364 EX2: CLR @PSW
1584 006202 012701 000400 MOV #400,R1 ;SET NUMBER OF CLKS
1585 006206 032737 000020 000774 BIT #20,UDES ;SEE IF CORE DUMP
1586 006214 001402 BEQ EX3 ;IF NOT: BR
1587 006216 012701 001000 MOV #1000,R1 ;SET # OF CLOCKS WORD COUNT X 4
1588 006222 022737 000014 000746 EX3: CMP #14,WAM ;BRANCH IF WAM 3
1589 006230 001412 BEQ EX5A
1590 006232 032737 002000 000774 BIT #2000,UDES ;BRANCH IF NRZ
1591 006240 001404 BEQ FX5
1592 006242 006301 ASL R1
1593 006244 062701 000246 ADD #246,R1 ;SET TO ALLOW FOR PRE/POSTAMBLE
1594 006250 000402 BR EX5A
1595 006252 062701 000010 EX5: ADD #10,R1 ;ADD CLOCKS FOR CRC AND LRC
1596 006256 022737 000014 000746 EX5A: CMP #14,WAM ;BRANCH IF NOT WAM 3
1597 006264 001065 BNE EX5C
1598 006266 117700 172462 MOVB @WTAD,R0 ;GET DATA CHARACTER
1599 006272 042700 177400 BIC #177400,R0 ;CLEAR UPPER BYTE
1600 006276 005737 001004 TST RDRVF ;BRANCH IF READ FORWARD
1601 006302 001403 BEQ EX5A1
1602 006304 005337 000754 DEC WTAD ;DEC POINTER
1603 006310 000416 BR EX5B
1604 006312 005237 000754 EX5A1: INC WTAD ;INCREMENT DATA PTR
1605 006316 032737 000020 000774 BIT #20,UDES ;BRANCH IF NOT CORE DUMP MODE
1606 006324 001410 BEQ EX5B
1607 006326 005237 000754 INC WTAD ;BUMP POINTER
1608 006332 005777 172416 TST @WTAD ;SEE IF END
1609 006336 001003 BNE EX5B ;IF NOT: BR
1610 006340 162737 000010 000754 SUB #10,WTAD ;RESTORE POINTER
1611 006346 052777 000040 172160 EX5B: BIS #40,@MR ;CLOCK UP (DOWN IF TU77)
1612 006354 017702 172154 MOV @MR,R2 ;READ MR

```


1613	006360	042702	177600		BIC	#177600,R2	:MASK OUT DATA	
1614	006364	000300			SWAB	R0	:POSITION DATA	
1615	006366	022700	177000		CMP	#177000,R0	:SEE IF PATTERN 3	
1616	006372	001404			BEQ	2\$:IF SO: BR	
1617	006374	000240			NOP			
1618	006376	000240			NOP			
1619	006400	052700	000200	1\$:	BIS	#200,R0	:SET ODD PARITY	
1620	006404	050002		2\$:	BIS	R0,R2	:LOAD NEW DATA	
1621	006406	005737	000602		TST	SLVTYP	:BRANCH IF TE16	
1622	006412	0014C7			BEQ	4\$		
1623	006414	005301			DEC	R1	:DECREMENT CLOCK COUNT	
1624	006416	001403			BEQ	3\$		
1625	006420	010277	172110		MOV	R2,@MR	:CLOCK DOWN (UP IF TU77)	
1626	006424	000714			BR	EX5A	:CONTINUE	
1627	006426	000137	011704	3\$:	JMP	EORP	:GO TO END OF RECORD	
1628								
1629	006432	010277	172076	4\$:	MOV	R2,@MR	:DO CLOCK DOWN	
1630	006436	000426			BR	EX5D		
1631								
1632	006440	052777	000040	172066	EX5C:	BIS	#40,@MR	:CLOCK UP (DOWN)
1633	006446	042777	000040	172060		BIC	#40,@MR	:CLOCK DOWN (UP)
1634	006454	017700	172054		MOV	@MR,R0	:GET MR	
1635	006460	000300			SWAB	R0		
1636	006462	032737	000010	000774	BIT	#10,UDES	:SEE IF EVEN PAR	
1637	006470	001405			BEQ	EX5C0	:IF NOT: BR	
1638	006472	010077	172262		MOV	R0,@RDAD		
1639	006476	005237	000760		INC	RDAD		
1640	006502	000402			BR	EX5C1		
1641	006504	110077	172250		EX5C0:	MOVB	R0,@RDAD	:PUT CHAR IN CORE
1642	006510	005237	000760		EX5C1:	INC	RDAD	
1643	006514	000240			EX5D:	NOP		
1644	006516	005301			DEC	R1	:SET IF DONE CLKS	
1645	006520	001256			BNE	EX5A	:IF NOT: BR	
1646	006522	000137	011704		JMP	EORP	:GO TO EOR	

```

1647                                     ;EXECUTE WAM 0*****
1648
1649 006526 012737 006712 000666 EXW2:  MOV    #EXW2H,RTRN    ;SET INTERRUPT RETURN ADDRESS
1650 006534 012701 000200          MOV    #200,R1      ;SET NUMBER OF CLOCKS = FC/2
1651 006540 032737 002000 000774          BIT    #2000,UDES   ;SEE IF PE
1652 006546 001402          BEQ    EXW2A       ;IF NOT: BR
1653 006550 012701 000100          MOV    #100,R1     ;ELSE SET CLKS = FC/4
1654 006554 012702 020236          EXW2A: MOV    #RBUF,R2    ;SET BUFFER ADDRESS
1655 006560 022737 000030 000746          CMP    #30,WAM     ;BRANCH IF NOT WRAP 4
1656 006566 001003          BNE    1$         ;
1657 006570 052777 000010 171722          BIS    #10,@CS     ;SET INHIBIT BUS ADDRESS INCREMENT
1658 006576 012777 000161 171704 1$:      MOV    #161,@C1    ;SET WRITE COMMAND AND GO
1659 006604 005077 171756          CLR    @PSW       ;ALLOW INTERRUPT
1660 006610 032777 000040 171716 EXW2B:  BIT    #40,@MR     ;
1661 006616 001774          BEQ    EXW2B      ;AWAIT CLOCK UP
1662 006620 017722 171710          MOV    @MR,(R2)+  ;GET DATA
1663 006624 032777 000040 171702 EXW2C:  BIT    #40,@MR     ;
1664 006632 001374          BNE    EXW2C      ;AWAIT CLOCK DOWN
1665 006634 017722 171674          MOV    @MR,(R2)+  ;GET DATA
1666 006640 005301          DEC    R1         ;SEE IF DONE ALL
1667 006642 001362          BNE    EXW2B      ;IF NOT: BR
1668 006644 012701 000003          EXW2E:  MOV    #3,R1     ;
1669 006650 005000          CLR    R0         ;SET DELAY
1670 006652 005300          EXW2F:  DEC    R0     ;
1671 006654 001376          BNE    EXW2F      ;
1672 006656 005301          DEC    R1         ;
1673 006660 001374          BNE    EXW2F      ;DELAY
1674 006662 032777 020000 171700          BIT    #20000,@SWR ;SEE IF ERROR PRINT
1675 006670 001004          BNE    EXW2G      ;IF NOT: BR
1676 006672 012704 017170          MOV    #WMSG24,R4 ;
1677 006676 004737 013120          JSR    PC,ITOUT   ;PRINT NO INTERRUPT
1678 006702 005777 171662          EXW2G:  TST    @SWR   ;SEE IF HALT ON ERROR
1679 006706 100001          BPL    EXW2H      ;IF NOT: BR
1680 006710 000000          HALT             ;
1681 006712 000240          EXW2H:  NOP             ;
1682 006714 012701 020236          MOV    #RBUF,R1   ;GET START OF READ BUFFER
1683 006720 012700 000400          MOV    #400,R0    ;SET SIZE
1684 006724 010102          MOV    R1,R2      ;
1685 006726 012203          EXW2J:  MOV    (R2)+,R3 ;
1686 006730 000303          SWAB   R3         ;
1687 006732 032737 000010 000774          BIT    #10,UDES   ;SEE IF EVEN PAR
1688 006740 001402          BEQ    EXW2J0     ;IF NOT: BR
1689 006742 010321          MOV    R3,(R1)+  ;SAVE PAR + DATA
1690 006744 000401          BR     EXW2J1     ;
1691 006746 110321          EXW2J0: MOV    R3,(R1)+  ;ASSEMBLE DATA IN BYTES
1692 006750          EXW2J1: DEC    R0     ;
1693 006752 001365          BNE    EXW2J      ;CONTINUE FOR ALL
1694 006754 032777 000200 171606          BIT    #200,@SWR  ;SEE IF STATUS CHECK
1695 006762 001002          BNE    EXW2K      ;IF NOT: BR
1696 006764 004737 007010          JSR    PC,WSTCK   ;ELSE GO CHECK STATUS
1697 006770 000240          EXW2K:  NOP             ;
1698 006772 032777 000400 171570          BIT    #400,@SWR  ;SEE IF DATA CHECK
1699 007000 001002          BNE    EXW2X      ;IF NOT: BR
1700 007002 004737 007410          JSR    PC,DCHK    ;ELSE GO CHECK DATA
1701 007006 000207          EXW2X:  RTS     PC   ;EXIT

```

```
1702
1703
1704 ;WRAP AROUND STATUS CHECK ROUTINE*****
1705 007010 000240 WSTCK: NOP
1706 007012 005037 000770 CLR SERFL ;CLEAR ERROR FLAG
1707 007016 005037 000616 CLR HDRFL ;CLEAR HEADER FLAG
1708 007022 005737 000602 TST SLVTYP ;BRANCH IF TU77
1709 007026 001004 BNE 10$
1710 007030 022737 015124 000620 CMP #MSLT2,EMADDR ;SEE IF TEST 2
1711 007036 001404 BEQ 2$ ;IF SO: BR
1712 007040 022737 016014 000620 10$: CMP #MSLT15,EMADDR ;SEE IF TEST 15
1713 007046 001012 BNE 5$ ;IF NOT: BR
1714 007050 022737 000003 001000 2$: CMP #3,PATRN ;SEE IF PATTERN 3
1715 007056 001006 BNE 5$ ;IF NOT: BR
1716 007060 052737 140000 001026 4$: BIS #140000,WDS
1717 007066 052737 000100 001030 BIS #100,WER ;SET EXPT PARITY ERROR
1718 007074 012737 017012 000670 5$: MOV #WMSG6,ERADD ;SET CODE=CS1
1719 007102 017702 171402 MOV @C1,R2 ;GET RCVD CS1
1720 007106 042702 140000 BIC #140000,R2 ;MASK OUT SC & TRE BITS
1721 007112 013705 001022 MOV WCS1,R5 ;GET EXPT CS1
1722 007116 042705 140000 BIC #140000,R5 ;MASK OUT SC & TRE BITS
1723 007122 004737 007264 JSR PC,WSTG ;GO CHK
1724 007126 012737 017037 000670 MOV #WMSG6D,ERADD ;SET CODE=CS2
1725 007134 017702 171360 MOV @CS,R2 ;SET RCVD CS2
1726 007140 042702 001000 BIC #1000,R2 ;CLEAR MXF BIT (DON'T CARE BIT)
1727 007144 013705 001024 MOV WCS2,R5 ;GET EXPT CS2
1728 007150 053705 000622 BIS DRVN,R5 ;SET DRIVE NUMBER IN EXPT CS2
1729 007154 004737 007264 JSR PC,WSTG ;GO CHK
1730 007160 012737 017045 000670 MOV #WMSG6E,ERADD ;SET CODE=DS
1731 007166 017702 171330 MOV @DS,R2 ;SET RCVD DS
1732 007172 013705 001026 MOV WDS,R5 ;GET EXPT DS
1733 007176 004737 007264 1$: JSR PC,WSTG ;GO CHK
1734 007202 012737 017052 000670 MOV #WMSG6F,ERADD ;SET CODE=ER
1735 007210 017702 171310 MOV @ER,R2 ;GET RCVD ER
1736 007214 000240 NOP
1737 007216 000240 NOP
1738 007220 022737 016235 000620 CMP #MSLT20,EMADDR ;SEE IF TEST 20
1739 007226 001002 BNE 12$ ;IF NOT: BR
1740 007230 042702 001000 BIC #1000,R2 ;MASK FCE
1741 007234 013705 001030 12$: MOV WER,R5 ;GET EXPT ER
1742 007240 004737 007264 JSR PC,WSTG ;GO CHK
1743 007244 005737 000770 TST SERFL ;SEE IF ANY ERRORS
1744 007250 001456 BEQ WSTX ;IF NOT: BR
1745 007252 005777 171312 TST @SWR ;SEE IF HALT ON ERROR
1746 007256 100053 BPL WSTX ;IF NOT: BR
1747 007260 000000 HALT
1748 007262 000451 BR WSTX ;CONTINUE
1749 007264 000240 WSTG: NOP
1750 007266 020205 CMP R2,R5 ;SEE IF EXPT=RCVD
1751 007270 001446 BEQ WSTX ;IF SO: BR
1752 007272 005237 000770 INC SERFL ;SET ERROR FLAG
1753 007276 032777 020000 171264 BIT #20000,@SWR ;SEE IF PRINT ERRORS
1754 007304 001040 BNE WSTX ;IF NOT: BR
1755 007306 005737 000616 TST HDRFL ;SEE IF DONE HEADER
1756 007312 001010 BNE WSTGO ;IF SO: BR
1757 007314 013704 000620 MOV EMADDR,R4
```

1758	007320	004737	013120		JSR	PC,TTOUT	:PRINT TEST HEADER
1759	007324	012704	017154		MOV	#WMSG23,R4	
1760	007330	004737	013120		JSR	PC,TTOUT	:PRINT STATUS TAG
1761	007334	012737	000001	000616	MOV	#1,HDRFL	:SET HEADER FLAG
1762	007342	013704	000670		MOV	ERADD,R4	
1763	007346	004737	013120		JSR	PC,TTOUT	:PRINT CODE
1764	007352	012704	014376		MOV	#MSG12,R4	
1765	007356	004737	013120		JSR	PC,TTOUT	:PRINT EXPT TAG
1766	007362	010503			MOV	R5,R3	
1767	007364	004737	013246		JSR	PC,OCTP	:PRINT EXPT STATUS
1768	007370	012704	014405		MOV	#MSG13,R4	
1769	007374	004737	013120		JSR	PC,TTOUT	:PRINT RCVD TAG
1770	007400	010203			MOV	R2,R3	
1771	007402	004737	013246		JSR	PC,OCTP	:PRINT RCVD STATUS
1772	007406	000207		WSTX:	RTS	PC	:RETURN
1773							

```

1774
1775
1776
1777 007410 000240          DCHK:  NOP
1778 007412 005737 001020      TST    CRCFLG          ;DO NOT DO A DATA CHACK
1779 007416 001402          BEQ    2$              ;IF CRC CORRECTION IN PROGRESS
1780 007420 000137 010250      1$:   JMP    DCHKX1
1781 007424 005737 001016      2$:   TST    DCHKFL          ;BRANCH IF DATA IS NOT TO BE CHECKED
1782 007430 001373          BNE    1$
1783 007432 005037 000616      CLR    HDRFL          ;CLEAR HEADER FLAG
1784 007436 005037 000764      CLR    DERFL          ;CLEAR DATA ERROR FLAG
1785 007442 005037 000772      CLR    CRCNT          ;CLEAR CHAR CNTR
1786 007446 032737 000010 000774  BIT    #10,UDES        ;SEE IF EVEN PARITY
1787 007454 001402          BEQ    DCHKA0          ;IF NOT: BR
1788 007456 000137 010272      JMP    DCHKE          ;ELSE GO CHECK EVEN
1789 007462 022737 000010 000746 DCHKA0: CMP    #10,WAM        ;SEE IF WAM 1
1790 007470 001006          BNE    DCHKA          ;IF NOT: BR
1791 007472 032737 002000 000774  BIT    #2000,UDES      ;SEE IF PE
1792 007500 001402          BEQ    DCHKA          ;IF NOT: BR
1793 007502 000137 010774      JMP    PRCHK          ;GO CHK DATA
1794 007506 012700 177400      DCHKA: MOV    #-400,R0   ;SET NUMBER OF CHARACTERS
1795 007512 022737 000012 000746  CMP    #12,WAM
1796 007520 001006          BNE    DCHKA1          ;IF NOT WRAP 2: BR
1797 007522 032737 000020 000774  BIT    #20,UDES
1798 007530 001402          BEQ    DCHKA1          ;IF NOT CORE DUMP: BR
1799 007532 012700 177000      MOV    #-1000,R0
1800 007536 022737 000030 000746 DCHKA1: CMP    #30,WAM   ;BRANCH IF WRAP 4
1801 007544 001404          BEQ    1$
1802 007546 022737 000006 000746  CMP    #6,WAM          ;SEE IF WRAP 0
1803 007554 001007          BNE    DCHKA2          ;IF NOT: BR
1804 007556 012700 177744      1$:   MOV    #-34,R0      ;SET NUMBER OF CHARACTERS READ
1805 007562 012701 017634      MOV    #WBUF+10,R1    ;SET POINTER
1806 007566 005037 000764      CLR    DERFL          ;CLEAR DATA ERROR FLAG
1807 007572 000431          BR     DCHKB0
1808 007574 022737 000012 000746 DCHKA2: CMP    #12,WAM   ;SEE IF WRAP 2
1809 007602 001021          BNE    DCHKB          ;IF NOT: BR
1810 007604 032737 002000 000774  BIT    #2000,UDES      ;SEE IF PE
1811 007612 001415          BEQ    DCHKB          ;IF NOT: BR
1812 007614 012700 177653      MOV    #-125,R0       ;POINT TO START OF DATA
1813 007620 012737 000001 000762  MOV    #1,W2FLG        ;SET WRAP 2 FLAG
1814 007626 004737 007646      JSR    PC,DCHKB        ;GO CHECK DATA
1815 007632 004737 011274      JSR    PC,W1DCHK       ;GO CHECK WRAP 1 DATA
1816 007636 005037 000762      CLR    W2FLG
1817 007642 000137 010232      JMP    DCHKX
1818 007646 005037 000764      DCHKB: CLR    DERFL
1819 007652 012701 017624      MOV    #WBUF,R1       ;SET GOOD POINTER
1820 007656 012702 020236      DCHKB0: MOV    #RBUF,R2  ;SET READ POINTER
1821 007662 032737 000020 000774  BIT    #20,UDES        ;SEE IF CORE DUMP
1822 007670 001416          BEQ    DCHKO          ;IF NOT: BR
1823 007672 022737 000012 000746  CMP    #12,WAM        ;SEE IF WAM 2
1824 007700 001011          BNE    DCHKD          ;IF NOT: BR
1825 007702 005737 001000      TST    PATRN          ;SEE IF PATTERN 0
1826 007706 001003          BNE    DCHKC          ;IF NOT: BR
1827 007710 012701 001054      MOV    #WCDP0,R1      ;SET CORE DUMP PATTERN 0
1828 007714 000404          BR     DCHKO          ;GO CHECK DATA
1829 007716 012701 001042      DCHKC: MOV    #WCDP2,R1 ;SET CORE DUMP WRITE PATTERN 2

```

1830	007722	000401				BR	DCHK0		:GO CHECK DATA
1831	007724	000240				DCHKD: NOP			
1832	007726	121112				DCHK0: CMPS	(R1),(R2)		:SEE IF DATA OK
1833	007730	001466				BEQ	DCHK2		:IF SO: BR
1834	007732	032777	020000	170630		BIT	#20000,@SWR		:SEE IF PRINT ERRORS
1835	007740	001062				BNE	DCHK2		:IF NOT: BR
1836	007742	005737	000616			TST	HDRFL		:SEE IF DONE HEADER
1837	007746	001004				BNE	DCHK1		:IF SO: BR
1838	007750	013704	000620			MOV	EMADDR,R4		
1839	007754	004737	013120			JSR	PC,TTOUT		:PRINT HEADER
1840	007760	005737	000764			DCHK1: TST	DERFL		:SEE IF FIRST ERROR
1841	007764	001014				BNE	DCHK1A		:IF NOT: BR
1842	007766	012704	017122			MOV	#WMSG16,R4		
1843	007772	004737	013120			JSR	PC,TTOUT		:PRINT DATA ERROR TAG
1844	007776	012704	017347			MOV	#WMSG32,R4		
1845	010002	004737	013120			JSR	PC,TTOUT		:PRINT PATRN TAG
1846	010006	013703	001000			MOV	PATRN,R3		
1847	010012	004737	013246			JSR	PC,OCIP		:PRINT PATTERN NUMBER
1848	010016	012737	000001	000616		DCHK1A: MOV	#1,HDRFL		:SET HEADER FLAG
1849	010024	012737	000001	000764		MOV	#1,DERFL		:SET DATA ERROR FLAG
1850	010032	012704	017146			MOV	#WMSG21,R4		
1851	010036	004737	013120			JSR	PC,TTOUT		:PRINT CHARACTER NUMBER TAG
1852	010042	013703	000772			MOV	CRCNT,R3		
1853	010046	004737	013246			JSR	PC,OCIP		:PRINT CHARACTER NUMBER
1854	010052	012704	017134			MOV	#WMSG17,R4		
1855	010056	004737	013120			JSR	PC,TTOUT		:PRINT GOOD TAG
1856	010062	111103				MOVB	(R1),R3		
1857	010064	004737	013474			JSR	PC,DOUT		:PRINT GOOD DATA
1858	010070	012704	017141			MOV	#WMSG20,R4		
1859	010074	004737	013120			JSR	PC,TTOUT		:PRINT BAD TAG
1860	010100	111203				MOVB	(R2),R3		
1861	010102	004737	013474			JSR	PC,DOUT		:PRINT BAD DATA
1862	010106	005737	000762			DCHK2: TST	W2FLG		:SEE IF WRAP 2 NRZ
1863	010112	001020				BNE	DCHK2B		:IF SO: BR
1864	010114	005201				INC	R1		:BUMP POINTER
1865	010116	032737	000020	000774		BIT	#20,UDES		:SEE IF CORE DUMP
1866	010124	001413				BEQ	DCHK2B		:IF NOT: BR
1867	010126	022737	000012	000746		CMF	#12,WAM		:SEE IF WAM 2
1868	010134	001006				BNE	DCHK2A		:IF NOT: BR
1869	010136	005201				INC	R1		:BUMP POINTER
1870	010140	005711				TST	(R1)		:SEE IF END OF PATTERN
1871	010142	001004				BNE	DCHK2B		:IF NOT: BR
1872	010144	162701	000010			SUB	#10,R1		:RESET POINTER TO START OF PATTERN
1873	010150	000401				BR	DCHK2B		:CONTINUE CHECK
1874	010152	000240				DCHK2A: NOP			
1875	010154	005202				DCHK2B: INC	R2		
1876	010156	022737	000030	000746		CMF	#30,WAM		:BRANCH IF WAM 4
1877	010164	001404				BEQ	1\$		
1878	010166	022737	000006	000746		CMF	#6,WAM		:SEE IF WAM 0
1879	010174	001002				BNE	DCHK3		:IF NOT: BR
1880	010176	062701	000010			1\$: ADD	#10,R1		:BUMP POINTER
1881	010202	005237	000772			DCHK3: INC	CRCNT		:BUMP CHAR CNTR
1882	010206	032777	000400	170354		BIT	#400,@SWR		:SEE IF CONT DATA CHK
1883	010214	001006				BNE	DCHKX		:IF NOT: BR
1884	010216	005200				INC	R2		:SEE IF DONE
1885	010220	001242				BNE	DCHK0		:IF NOT: BR

1886	010222	005737	000762		TST	W2FLG	
1887	010226	001401			BEQ	DCHKX	
1888	010230	000207			RTS	PC	
1889	010232	005777	170332	DCHKX:	TST	@SWR	:SEE IF HALT ON ERROR
1890	010236	100004			BPL	DCHKX1	:IF NOT: BR
1891	010240	005737	000764		TST	DERFL	:SEE IF DATA ERROR OCCURED
1892	010244	001401			BEQ	DCHKX1	:IF NOT: BR
1893	010246	000000			HALT		
1894	010250	005037	000772	DCHKX1:	CLR	CRCNT	:CLEAR CHAR CNTR
1895	010254	005037	000616		CLR	HDRFL	:CLEAR HEADER FLAG
1896	010260	005037	000764		CLR	DERFL	:CLEAR DATA ERROR FLAG
1897	010264	005037	000766		CLR	PREFL	:CLEAR PREAMBLE FLAG
1898	010270	000207			RTS	PC	:RETURN

```

1899
1900
1901                                     :EVEN PARITY DATA CHECK*****
1902 010272 000240 DCHKE: NOP
1903 010274 022737 000006 000746 CMP #6,WAM ;SEE IF WRAP 0
1904 010302 001005 BNE 1$ ;IF NOT: BR
1905 010304 012700 177744 MOV #-34,R0 ;SET NUMBER OF CHARACTERS READ
1906 010310 012701 017634 MOV #WBUF+10,R1 ;SET POINTER
1907 010314 000404 BR 2$
1908 010316 012700 177400 1$: MOV #-400,R0 ;SET NUMBER OF CHARACTERS
1909 010322 012701 017624 MOV #WBUF,R1 ;R1=START OF WRITE BUFFER
1910 010326 012702 020236 2$: MOV #RBUF,R2 ;R2=START OF READ BUFFER
1911 010332 111105 DCKE0: MOVB (R1),R5 ;GET EXPT DATA
1912 010334 005003 CLR R3
1913 010336 012704 000010 MOV #10,R4 ;SET NUMBER OF BITS
1914 010342 032705 000001 DCKE1: BIT #1,R5 ;SEE IF ONE BIT
1915 010346 001401 BEQ DCKE2 ;IF NOT: BR
1916 010350 005203 INC R3 ;COUNT ONE BITS FOR PARITY CHECK
1917 010352 005304 DCKE2: DEC R4 ;SEE IF DONE
1918 010354 001402 BEQ DCKE3 ;IF SO: BR
1919 010356 006005 ROR R5 ;POINT TO NEXT BIT
1920 010360 000770 BR DCKE1
1921 010362 000240 DCKE3: NOP
1922 010364 111105 MOVB (R1),R5 ;GET EXPT DATA
1923 010366 042705 177400 BIC #177400,R5 ;MASK DATA FIELD
1924 010372 005703 TST R3
1925 010374 001003 BNE DCKE4 ;IF NO ONE BITS SET: BR
1926 010376 012705 100020 MOV #100020,R5
1927 010402 000405 BR DCKE5
1928 010404 032703 000001 DCKE4: BIT #1,R3 ;SEE IF ODD NUMBER OF ONE BITS
1929 010410 001402 BEQ DCKE5 ;IF NOT: BR
1930 010412 052705 100000 BIS #100000,R5 ;SET EVEN PARITY BIT=1
1931 010416 042712 077400 DCKE5: BIC #77400,(R2) ;MASK DATA FIELD
1932 010422 020512 CMP R5,(R2) ;SEE IF DATA + PARITY GOOD
1933 010424 001477 BEQ DCKE10 ;IF SO: BR
1934 010426 032777 020000 170134 BIT #20000,@SWR ;SEE IF ERROR PRINT
1935 010434 001073 BNE DCKE10 ;IF NOT: BR
1936 010436 005737 000616 TST HDRFL ;SEE IF DONE HEADER
1937 010442 001004 BNE DCKE6 ;IF SO: BR
1938 010444 013704 000620 MOV EMADDR,R4
1939 010450 004737 013120 JSR PC,TTOUT ;PRINT HEADER
1940 010454 005737 000764 DCKE6: TST DERFL ;SEE IF FIRST BAD CHAR
1941 010460 001014 BNE DCKE7 ;IF NOT: BR
1942 010462 012704 017122 MOV #WMSG16,R4
1943 010466 004737 013120 JSR PC,TTOUT ;PRINT BAD DATA TAG
1944 010472 012704 017347 MOV #WMSG32,R4
1945 010476 004737 013120 JSR PC,TTOUT ;PRINT PATTERN TAG
1946 010502 013703 001000 MOV PATRN,R3
1947 010506 004737 013246 JSR PC,OCTP ;PRINT PATTERN NUMBER
1948 010512 000240 DCKE7: NOP
1949 010514 012737 000001 000764 MOV #1,DERFL ;SET DATA ERROR FLAG
1950 010522 012737 000001 000616 MOV #1,HDRFL ;SET HEADER FLAG
1951 010530 012704 017146 MOV #WMSG21,R4
1952 010534 004737 013120 JSR PC,TTOUT ;PRINT CHAR NUMBER TAG
1953 010540 013703 000772 MOV CNT,R3
1954 010544 004737 013246 JSR PC,OCTP ;PRINT CHAR NUMBER
  
```



```

1955 010550 012704 017134      MOV      #WMSG17,R4
1956 010554 004737 013120      JSR      PC,TTOUT      :PRINT GOOD DATA TAG
1957 010560 110503                MOV      R5,R3
1958 010562 004737 013474      JSR      PC,DOUT       :PRINT EXPT DATA
1959 010566 010503                MOV      R5,R3
1960 010570 004737 010700      JSR      PC,DCKEP      :GO PRINT PARITY BIT
1961 010574 000240                NOP
1962 010576 012704 017141      MOV      #WMSG20,R4
1963 010602 004737 013120      JSR      PC,TTOUT      :PRINT BAD TAG
1964 010606 111203                MOV      (R2),R3
1965 010610 004737 013474      JSR      PC,DOUT       :PRINT BAD DATA
1966 010614 011203                MOV      (R2),R3
1967 010616 004737 010700      JSR      PC,DCKEP      :GO PRINT PARITY BIT
1968 010622 000240                NOP
1969 010624 005201                DCKE10: INC      R1
1970 010626 022737 000006 000746  CMP      #6,WAM        :SEE IF WRAP 0
1971 010634 001002                BNE      1$           :IF NOT: BR
1972 010636 062701 000010                ADD      #10,R1       :BUMP POINTER
1973 010642 005722                1$:  TST      (R2)+     :BUMP POINTERS
1974 010644 005237 000772                INC      CRCNT        :BUMP CHAR CNTR
1975 010650 032777 000400 167712  BIT      #400,@SWR    :SEE IF CONTINLF DATA CHECK
1976 010656 001402                BEQ      DCKE11      :IF SO: BR
1977 010660 000137 010232                JMP      DCHKX        :GO TO END OF DATA CHECK
1978 010664 005200                DCKE11: INC      R0     :SEE IF DONE
1979 010666 001402                BEQ      DCKE12      :IF SO: BR
1980 010670 000137 010332                JMP      DCKE0        :ELSE CONTINUE
1981 010674 000137 010232                DCKE12: JMP      DCHKX  :GO TO END OF DATA CHECK
1982 010700 000240                DCKEP:  NOP
1983 010702 012737 000240 000612  MOV      #240,TOB
1984 010710 004737 013220                JSR      PC,TOG       :SPACE
1985 010714 012737 000260 000612  MOV      #260,TOB     :SET PAR=0
1986 010722 005703                TST      R3           :SEE IF PARITY REALLY=0
1987 010724 100002                BPL      DCKEPO      :IF SO: BR
1988 010726 005237 000612                INC      TOB          :ELSE SET TO 1
1989 010732 004737 013220                DCKEPO: JSR      PC,TOG :PRINT PARITY BIT
1990 010736 000207                RTS      PC           :RETURN
1991

```

```

1992
1993
1994
1995 010740 012700 000051      PDCHK:  MOV    #51,R0          ;SET SIZE OF POSTAMBLE
1996 010744 012701 017502      MOV    #POST,R1         ;SET POINTER TO POSTAMBLE
1997 010750 005037 000616      CLR    HDRFL            ;CLEAR HEADER FLAG
1998 010754 005037 000772      CLR    CRCNT            ;CLEAR CHAR CNTR
1999 010760 005037 000764      CLR    DERFL            ;CLEAR DATA ERROR FLAG
2000 010764 000240
2001 010766 000240
2002 010770 000137 011012      NOP
2003                                NOP
2004                                JMP    PD0              ;GO CHECK POSTAMBLE
2004 010774 012700 000051      PRCHK:  MOV    #51,R0          ;SET SIZE OF PREAMBLE
2005 011000 012701 017360      MOV    #PRE,R1          ;SET POINTER TO PREAMBLE
2006 011004 012702 020236      MOV    #RBUF,R2         ;SET POINTER TO START OF READ BUFFER
2007 011010 022122
2008 011012 121112      PD0:    CMPB   (R1),(R2)+      ;BUMP ADDRESS POINTERS
2009 011014 001004
2010 011016 126162 000001 000001  CMPB   1(R1),1(R2)      ;CHECK DATA
2011 011024 001477
2012 011026 032777 020000 167534  PD1:    BNE    PD1              ;IF NOT GOOD: BR
2013 011034 001073
2014 011036 005737 000616      BEQ    PD5              ;COMPARE COMPLIMENT BYTE
2015 011042 001020
2016 011044 013704 000620
2017 011050 004737 013120
2018 011054 005737 000766      BIT    #20000,@SWR      ;IF GOOD: BR
2019 011060 001403
2020 011062 012704 017273      BNE    PD5              ;SEE IF PRINT INHIBIT
2021 011066 000402
2022 011070 012704 017255      TST   HDRFL            ;IF SO: BR
2023 011074 004737 013120      BNE    PD4              ;SEE IF DONE HEADER
2024 011100 005237 000616
2025 011104 012704 017146
2026 011110 004737 013120
2027 011114 013703 000772      MOV   EMADDR,R4        ;IF SO: BR
2028 011120 004737 013246      JSR   PC,TTOUT          ;PRINT TEST HEADER
2029 011124 012704 017134      TST   PRFL              ;SEE IF PREAMBLE CHECK
2030 011130 004737 013120      BEQ   PD2              ;IF NOT: BR
2031 011134 116103 000001
2032 011140 004737 013474      MOV   #WMSG29,R4       ;SET POSTAMBLE HEADER
2033 011144 012737 000240 000612  BR    PD3              ;PRINT PREAMBLE HEADER
2034 011152 004737 013220
2035 011156 111103
2036 011160 004737 013474      JSR   PC,TTOUT          ;PRINT HEADER
2037 011164 012704 017141
2038 011170 004737 013120      INC   HDRFL
2039 011174 116203 000001
2040 011200 004737 013474      MOV   #WMSG21,R4       ;PRINT CHAR NUMBER TAG
2041 011204 012737 000240 000612  JSR   PC,TTOUT          ;PRINT CHAR NUMBER
2042 011212 004737 013220
2043 011216 111203
2044 011220 004737 013474      MOV   #WMSG17,R4       ;PRINT CHAR NUMBER
2045 011224 022122
2046 011226 005237 000772      JSR   PC,TTOUT          ;PRINT GOOD TAG
2047 011232 005300
                                MOVB  1(R1),R3           ;PRINT GOOD CHAR
                                JSR   PC,DOUT              ;PRINT COMPLEMENT
                                MOV   #240,TOB            ;PRINT BAD TAG
                                JSR   PC,TOG              ;PRINT BAD CHAR
                                MOVB  (R1),R3
                                JSR   PC,DOUT              ;PRINT COMPLEMENT
                                MOV   #WMSG20,R4
                                JSR   PC,TTOUT          ;BUMP ADDRESS POINTERS
                                MOVB  1(R2),R3
                                JSR   PC,DOUT              ;BUMP CHAR NUMBER
                                JSR   PC,TOG              ;SEE IF DONE
                                MOV   #240,TOB
                                JSR   PC,TOG
                                MOVB  (R2),R3
                                JSR   PC,DOUT
                                CMP    (R1)+,(R2)+
                                INC    CRCNT
                                DEC    R0
  
```

2048	011234	001266		BNE	PDO	:IF NOT: BR
2049	011236	005737	000766	TST	PREFL	:SEE IF PREAMBLE
2050	011242	001402		BEQ	PD6	:IF SO: BR
2051	011244	000137	010232	JMP	DCHKX	:GO TO EXIT ROUTINE
2052	011250	005237	000766	INC	PREFL	:SET PREAMBLE FLAG
2053	011254	005037	000616	CLR	HDRFL	:CLEAR HEADER FLAG
2054	011260	005037	000772	CLR	CRCNT	:CLEAR CHAR CNTR
2055	011264	005037	000764	CLR	DERFL	:CLEAR DATA ERROR FLAG
2056	011270	000137	011274	JMP	WIDCHK	:GO CHECK WRAP 1 DATA
2057						

PD6:

```

2058
2059
2060 ;WAM 1 PE DATA CHECK*****
2061 011274 012700 177400 W1DCHK: MOV #-400,R0 ;SET NUMBER OF CHAR TO CHECK
2062 011300 012701 017624 MOV #WBUFF,R1 ;SET WRITE DATA POINTER
2063 011304 012702 020236 MOV #RBUFF,R2 ;SET READ DATA POINTER
2064 011310 062702 000124 ADD #124,R2 ;POINT TO START OF DATA
2065 011314 005737 000762 TST W2FLG ;SEE IF WRAP 2
2066 011320 001401 BEQ W1D0 ;IF NOT WAM 2: BR
2067 011322 005302 DEC R2 ;RESET PCINTER
2068 011324 111105 W1D0: MOV (R1),R5
2069 011326 120512 CMPB R5,(R2) ;CHECK DATA
2070 011330 001007 BNE W1D1 ;IF NOT GOOD:BR
2071 011332 005737 000762 TST W2FLG ;SEE IF WRAP 2
2072 011336 001001 BNE W1DOA ;IF SO: BR
2073 011340 105105 COMB R5 ;COMPLIMENT EXPT DATA
2074 011342 120562 000001 W1DOA: CMPB R5,1(R2) ;CHECK COMPLIMENT DATA
2075 011346 001510 BEQ W1D3 ;IF GOOD: BR
2076 011350 032777 020000 167212 W1D1: BIT #20000,@SWR ;SEE IF PRINT INHIBIT
2077 011356 001104 BNE W1D3 ;IF SO: BR
2078 011360 005737 000616 TST HDRFL ;SEE IF DONE HEADER
2079 011364 001020 BNE W1D2 ;IF SO: BR
2080 011366 013704 000620 MOV EMADDR,R4
2081 011372 004737 013120 JSR PC,TTOUT ;PRINT TEST HEADER
2082 011376 012704 017122 MOV #WMSG16,R4
2083 011402 004737 013120 JSR PC,TTOUT ;PRINT BAD DATA TAG
2084 011406 012704 017347 MOV #WMSG32,R4
2085 011412 004737 013120 JSR PC,TTOUT ;PRINT PATRN TAG
2086 011416 013703 001000 MOV PATRN,R3
2087 011422 004737 013246 JSR PC,OCIP ;PRINT PATTERN NUMBER
2088 011426 012737 000001 000616 W1D2: MOV #1,HDRFL ;SET HEADER FLAG
2089 011434 012704 017146 MOV #WMSG21,R4
2090 011440 004737 013120 JSR PC,TTOUT ;PRINT CHAR NUMBER TAG
2091 011444 013703 000772 MOV CRCNT,R3
2092 011450 004737 013246 JSR PC,OCIP ;PRINT CHAR NUMBER
2093 011454 012704 017134 MOV #WMSG17,R4
2094 011460 004737 013120 JSR PC,TTOUT ;PRNT GOOD TAG
2095 011464 111105 MOV (R1),R5
2096 011466 110503 MOV R5,R3 ;GET GOOD CHAR
2097 011470 005737 000762 TST W2FLG ;SEE IF WRAP 2
2098 011474 001001 BNE W1D2A ;IF SO: BR
2099 011476 105103 COMB R3 ;ELSE COMPLIMENT CHAR
2100 011500 004737 013474 000612 W1D2A: JSR PC,DOUT ;PRINT CHARACTER
2101 011504 012737 000240 MOV #240,TOB
2102 011512 004737 013220 JSR PC,TOG ;SPACE
2103 011516 110503 MOV R5,R3
2104 011520 004737 013474 JSR PC,DOUT ;PRINT CHAR
2105 011524 012704 017141 MOV #WMSG20,R4
2106 011530 004737 013120 JSR PC,TTOUT ;PRINT BAD TAG
2107 011534 116203 000001 MOVB 1(R2),R3
2108 011540 004737 013474 JSR PC,DOUT ;PRINT BAD CHAR
2109 011544 012737 000240 000612 MOV #240,TOB
2110 011552 004737 013220 JSR PC,TOG ;SPACE
2111 011556 111203 MOV (R2),R3
2112 011560 004737 013474 JSR PC,DOUT ;PRINT CHAR
2113 011564 005237 000764 INC DERFL ;SET DATA ERROR FLAG
  
```

```
2114 011570 122122          W1D3:  CMPB   (R1)+,(R2)+  ;BUMP ADDRESS
2115 011572 105722          TSTB   (R2)+             ;BUMP ADDRESS
2116 011574 005237 000772  INC    CRCNT             ;BUMP CHAR CNTR
2117 011600 000406          BR     W1D5
2118 011602 005737 000762  W1D4:  TST    W2FLG          ;SEE IF WRAP 2
2119 011606 001401          BEQ    W1D4A             ;IF NOT: BR
2120 011610 000207          RTS    PC                ;ELSE RETURN
2121 011612 000137 010740  W1D4A: JMP    PSCHK            ;GO CHECK POSTAMBLE
2122 011616 005200          W1D5:  INC    R0
2123 011620 001770          BEQ    W1D4
2124 011622 000137 011324  JMP    W1D0
2125
2126                          ;PREAMBLE/POSTAMBLE GENERATE SUBROUTINE*****
2127
2128 011626 000240          PPGEN: NOP
2129 011630 012700 000050  MOV    #50,R0            ;SET SIZE OF PREAMBLE
2130 011634 012701 017360  MOV    #PRE,R1
2131 011640 005721          TST    (R1)+             ;SET ADDRESS OF PRE
2132 011642 012721 177400  1$:   MOV    #177400,(R1)+   ;FILL TABLE
2133 011646 005300          DFC    R0                ;SEE IF DONE
2134 011650 001374          BNE    1$                ;IF NOT: BR
2135 011652 012701 017502  MOV    #POST,R1         ;SET ADDRESS OF POST
2136 011656 012700 000050  MOV    #50,R0            ;SET SIZE OF POST
2137 011662 012721 000377  MOV    #377,(R1)+       ;SET SYNC CHAR
2138 011666 012721 177400  2$:   MOV    #177400,(R1)+   ;FILL TABLE
2139 011672 005300          DEC    R0                ;SEE IF DONE
2140 011674 001374          BNE    2$                ;IF NOT: BR
2141 011676 000207          RTS    PC                ;RETURN
```

```
2142
2143
2144 ;END OF RECORD FORCE SUBROUTINE*****
2145 011700 005237 000674 EORPA: INC TEMP2 ;SET WRAP FLAG
2146 011704 017700 166624 EORP: MOV @MR,R0 ;GET MAINT REG
2147 011710 042700 000036 BIC #36,R0 ;CLEAR CURRENT OP CODE
2148 011714 052700 000024 BIS #24,R0 ;SET EOR CLEAR OP CODE
2149 011720 010077 166610 MOV R0,@MR ;DO EOR
2150 011724 042777 000037 166602 BIC #37,@MR ;CLEAR EOR AND MM
2151 011732 005000 CLR R0
2152 011734 012701 000002 MOV #2,R1
2153 011740 032777 000001 166542 EORP1: BIT #1,@C1 ;SEE IF GO GONE
2154 011746 001427 BEQ EORP2 ;IF SO: BR
2155 011750 005300 DEC R0
2156 011752 001372 BNE EORP1 ;AWAIT GO RESET
2157 011754 005301 DEC R1
2158 011756 001370 BNE FORP1
2159 011760 032777 020000 166602 BIT #20000,@SWR ;SEE IF ERROR PRINT INHIBIT
2160 011766 001017 BNE EORP2 ;IF SO: BR
2161 011770 005737 000616 TST HDRFL ;SEE IF DONE HEADER
2162 011774 001004 BNE EORP1A ;IF SO: BR
2163 011776 013704 000620 MOV EMADDR,R4
2164 012002 004737 013120 JSR PC,TTOUT ;PRINT HEADER
2165 012006 012704 017312 EORP1A: MOV #WMSG31,R4
2166 012012 004737 013120 JSR PC,TTOUT ;PRINT EOR GO BIT ERROR
2167 012016 005777 166546 TST @SWR ;SEE IF HALT ON ERROR
2168 012022 100001 BPL EORP2 ;IF NOT: BR
2169 012024 000000 HALT
2170 012026 000240 FORP2: NOP
2171 012030 005737 000674 TST TEMP2 ;SEE IF WAM
2172 012034 001024 BNE EORPX ;IF NOT: BR
2173 012036 022737 000014 000746 CMP #14,WAM ;BRANCH IF NOT WRAP 3
2174 012044 001003 BNE 2$
2175 012046 105777 166436 1$: TSTB @C1 ;WAIT FOR READY
2176 012052 100375 BPL 1$
2177 012054 032777 000200 166506 2$: BIT #200,@SWR ;SEE IF STATUS CHECK
2178 012062 001002 BNE EORP3 ;IF NOT: BR
2179 012064 004737 007010 JSR PC,WSTCK ;ELSE GO CHECK STATUS
2180 012070 000240 EORP3: NOP
2181 012072 032777 000400 166470 BIT #400,@SWR ;SEE IF DATA CHECK
2182 012100 001002 BNE EORPX ;IF NOT: BR
2183 012102 004737 007410 JSR PC,DCHK ;ELSE GO CHECK DATA
2184 012106 000240 EORPX: NOP
2185 012110 005037 000674 CLR TEMP2 ;CLEAR FLAG
2186 012114 000207 RTS PC ;RETURN
2187
```

```

2188
2189
2190
2191 012116 000240
2192 012120 032777 040000 166442 SCOPE: NOP
2193 012126 001001 BIT #40000,@SWR ;SEE IF LOOP ON ERROR
2194 012130 000207 BNE 1$ ;IF SO: BR
2195 012132 000240 RTS PC ;ELSE EXIT
2196 012134 005726 1$: NOP
2197 012136 000240 TST (SP)+ ;RESET STACK
2198 012140 000240 NOP
2199 012142 000177 166542 JMP @SCOLP ;LOOP ON ERROR
2200
2201 ;TEST ITERATION SUBROUTINE*****
2202
2203 012146 032777 004000 166474 ITER: BIT #4000,@SWR ;SEE IF ITERATIONS
2204 012154 001403 BEQ 2$ ;IF SO: BR
2205 012156 005037 000700 1$: CLR ITCNT ;CLEAR ITERATION COUNTER
2206 012162 000207 RTS PC ;ELSE EXIT
2207 012164 005737 001014 2$: TST @MPCNTR ;DO SINGLE SUBTEST ITERATION
2208 012170 001772 BEQ 1$ ;ON FIRST PASS
2209 012172 005237 000700 INC ITCNT ;BUMP COUNTER
2210 012176 023737 000700 000604 CMP ITCNT,ITAMT ;SEE IF DONE ALL
2211 012204 001764 BEQ 1$ ;IF SO: BR
2212 012206 005726 TST (SP)+ ;RESET STACK
2213 012210 017700 166476 MOV @ITRLP,R0 ;SET ITERATION POINTER
2214 012214 000110 JMP (R0) ;GO ITERATE
2215
    
```

```

2216
2217
2218 ;INITIALIZE SUBROUTINE*****
2219 012216 012777 000040 166274 INIT: MOV #40,@CS ;INIT
2220 012224 013777 000622 166266 MOV DRVN,@CS ;SELECT DRIVE
2221 012232 013777 000662 166302 MOV SLVN,@TC ;SELECT SLAVE
2222 012240 013746 000774 MOV UDES,-(SP) ;GET TEST'S UNIT DESCRIPTION
2223 012244 042716 174377 BIC #174377,(SP) ;CLEAR ALL BUT DENSITY SELECT BITS
2224 012250 022726 001400 CMP #1400,(SP)+ ;BRANCH IF NOT NRZ (800 BPI)
2225 012254 001005 BNE 1$
2226 012256 032777 000040 166236 BIT #40,ADS ;BRANCH IF SLAVE IS IN NRZ MODE
2227 012264 001420 BEQ 4$ ;(PES = 0)
2228 012266 000404 BR 2$ ;GO CHANGE DENSITY
2229 012270 032777 000040 166224 1$: BIT #40,ADS ;BRANCH IF SLAVE IS IN PE MODE
2230 012276 001013 BNE 4$ ;(PES = 1)
2231 012300 012777 000007 166202 2$: MOV #7,@C1 ;REWIND SLAVE
2232 012306 032777 000200 166206 20$: BIT #200,ADS ;WAIT FOR READY
2233 012314 001774 BEQ 20$
2234 012316 032777 020000 166176 3$: BIT #20000,ADS ;LOOP UNTIL REWIND IS COMPLETE
2235 012324 001374 BNE 3$ ;(PIP = 0)
2236 012326 053777 000774 166206 4$: BIS UDES,@TC ;LOAD UNIT DESCRIPTION
2237 012334 032777 000002 166160 BIT #2,ADS ;BRANCH IF NOT AT BOT
2238 012342 001407 BEQ 6$
2239 012344 012777 000025 166136 MOV #25,@C1 ;ERASE TO GET OFF BOT
2240 012352 032777 000200 166142 5$: BIT #200,ADS ;LOOP UNTIL DONE
2241 012360 001774 BEQ 5$
2242 012362 032777 000020 166132 6$: BIT #20,ADS ;WAIT FOR SETTLEDOWN BIT TO CLEAR
2243 012370 001374 BNE 6$
2244 012372 012777 000011 166110 MOV #11,@C1 ;DU A DRIVE CLEAR
2245 012400 032777 000004 166130 BIT #4,@DT ;BRANCH IF TE16
2246 012406 001404 BEQ 7$
2247 012410 005737 000602 TST SLVTYP ;BRANCH IF TU77 (SLVTYP = 1)
2248 012414 001012 BNE 100$
2249 012416 000403 BR 99$ ;TAKE ERROR EXIT
2250 012420 005737 000602 7$: TST SLVTYP ;BRANCH IF OPERATOR SELECTED TE16
2251 012424 001406 BEQ 100$
2252 012426 012704 015003 99$: MOV #MSG65,R4 ;TYPE 'INCORRECT SLAVE TYPE...! PROGRAM ABORTED'
2253 012432 004737 013120 JSR PC,TTOUT
2254 012436 012716 002304 MOV #TEND,(SP) ;GO TO END OF PROGRAM
2255 012442 000207 100$: RTS PC ;RETURN TO CALLER
2256
2257 ;MAG TAPE INTERRUPT HANDLER*****
2258
2259 012444 000240 MTINT: NOP
2260 012446 013716 000666 MOV RTRN,(SP) ;SET RETURN TO (RTRN)
2261 012452 000002 RTI ;RETURN
2262
2263 ;TTY INTERRUPT HANDLER*****
2264
2265 012454 017746 166114 TTYINT: MOV @TKB,-(SP) ;GET CHARACTER
2266 012460 042716 000200 BIC #200,(SP) ;CLEAR PARITY BIT
2267 012464 122716 000003 CMPB #3,(SP) ;BRANCH IF NOT CONTROL C
2268 012470 001006 BNE 1$
2269 012472 005737 001260 TST CHNFLG ;INHIBIT ^C IF CHAIN MODE
2270 012476 001003 BNE 1$
2271 012500 000005 RESET
  
```



```

2289 :*****
2290 :TTY ENTRY SUBROUTINE:
2291 :
2292 :THIS SUBROUTINE IS USED BY THE TEST CONDITION
2293 :ENTRY ROUTINE TO READ THE RESPONSE ENTERED
2294 :AT THE TTY AND CHECK THEM FOR LEGALITY AND
2295 :LIMITS. ALL RESPONSE MUST BE TYPED IN OCTAL
2296 :(0-7) AND MUST FALL WITHIN THE LIMITS SET BY
2297 :THE CALLING ROUTINE.
2298 :IF AN ENTRY IS ILLEGAL OR OUTSIDE THE LIMITS,
2299 :A QUESTION MARK IS TYPED (?) AND THE RESPONSE
2300 :MAY BE REENTERED.
2301 :ENTRIES MAY NOT EXCEED SIX (6) CHARACTERS AND
2302 :MAY BE TERMINATED AT LESS THAN SIX BY TYPING A
2303 :CARRIAGE RETURN
2304 :*****
2305
2306 012576 010146 TTR: MOV R1, -(SP) ;SAVE CHAR COUNT
2307 012600 011601 10$: MOV (SP), R1 ;RESET CHAR COUNT (FOR ^U)
2308 012602 005037 000672 CLR TEMP1 ;CLEAR FIRST CHARACTER FLAG
2309 012606 005000 CLR R0
2310 012610 004737 013056 1$: JSR PC, TTIN ;GO READ CHARACTER
2311 012614 122737 000003 000614 CMPB #3, TIB ;BRANCH IF NOT ^C
2312 012622 001003 BNE 11$
2313 012624 000005 RESET
2314 012626 000137 000200 JMP @200 ;RESTART AT 200
2315 012632 122737 000015 000614 11$: CMPB #15, TIB ;SEE IF CR
2316 012640 001004 BNE 2$ ;IF NOT: BR
2317 012642 005737 000672 TST TEMP1 ;SEE IF FIRST CHARACTER
2318 012646 001471 BEQ 9$ ;IF SO: BR
2319 012650 000457 BR 6$ ;ELSE GO LOAD VALUE
2320 012652 122737 000025 000614 2$: CMPB #25, TIB ;BRANCH IF NOT CONTROL U
2321 012660 001005 BNE 21$
2322 012662 012704 014645 MOV #MSG59, R4 ;TYPE<CR><LF>
2323 012666 004737 013120 JSR PC, TTOUT
2324 012672 000742 BR 10$
2325 012674 122737 000177 000614 21$: CMPB #177, TIB ;BRANCH IF NOT 'RUBOUT'
2326 012702 001012 BNE 3$
2327 012704 000241 CLC ;REMOVE LAST TYPED CHAR
2328 012706 006000 ROR R0
2329 012710 006200 ASR R0
2330 012712 006200 ASR R0
2331 012714 012704 014647 MOV #MSG60, R4 ;TYPE '\ '
2332 012720 004737 013120 JSR PC, TTOUT
2333 012724 005201 INC R1 ;DECREMENT CHAR RECEIVED COUNT
2334 012726 000730 BR 1$ ;GET NEXT CHAR
2335 012730 122737 000060 000614 3$: CMPB #60, TIB ;SEE IF CHAR IS LESS THAN 0
2336 012736 101402 BLOS 4$ ;IF NOT: BR
2337 012740 000137 013036 JMP TINNER ;ELSE GO TO ERROR
2338 012744 122737 000070 000614 4$: CMPB #70, TIB ;SEE IF CHAR IS GREATER THAN 7
2339 012752 101002 BHI 5$ ;IF NOT: BR
2340 012754 000137 013036 JMP TINNER ;ELSE GO TO ERROR
2341 012760 005237 000672 5$: INC TEMP1 ;SET FIRST CHARACTER FLAG
2342 012764 006300 ASL R0
2343 012766 006300 ASL R0 ;SHIFT 3 LEFT
2344 012770 006300 ASI R0
    
```

2345	012772	042737	177770	000614		BIC	#177770,TIB	:STRIP ASCII
2346	013000	053700	000614			BIS	TIB,R0	:LOAD CHARACTER
2347	013004	005301				DEC	R1	:SEE IF DONE
2348	013006	001300				SNE	1\$:IF NOT: BR
2349	013010	020002			6\$:	CMP	R0,R2	:SEE IF EXCEEDED MAXIMUM LIMIT
2350	013012	101402				BLOS	7\$:IF NOT: BR
2351	013014	000137	013036			JMP	TINER	:ELSE GO TO ERROR
2352	013020	020300			7\$:	CMP	R3,R0	:SEE IF BELOW MINIMUM LIMIT
2353	013022	101402				BLOS	8\$:IF NOT: BR
2354	013024	000137	013036			JMP	TINER	:ELSE GO TO ERROR
2355	013030	010015			8\$:	MOV	R0,(R5)	:LOAD VALUE
2356	013032	005726			9\$:	TST	(SP)+	:POP CHAR COUNT OFF STACK
2357	013034	000207				RTS	PC	:EXIT

```

2358
2359
2360 ;TTY ENTRY ERROR SUBROUTINE*****
2361 013036 012704 014515 T1NER: MOV #MSG40,R4
2362 013042 004737 013120 JSR PC,TTOUT ;PRINT?
2363 013046 005726 TST (SP)+ ;POP CHAR COUNT OFF STACK
2364 013050 162716 000020 SUB #20,(SP) ;RESET SP TO START OF VALUE ROUTINE
2365 013054 000207 RTS PC ;REDO VALUE ENTRY
2366
2367 ;TTY READ SUBROUTINE*****
2368
2369 013056 005277 165510 TTIN: INC @TKS
2370 013062 105777 165504 1$: TSTB @TKS
2371 013066 100375 BPL 1$
2372 013070 017737 165500 000614 MOV @TKB,TIB
2373 013076 042737 000200 000614 BIC #200,TIB
2374 013104 013737 000614 000612 MOV TIB,TOB ;MOVE CHAR TO TTY OUPUT BFR
2375 013112 004737 013220 JSR PC,TOG ;ECHO CHARACTER
2376 013116 000207 RTS PC
2377
2378 ;TTY OUTPUT SUBROUTINE*****
2379
2380 013120 112437 000612 TTOUT: MOVB (R4)+,TOB
2381 013124 122737 000043 000612 CMPB #43,TOB
2382 013132 001440 BEQ TEX
2383 013134 122737 000045 000612 CMPB #45,TOB
2384 013142 001403 BEQ 1$
2385 013144 004737 013220 JSR PC,TOG
2386 013150 000763 BR TTOUT
2387 013152 112737 000015 000612 1$: MOVB #15,TOB
2388 013160 004737 013220 JSR PC,TOG
2389 013164 012703 000004 MOV #4,R3
2390 013170 005037 000612 2$: CLR TOB
2391 013174 004737 013220 JSR PC,TOG
2392 013200 005303 DEC R3
2393 013202 001372 BNE 2$ ;DO FILLERS
2394 013204 112737 000012 000612 MOVB #12,TOB
2395 013212 004737 013220 JSR PC,TOG
2396 013216 000740 BR TTOUT
2397 013220 105777 165352 TOG: TSTB @TPS
2398 013224 100375 BPL TOG
2399 013226 113777 000612 165344 MOVB TOB,@TPB
2400 013234 000207 RTS PC
2401
2402
2403 ;OCTAL OUTPUT SUBROUTINE*****
2404
2405 013236 012737 000001 013472 OCTPE: MOV #1,OFL
2406 013244 000402 BR OCTPE1
2407 013246 005037 013472 OCTP: CLR OFL ;CLEAR FLAG FOR LEADING ZERO
2408 013252 010304 OCTPE1: MOV R3,R4 ;SEE IF NUMBER IS ZERO
2409 013254 001007 BNE OCTP0 ;IF NOT ZERO: BR
2410 013256 005737 013472 TST OFL ;SEE IF PRINT ALL 0
2411 013262 001004 BNE OCTP0 ;IF SO: BR
2412 013264 004737 013452 JSR PC,OCTPG1 ;ELSE PRINT ZERO
2413 013270 000137 013414 JMP OCTP3 ;SPACE AND EXIT
    
```

```

2414 013274 032704 100000      OCTP0: BI      #100000,R4      ;SEE IF MSD - 1
2415 013300 001406              BEQ      OCTP1      ;IF NOT: BR
2416 013302 012704 000001      MOV      #1,R4
2417 013306 004737 013430      JSR      PC,OCTPG   ;PRINT 1
2418 013312 000137 013324      JMP      OCTP2
2419 013316 005004              OCTP1: CLR      R4
2420 013320 004737 013430      JSR      PC,OCTPG   ;PRINT 0
2421 013324 010304              OCTP2: MOV      R3,R4
2422 013326 006004              ROR      R4
2423 013330 006004              ROR      R4
2424 013332 006004              ROR      R4      ;POSITION DIGIT
2425 013334 006004              ROR      R4
2426 013336 000304              SWAB     R4
2427 013340 004737 013430      JSR      PC,OCTPG   ;PRINT DIGIT 2
2428 013344 010304              MOV      R3,R4
2429 013346 006004              ROR      R4
2430 013350 000304              SWAB     R4
2431 013352 004737 013430      JSR      PC,OCTPG   ;PRINT DIGIT 3
2432 013356 010304              MOV      R3,R4
2433 013360 006104              ROL      R4
2434 013362 006104              RCL      R4
2435 013364 000304              SWAB     R4
2436 013366 004737 013430      JSR      PC,OCTPG   ;PRINT DIGIT 4
2437 013372 010304              MOV      R3,R4
2438 013374 006004              ROR      R4
2439 013376 006004              ROR      R4
2440 013400 006004              ROR      R4
2441 013402 004737 013430      JSR      PC,OCTPG
2442 013406 010304              MOV      R3,R4
2443 013410 004737 013430      JSR      PC,OCTPG
2444 013414 012737 000240 000612 OCTP3: MOV      #240,TOB ;PRINT DIGIT 5
2445 013422 004737 013220      JSR      PC,TOG
2446 013426 000207              RTS      PC      ;PRINT SPACE
2447 013430 042704 177770      OCTPG: BIC      #177770,R4 ;EXIT
2448 013434 001004              BNE      OCTPG0
2449 013436 005737 013472      TST      OFL
2450 013442 001001              BNE      OCTPG0
2451 013444 000207              RTS      PC
2452
2453 013446 005237 013472      OCTPG0: INC      OFL
2454 013452 052704 000260      OCTPG1: RIS      #260,R4
2455 013456 010437 000612      MOV      R4,TOB
2456 013462 004737 013220      JSR      PC,TOG
2457 013466 010304              MOV      R3,R4
2458 013470 000207              RTS      PC
2459 013472 000000      OFL:      C
2460
;FIRST CHAR FLAG
  
```

```

2461
2462 ;DATA CHARACTER OUTPUT SUBROUTINE*****
2463
2464 013474 012704 000010 DOUT: MOV #10,R4 ;SET NUMBER TO PRINT
2465 013500 110337 000612 MOVB R3,TOB
2466 013504 105777 165066 1$: TSTB @TPS
2467 013510 100375 BPL 1$
2468 013512 132737 000200 000612 BITB #200,TOB
2469 013520 001404 BEQ 2$
2470 013522 012777 000061 165050 MOV #061,@TPB
2471 013530 000403 BR 3$
2472 013532 012777 000060 165040 2$: MOV #060,@TPB
2473 013540 006337 000612 3$: ASL TOB
2474 013544 005304 DEC R4
2475 013546 001356 BNE 1$
2476 013550 000207 RTS PC
2477
2478 013552 013703 000676 DOUTD: MOV TEMP3,R3
2479 013556 000303 SWAB R3
2480 013560 004737 013474 JSR PC,DOUT
2481 013564 013703 000676 MOV TEMP3,R3
2482 013570 004737 013474 JSR PC,DOUT
2483 013574 000207 RTS PC
2484
2485 ;SERIAL NUMBER PRINT SUBROUTINE*****
2486
2487 013576 010304 SNPT: MOV R3,R4
2488 013600 000304 SWAB R4
2489 013602 006004 ROR R4
2490 013604 006004 ROR R4
2491 013606 006004 ROR R4
2492 013610 006004 ROR R4
2493 013612 004737 013654 JSR PC,SNPG ;GET FIRST DIGIT
2494 013616 010304 MOV R3,R4 ;PRINT
2495 013620 000304 SWAB R4 ;GET SECOND DIGIT
2496 013622 004737 013654 JSR PC,SNPG ;PRINT
2497 013626 010304 MOV R3,R4
2498 013630 006004 ROR R4
2499 013632 006004 ROR R4
2500 013634 006004 ROR R4
2501 013636 006004 ROR R4
2502 013640 004737 013654 JSR PC,SNPG ;PRINT THIRD DIGIT
2503 013644 010304 MOV R3,R4
2504 013646 004737 013654 JSR PC,SNPG ;PRINT FOURTH DIGIT
2505 013652 000207 RTS PC ;EXIT
2506 013654 012737 000260 000612 SNPG: MOV #260,TOB ;SET BASE = 0
2507 013662 042704 177760 BIC #177760,R4 ;MASK DIGIT
2508 013666 050437 000612 BIS R4,TOB ;SET ASCII
2509 013672 004737 013220 JSR PC,TOG ;TYPE DIGIT
2510 013676 000207 RTS PC ;RETURN
  
```

```

2511
2512           ;THIS ROUTINE GETS THE NEW VALUE FOR THE SOFTWARE SWITCH REG
2513
2514 013700 022737 000176 000570 GTSWR: CMP    #SWREG,SWR    ;BRANCH IF SOFTWARE SWR NOT
2515 013706 001032                BNE    1$          ;INVOKED
2516 013710 004737 013776                JSR    PC,.SAVE    ;SAVE REGISTERS ON THE STACK
2517 013714 012704 016306                MOV    #SMSWR,R4   ;TYPE 'SWR = '
2518 013720 004737 013120                JSR    PC,TTOUT
2519 013724 017703 164640                MOV    @SWR,R3     ;GET CURRENT SETTING
2520 013730 004737 013236                JSR    PC,OCPE     ;AND TYPE THEM
2521 013734 012704 016316                MOV    #SMNEW,R4   ;TYPE 'NEW = '
2522 013740 004737 013120                JSR    PC,TTOUT
2523 013744 013705 000570                MOV    SWR,R5      ;TTR ROUTINE RETURN NEW VALUE TO (R5)
2524 013750 012701 000007                MOV    #7,R1       ;LIMIT RESPONSE TO 7 CHARS
2525 013754 012702 177777                MOV    #177777,R2 ;BETWEEN 0 AND 177777
2526 013760 012703 000000                MOV    #0,R3
2527 013764 004737 012576                JSR    PC,TTR      ;GET RESPONSE
2528 013770 004737 014020                JSR    PC,.RESTORE ;RESTORE REGISTERS
2529 013774 000207                1$:   RTS    PC     ;RETURN TO CALLER
2530
2531           ;;ROUTINE TO SAVE REGISTERS ON THE STACK
2532 013776 010546                .SAVE: MOV    %5,-(SP) ;:R5 IS SAVED AT 12(SP)
2533 014000 010446                MOV    %4,-(SP) ;:R4 IS SAVED AT 10(SP)
2534 014002 010346                MOV    %3,-(SP) ;:R3 IS SAVED AT 6(SP)
2535 014004 010246                MOV    %2,-(SP) ;:R2 IS SAVED AT 4(SP)
2536 014006 010146                MOV    %1,-(SP) ;:R1 IS SAVED AT 2(SP)
2537 014010 010046                MOV    %0,-(SP) ;:R0 IS SAVED AT (SP)
2538 014012 016646 000014                MOV    14(SP),-(SP) ;:PUSH RETURN PC ON THE STACK
2539 014016 000207                RTS    PC          ;:RETURN TO CALLER
2540
2541           ;;ROUTINE TO RESTORE REGISTERS SAVED ON THE STACK
2542 014020 012666 000014                .RESTORE:MOV   (SP)+,14(SP) ;:STORE RETURN PC ON STACK
2543 014024 012600                MOV    (SP)+,%0
2544 014026 012601                MOV    (SP)+,%1
2545 014030 012602                MOV    (SP)+,%2
2546 014032 012603                MOV    (SP)+,%3
2547 014034 012604                MOV    (SP)+,%4
2548 014036 012605                MOV    (SP)+,%5
2549 014040 000207                RTS    PC          ;:RETURN
2550
    
```

```

2551
2552                                     :MESSAGE TABLE*****
2553
2554 014042 022445 046524 031460 MSG1: .ASCII'%TM03-TE16/TU77 CONTROL LOGIC TEST PART II (CZTEBBO)';++B
2555 014050 052055 030505 027466
2556 014056 052524 033467 041440
2557 014064 047117 051124 046117
2558 014072 046040 043517 041511
2559 014100 052040 051505 020124
2560 014106 040520 052122 044440
2561 014114 020111 024040 055103
2562 014122 042524 041102 024460
2563 014130 025045 025052 051501 .ASCII /%***ASSURE TAPE IS AT BOT***/
2564 014136 052523 042522 052040
2565 014144 050101 020105 051511
2566 014152 040440 020124 047502
2567 014160 025124 025052
2568 014164 052045 050131 020105 .ASCII /%TYPE <CR> TO TERMINATE RESPONSE & ^C TO RESTART%/
2569 014172 041474 037122 052040
2570 014200 020117 042524 046522
2571 014206 047111 052101 020105
2572 014214 042522 050123 047117
2573 014222 042523 023040 057040
2574 014230 020103 047524 051040
2575 014236 051505 040524 052122
2576 014244 021445
2577 014246 054105 052120 047055 MSG6: .ASCII /EXPT-NOT RECVD#/
2578 014254 052117 051040 041505
2579 014262 042126 043
2580 014265 122 053103 026504 MSG7: .ASCII /RCVD-NOT EXPT#/
2581 014272 047516 020124 054105
2582 014300 052120 043
2583 014303 045 047516 026516 MSG9: .ASCII /%NON-EXIST SLAVE #/
2584 014310 054105 051511 020124
2585 014316 046123 053101 020105
2586 014324 043
2587 014325 045 042522 042101 MSG10: .ASCII /%READ CONT BUS PAR #/
2588 014332 041440 047117 020124
2589 014340 052502 020123 040520
2590 014346 020122 043
2591 014351 045 051127 052111 MSG11: .ASCII /%WRITE CONT BUS PAR #/
2592 014356 020105 047503 052116
2593 014364 041040 051525 050040
2594 014372 051101 021440
2595 014376 042440 050130 020124 MSG12: .ASCII / EXPT #/
2596 014404 043
2597 014405 040 041522 042126 MSG13: .ASCII / RCVD #/
2598 014412 021440
2599 014414 046445 020122 044502 MSG14: .ASCII /%R BITS 4-0#/
2600 014422 051524 032040 030055
2601 014430 043
2602 014431 045 051115 041040 MSG15: .ASCII /%R BITS 15-7#/
2603 014436 052111 020123 032461
2604 014444 033455 043
2605 014447 045 052111 051105 MSG16: .ASCII /%!TER: #/
2606 014454 020072 043
  
```


2607	014457	045	041524	041040	MSG18:	.ASCII	/%TC BITS 12-0 #/
2608	014464	052111	020123	031061			
2609	014472	030055	021440				
2610	014476	043045	020103	044502	MSG19:	.ASCII	/%FC BITS 15-0 #/
2611	014504	051524	030440	026465			
2612	014512	020060	043				
2613	014515	040	020077	043	MSG40:	.ASCII	/ ? #/
2614	014521	045	042445	042116	MSG41:	.ASCII	/%END OF PASS #/
2615	014526	047440	020106	040520			
2616	014534	051523	021440				
2617	014540	051045	043505	051511	MSG44:	.ASCII	/%REGISTER START: #/
2618	014546	042524	020122	052123			
2619	014554	051101	035124	021440			
2620	014562	053045	041505	047524	MSG45:	.ASCII	/%VECTOR ADDRESS: #/
2621	014570	020122	042101	051104			
2622	014576	051505	035123	021440			
2623	014604	052045	030115	020063	MSG57:	.ASCII	/%TMO3 DRIVE: #/
2624	014612	051104	053111	035105			
2625	014620	021440					
2626	014622	052045	030505	027466	MSG58:	.ASCII	'%TE16/TU77 SLAVE: #' ; ++B
2627	014630	052524	033467	051440			
2628	014636	040514	042526	020072			
2629	014644	043					
2630	014645	045	043		MSG59:	.ASCII	/%#/
2631	014647	134	043		MSG60:	.ASCII	/\#/
2632	014651	045	042522	047515	MSG62:	.ASCII	/%REMOVE TMDP FROM SLAVE TO BE TESTED%/
2633	014656	042526	052040	042115			
2634	014664	020120	051106	046517			
2635	014672	051440	040514	042526			
2636	014700	052040	020117	042502			
2637	014706	052040	051505	042524			
2638	014714	022504	043				
2639	014717	045	040510	042122	MSG63:	.ASCII	/%HARDWARE SWR IN USE%/
2640	014724	040527	042522	051440			
2641	014732	051127	044440	020116			
2642	014740	051525	022505	043			
2643	014745	045	046123	053101	MSG64:	.ASCII	/%SLAVE TYPE (0-TE16,i-TU77): #/
2644	014752	020105	054524	042520			
2645	014760	024040	036460	042524			
2646	014766	033061	030454	052075			
2647	014774	033525	024467	020072			
2648	015002	043					
2649	015003	045	044445	041516	MSG65:	.ASCII	/%!INCORRECT SLAVE TYPE... PROGRAM ABORTED%/
2650	015010	051117	042522	052103			
2651	015016	051440	040514	042526			
2652	015024	052040	050131	020505			
2653	015032	020441	050040	047522			
2654	015040	051107	046501	040440			
2655	015046	047502	052122	042105			
2656	015054	021445					

```
2657 ;TEST HEADER*****
2658
2659 015056 022445 047514 044507 MSLT1: .ASCII /%%LOGIC TEST 1: WRAP 3,NRZ,NORMAL,ODD#/
2660 015064 020103 042524 052123
2661 015072 030440 020072 051127
2662 015100 050101 031440 047054
2663 015106 055122 047054 051117
2664 015114 040515 026114 042117
2665 015122 021504
2666 015124 022445 047514 044507 MSLT2: .ASCII /%%LOGIC TEST 2: WRAP 3,PE,NORMAL,ODD#/
2667 015132 020103 042524 052123
2668 015140 031040 020072 051127
2669 015146 050101 031440 050054
2670 015154 026105 047516 046522
2671 015162 046101 047454 042104
2672 015170 043
2673 015171 045 046045 043517 MSLT3: .ASCII /%%LOGIC TEST 3: WRAP 2,NRZ,NORMAL,ODD#/
2674 015176 041511 052040 051505
2675 015204 020124 035063 053440
2676 015212 040522 020120 026062
2677 015220 051116 026132 047516
2678 015226 046522 046101 047454
2679 015234 042104 043
2680 015237 045 046045 043517 MSLT4: .ASCII /%%LOGIC TEST 4: WRAP 2,PE,NORMAL,ODD#/
2681 015244 041511 052040 051505
2682 015252 020124 035064 053440
2683 015260 040522 020120 026062
2684 015266 042520 047054 051117
2685 015274 040515 026114 042117
2686 015302 021504
2687 015304 022445 047514 044507 MSLT5: .ASCII /%%LOGIC TEST 5: WRAP 1,NRZ,NORMAL,ODD#/
2688 015312 020103 042524 052123
2689 015320 032440 020072 051127
2690 015326 050101 030440 047054
2691 015334 055122 047054 051117
2692 015342 040515 026114 042117
2693 015350 021504
2694 015352 022445 047514 044507 MSLT6: .ASCII /%%LOGIC TEST 6: WRAP 1,PE,NORMAL,ODD#/
2695 015360 020103 042524 052123
2696 015366 033040 020072 051127
2697 015374 050101 030440 050054
2698 015402 026105 047516 046522
2699 015410 046101 047454 042104
2700 015416 043
2701 015417 045 046045 043517 MSLT7: .ASCII /%%LOGIC TEST 7: WRAP 0,NRZ,NORMAL,ODD#/
2702 015424 041511 052040 051505
2703 015432 020124 035067 053440
2704 015440 040522 020120 026060
2705 015446 051116 026132 047516
2706 015454 046522 046101 047454
2707 015462 042104 043
2708 015465 045 046045 043517 MSLT10: .ASCII /%%LOGIC TEST 10: WRAP 0,PE,NORMAL,ODD#/
2709 015472 041511 052040 051505
2710 015500 020124 030061 020072
2711 015506 051127 050101 030040
2712 015514 050054 026105 047516
```

2713	015522	046522	046101	047454	
2714	015530	042104	043		
2715	015533	045	046045	043517	MSLT11: .ASCII /%%LOGIC TEST 11: CORE DUMP WRITE (M8906)#/
2716	015540	041511	052040	051505	
2717	015546	020124	030461	020072	
2718	015554	047503	042522	042040	
2719	015562	046525	020120	051127	
2720	015570	052111	020105	046450	
2721	015576	034470	033060	021451	
2722	015604	022445	047514	044507	MSLT12: .ASCII /%%LOGIC TEST 12: CORE DUMP READ (M8906)#/
2723	015612	020103	042524	052123	
2724	015620	030440	035062	041440	
2725	015626	051117	020105	052504	
2726	015634	050115	051040	040505	
2727	015642	020104	046450	034470	
2728	015650	033060	021451		
2729	015654	022445	047514	044507	MSLT13: .ASCII /%%LOGIC TEST 13: EVEN PARITY WRITE (M8933 M8934)#/
2730	015662	020103	042524	052123	
2731	015670	030440	035063	042440	
2732	015676	042526	020116	040520	
2733	015704	044522	054524	053440	
2734	015712	044522	042524	024040	
2735	015720	034115	031471	020063	
2736	015726	034115	031471	024464	
2737	015734	043			
2738	015735	045	046045	043517	MSLT14: .ASCII /%%LOGIC TEST 14: EVEN PARITY READ(M8933 M8934)#/
2739	015742	041511	052040	051505	
2740	015750	020124	032061	020072	
2741	015756	053105	047105	050040	
2742	015764	051101	052111	020131	
2743	015772	042522	042101	046450	
2744	016000	034470	031463	046440	
2745	016006	034470	032063	021451	
2746	016014	022445	047514	044507	MSLT15: .ASCII /%%LOGIC TEST 15: READ REVERSE(M8906)#/
2747	016022	020103	042524	052123	
2748	016030	030440	035065	051040	
2749	016036	040505	020104	042522	
2750	016044	042526	051522	024105	
2751	016052	034115	030071	024466	
2752	016060	043			
2753	016061	045	046045	043517	MSLT16: .ASCII /%%LOGIC TEST 16: CRC CORRECTION SINGLE TRACK,ALL FRAMES#/
2754	016066	041511	052040	051505	
2755	016074	020124	033061	020072	
2756	016102	051103	020103	047503	
2757	016110	051122	041505	044524	
2758	016116	047117	051440	047111	
2759	016124	046107	020105	051124	
2760	016132	041501	026113	046101	
2761	016140	020114	051106	046501	
2762	016146	051505	043		
2763	016151	045	046045	043517	MSLT17: .ASCII /%%LOGIC TEST 17: CRC CORRECTION MULTIPLE BAD TRACKS#/
2764	016156	041511	052040	051505	
2765	016164	020124	033461	020072	
2766	016172	051103	020103	047503	
2767	016200	051122	041505	044524	
2768	016206	047117	046440	046125	

2769	016214	044524	046120	020105
2770	016222	040502	020104	051124
2771	016230	041501	051513	043
2772	016235	045	046045	043517
2773	016242	041511	052040	051505
2774	016250	020124	030062	020072
2775	016256	042522	042101	051040
2776	016264	053105	051105	042523
2777	016272	047054	055122	053454
2778	016300	040522	020120	021463

MSLT20: .ASCII /%LOGIC TEST 20: READ REVERSE,NRZ,WRAP 3#1

```
2779
2780 ;TAG MESSAGE
2781
2782 016306 051445 051127 036440 SMSWR: .ASCII /%SWR = #/
2783 016314 021440
2784 016316 047040 053505 036440 SMNEW: .ASCII / NEW = #/
2785 016324 021440
2786
2787 .EVEN
2788 ;WRITE BUFFER
2789
2790 016326 000100 WDATA:
2791 016326 177777 -1
2792 016330 177777 -1
2793 016332 177777 -1
2794 016334 177777 -1
2795 016336 177777 -1
2796 016340 177777 -1
2797 016342 177777 -1
2798 016344 177777 -1
2799 016346 177777 -1
2800 016350 177777 -1
2801 016352 177777 -1
2802 016354 177777 -1
2803 016356 177777 -1
2804 016360 177777 -1
2805 016362 177777 -1
2806 016364 177777 -1
2807 016366 177777 -1
2808 016370 177777 -1
2809 016372 177777 -1
2810 016374 177777 -1
2811 016376 177777 -1
2812 016400 177777 -1
2813 016402 177777 -1
2814 016404 177777 -1
2815 016406 177777 -1
2816 016410 177777 -1
2817 016412 177777 -1
2818 016414 177777 -1
2819 016416 177777 -1
2820 016420 177777 -1
2821 016422 177777 -1
2822 016424 177777 -1
2823 016426 177777 -1
2824 016430 177777 -1
2825 016432 177777 -1
2826 016434 177777 -1
2827 016436 177777 -1
2828 016440 177777 -1
2829 016442 177777 -1
2830 016444 177777 -1
2831 016446 177777 -1
2832 016450 177777 -1
2833 016452 177777 -1
2834 016454 177777 -1
```

2835	016456	177777	-1
2836	016460	177777	-1
2837	016462	177777	-1
2838	016464	177777	-1
2839	016466	177777	-1
2840	016470	177777	-1
2841	016472	177777	-1
2842	016474	177777	-1
2843	016476	177777	-1
2844	016500	177777	-1
2845	016502	177777	-1
2846	016504	177777	-1
2847	016506	177777	-1
2848	016510	177777	-1
2849	016512	177777	-1
2850	016514	177777	-1
2851	016516	177777	-1
2852	016520	177777	-1
2853	016522	177777	-1
2854	016524	177777	-1

2855
2856
2857 ;READ BUFFER
2858

2859	016526	000100	RDATA:
2860	016526	000000	0
2861	016530	000000	0
2862	016532	000000	0
2863	016534	000000	0
2864	016536	000000	0
2865	016540	000000	0
2866	016542	000000	0
2867	016544	000000	0
2868	016546	000000	0
2869	016550	000000	0
2870	016552	000000	0
2871	016554	000000	0
2872	016556	000000	0
2873	016560	000000	0
2874	016562	000000	0
2875	016564	000000	0
2876	016566	000000	0
2877	016570	000000	0
2878	016572	000000	0
2879	016574	000000	0
2880	016576	000000	0
2881	016600	000000	0
2882	016602	000000	0
2883	016604	000000	0
2884	016606	000000	0
2885	016610	000000	0
2886	016612	000000	0
2887	016614	000000	0
2888	016616	000000	0
2889	016620	000000	0
2890	016622	000000	0

2891	016624	000000	0
2892	016626	000000	0
2893	016630	000000	0
2894	016632	000000	0
2895	016634	000000	0
2896	016636	000000	0
2897	016640	000000	0
2898	016642	000000	0
2899	016644	000000	0
2900	016646	000000	0
2901	016650	000000	0
2902	016652	000000	0
2903	016654	000000	0
2904	016656	000000	0
2905	016660	000000	0
2906	016662	000000	0
2907	016664	000000	0
2908	016666	000000	0
2909	016670	000000	0
2910	016672	000000	0
2911	016674	000000	0
2912	016676	000000	0
2913	016700	000000	0
2914	016702	000000	0
2915	016704	000000	0
2916	016706	000000	0
2917	016710	000000	0
2918	016712	000000	0
2919	016714	000000	0
2920	016716	000000	0
2921	016720	000000	0
2922	016722	000000	0
2923	016724	000000	0

;WRAP AROUND MESSAGES*****

2927	016726	051445	052105	050125	WMSG2: .ASCII /%SETUP ERROR%#/ 051122 051127
2928	016734	042440	051122	051127	
2929	016742	021445			
2930	016744	050045	052101	047122	WMSG3: .ASCII /%PATRN NUMBER #/ 046525 042502
2931	016752	047040	046525	042502	
2932	016760	020122	020075	043	
2933	016765	045	047516	026516	WMSG4: .ASCII /%NON-EXISTANT DRIVE%#/ 047516 026516
2934	016772	054105	051511	040524	
2935	017000	052116	042040	044522	
2936	017006	042526	021445		
2937	017012	041445	030523	021440	WMSG6: .ASCII /%CS1 #/ 030523 021440
2938	017020	053445	020103	043	WMSG6A: .ASCII /%WC #/ 020103 043
2939	017025	045	040502	021440	WMSG6B: .ASCII /%SA #/ 040502 021440
2940	017032	043045	020103	043	WMSG6C: .ASCII /%FC #/ 020103 043
2941	017037	045	051503	020062	WMSG6D: .ASCII /%CS2 #/ 051503 020062
2942	017044	043			
2943	017045	045	051504	021440	WMSG6E: .ASCII /%DS #/ 051504 021440
2944	017052	042445	020122	043	WMSG6F: .ASCII /%ER #/ 020122 043
2945	017057	045	051501	021440	WMSG6G: .ASCII /%AS #/ 051501 021440
2946	017064	041445	020103	043	WMSG6H: .ASCII /%CC #/ 020103 043

2947	017071	045	041104	021440	WMSG6I:	.ASCII	/ZDB #/
2948	017076	046445	020122	043	WMSG6J:	.ASCII	/ZMR #/
2949	017103	045	052104	021440	WMSG6K:	.ASCII	/ZDT #/
2950	017110	052045	020103	043	WMSG6L:	.ASCII	/ZTC #/
2951	017115	045	047123	021440	WMSG6M:	.ASCII	/ZSN #/
2952	017122	041045	042101	042040	WMSG16:	.ASCII	/ZBAD DATA#/
2953	017130	052101	021501				
2954	017134	043445	020072	043	WMSG17:	.ASCII	/ZG: #/
2955	017141	045	035102	021440	WMSG20:	.ASCII	/ZB: #/
2956	017146	041445	035116	021440	WMSG21:	.ASCII	/ZCN: #/
2957	017154	041045	042101	051440	WMSG23:	.ASCII	/ZBAD STATUS#/
2958	017162	040524	052524	021523			
2959	017170	047045	020117	047111	WMSG24:	.ASCII	/ZNO I 'ERRUPT#/
2960	017176	042524	051122	050125			
2961	017204	021524					
2962	017206	047045	020117	046103	WMSG25:	.ASCII	/ZNO CLOCK UP#/
2963	017214	041517	020113	050125			
2964	017222	043					
2965	017223	045	047516	041440	WMSG26:	.ASCII	/ZNO CLOCK DOWN#/
2966	017230	047514	045503	042040			
2967	017236	053517	021516				
2968	017242	042045	052101	020101	WMSG27:	.ASCII	/ZDATA PAT: #/
2969	017250	040520	035124	043			
2970	017255	045	040502	020104	WMSG28:	.ASCII	/ZBAD PREAMBLE#/
2971	017262	051120	040505	041115			
2972	017270	042514	043				
2973	017273	045	040502	020104	WMSG29:	.ASCII	/ZBAD POSTAMBLE#/
2974	017300	047520	052123	046501			
2975	017306	046102	021505				
2976	017312	042445	051117	041440	WMSG31:	.ASCII	/ZFOR CLEAR DID NOT CLEAR GO%#/
2977	017320	042514	051101	042040			
2978	017326	042111	047040	052117			
2979	017334	041440	042514	051101			
2980	017342	043440	022517	043			
2981	017347	040	040520	051124	WMSG32:	.ASCII	/ PATRN #/
2982	017354	020116	043				
2983							
2984		017360					
2985	017360	000000			PRE:	0	EVEN
2986	017362	000000				0	
2987	017364	000000				0	
2988	017366	000000				0	
2989	017370	000000				0	
2990	017372	000000				0	
2991	017374	000000				0	
2992	017376	000000				0	
2993	017400	000000				0	
2994	017402	000000				0	
2995	017404	000000				0	
2996	017406	000000				0	
2997	017410	000000				0	
2998	017412	000000				0	
2999	017414	000000				0	
3000	017416	000000				0	
3001	017420	000000				0	
3002	017422	000000				0	

3003	017424	000000	0
3004	017426	000000	0
3005	017430	000000	0
3006	017432	000000	0
3007	017434	000000	0
3008	017436	000000	0
3009	017440	000000	0
3010	017442	000000	0
3011	017444	000000	0
3012	017446	000000	0
3013	017450	000000	0
3014	017452	000000	0
3015	017454	000000	0
3016	017456	000000	0
3017	017460	000000	0
3018	017462	000000	0
3019	017464	000000	0
3020	017466	000000	0
3021	017470	000000	0
3022	017472	000000	0
3023	017474	000000	0
3024	017476	000000	0
3025	017500	000000	0
3026	017502	000000	0
3027	017504	000000	0
3028	017506	000000	0
3029	017510	000000	0
3030	017512	000000	0
3031	017514	000000	0
3032	017516	000000	0
3033	017520	000000	0
3034	017522	000000	0
3035	017524	000000	0
3036	017526	000000	0
3037	017530	000000	0
3038	017532	000000	0
3039	017534	000000	0
3040	017536	000000	0
3041	017540	000000	0
3042	017542	000000	0
3043	017544	000000	0
3044	017546	000000	0
3045	017550	000000	0
3046	017552	000000	0
3047	017554	000000	0
3048	017556	000000	0
3049	017560	000000	0
3050	017562	000000	0
3051	017564	000000	0
3052	017566	000000	0
3053	017570	000000	0
3054	017572	000000	0
3055	017574	000000	0
3056	017576	000000	0
3057	017600	000000	0
3058	017602	000000	0

POST:

3059	017604	000000		0
3060	017606	000000		0
3061	017610	000000		0
3062	017612	000000		0
3063	017614	000000		0
3064	017616	000000		0
3065	017620	000000		0
3066	017622	000000		0
3067	017624	000000	WBUFF:	0
3068		020236		0
3069	020236	000000	RBUFF:	0
3070				0
3071		000001		.END

ST2	001730	667	960#											
SWR	000570	709#	868	872*	880*	971	1005	1011	1041	1394	1518	1522	1549	1553
		1674	1678	1694	1698	1745	1753	1834	1882	1889	1934	1975	2012	2076
		2159	2167	2177	2181	2192	2203	2275	2277*	2284*	2514	2519	2523	
SWREG	000176	658#	872	880	2275	2284	2514							
TADX	001172	858#												
TC	000542	692#	995*	1540*	1572	2221*	2236*							
TEMP1	000672	748#	2308*	2317	2341*									
TEMP2	000674	749#	2145*	2171	2185*									
TEMP3	000676	750#	2478	2481										
TEND	002304	858	1016	1022#	2254									
TENDX	002472	1040	1042	1044#										
TEX	013234	2382	2400#											
TIB	000614	725#	2311	2315	2320	2325	2335	2338	2345*	2346	2372*	2373*	2374	
TINER	013036	2337	2340	2351	2354	2361#								
TKB	000574	711#	2265	2372										
TKS	000572	710#	968*	2369*	2370									
TLAST	001174	859#	1015											
TOB	000612	724#	1983*	1985*	1988*	2033*	2041*	2101*	2109*	2374*	2380*	2381	2383	2387*
		2390*	2394*	2399	2444*	2455*	2465*	2468	2473*	2506*	2508*			
TOG	013220	1984	1989	2034	2042	2102	2110	2375	2385	2388	2391	2395	2397#	2398
		2445	2456	2509										
TPB	000600	713#	2399*	2470*	2472*									
TPS	000576	712#	2397	2466										
TREF	000732	764#												
TR00	000624	729#												
TR01	000626	730#												
TR02	000630	731#												
TR03	000632	732#												
TR04	000634	733#												
TR05	000636	734#												
TR06	000640	735#												
TR07	000642	736#												
TR10	000644	737#												
TR11	000646	738#												
TR12	000650	739#												
TR13	000652	740#												
TR14	000654	741#												
TR15	000656	742#												
TSCD	001754	894	968#	1013	1045									
TSCDA	002150	975	998#											
TSCDO	002156	999#	1010											
TSCD1	002164	1000#	1021											
TSCD2	002212	1005#	1066	1094	1122	1151	1179	1199	1304	1357	1377			
TSCD3	002230	1006	1009#											
TSTIBL	001066	824#	998	1019										
TTIN	013056	2310	2369#											
TTINT	012454	652	2265#											
TTOUT	013120	888	891	897	906	931	940	950	1025	1397	1521	1552	1677	1758
		1760	1763	1765	1769	1839	1843	1845	1851	1855	1859	1939	1943	1945
		1952	1956	1963	2017	2023	2026	2030	2038	2081	2083	2085	2090	2094
		2106	2164	2166	2253	2280	2323	2332	2362	2380#	2386	2396	2518	2522
TTR	012576	904	913	938	947	957	1404	2306#	2527					
T24FL	000742	768#												
UDES	000774	781#	1057*	1076*	1086*	1104*	1114*	1133*	1143*	1161*	1170*	1188*	1208*	1218*
		1228*	1263*	1326*	1369*	1479	1511	1537	1540	1585	1590	1605	1636	1651

CZTEBB0 TMO3-TE16/TU77 CLT II
CZTEBB.P11 15-NOV-78 13:16

MACY11 30A(1052) 21-DEC-78 13:12^{E 7} PAGE 84
CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0082

\$CATCH	554#	639
\$CHAIN	554#	875
\$CHMO	554#	974
\$RESTO	554#	2541
\$SAVE	554#	2531
.\$ACT1	554#	640
.\$EOP	554#	1031

. ABS. 020240 000

ERRORS DETECTED. 0

DSKZ: CZTEBB, DSKZ: CZTEBB, SEQ/CRF/SOL=CZTEAB.SML/ML, CZTEBB.P11
RUN-TIME: 47.9 SECONDS
RUN-TIME RATIO: 134/12-10.9
CORE USED: 9K (17 PAGES)