

TM03/TU45

CONTROL LOGIC TEST PART 2
CZTUPA0

AH-E491A-MC
COPYRIGHT © 75-78
FICHE 1 OF 1

JUL 1978
digital
MADE IN USA

Microfiche grid containing 10 columns and 15 rows of frames. Each frame contains a small document page with text and diagrams, including a schematic diagram in the top-right frame of the first row.



.REM 2

IDENTIFICATION

PRODUCT CODE: AC-E490A-MC
PRODUCT NAME: CZTUPA0 TM03-TU45 CONTROL LOGIC TEST PART II
DATE CREATED: 25 MAY 1978
MAINTAINER: CSS - NASHUA
AUTHOR: J. G. ADAMS/R. J. COLLINS

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 PROVIDE IN WRITING BY DIGITAL.

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

COPYRIGHT (c) 1975, 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 TM03 FORMATTER. EACH TEST WILL ATTEMPT TO ISOLATE FAILURES TO THE MODULE LEVEL AND PROVIDE PRINTOUT INFORMATION WHICH WILL IDENTIFY THE FAILING MODULE. THE LEVEL OF FAULT ISOLATION IS POSSIBLE BECAUSE OF TM03 THE STRUCTURE AND ITS MAINTAINENCE MODES.

2. REQUIREMENTS (HARDWARE)

- A. ANY PDP-11 PROCESSOR
B. 8K OF CORE
C. CONSOLE TTY
D. TM03 MAGTAPE CONTROLLER
E. MASSBUS CONTROLLER (RH)
F. TU45 MAGTAPE TRANSPORT

3. LOADING PROCEDURE

USE STANDARD PROCEDURE FOR LOADING BINARY PAPER TAPE.

4. STARTING PROCEDURE

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

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

** NOTE: SEE ALSO SEC 5. CONSOLE SWITCH SETTINGS
** TYPE C TO RESTART PROGRAM (@200)

4.1 AUTOMATIC MODE OPERATION

IF THIS PROGRAM IS LOADED AND RUN IN AUTOMATIC (CHAIN) MODES
DEFAULT RESPONSES TO OPERATOR REQUESTS ARE USED. THE SOFTWARE
SWR IS INVOKED WITH A SWITCH SETTING OF 100000 (HALT ON ERROR)
IF IN ACT11 CHAIN MODE. NO OPERATOR INTERVENTION IS REQUIRED.

**EXCEPTION: IF THIS PROGRAM IS LOADED VIA TMDP CHAIN MODE THE
PROGRAM WILL NOT TEST TM03 DRIVE #0, TU45 SLAVE #0.

4.2 SAMPLE START AT 200

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

PARAMETER REQUEST: <DEFAULT> (RESPONSE) LOCATION:

TM03-TU45 CONTROL LOGIC TEST PART II (CZTUPA0)
ASSURE TAPE IS AT BOT
TYPE C TO RESTART

REGISTER START: <172440> (CR)	REGS:
VECTOR ADDRESS: <224> (CR)	VECT:
TM03 DRIVE: <0> (CR)	DRVN:
TU45 SLAVE: <0> (CR)	SLVN:
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 FOLOWING ARE TWO EXAMPLES OF ERRORS DETECTED BY THE WRAP AROUND DATA TESTS. NOTE THAT EACH WRAP AROUND TEST MAY BE ACCOMPANIED BY EITHER A STATUS ERROR OF A DATA ERROR OR BOTH.

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

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

LOGIC TEST 3: WRAP 2, NRZ, NORMAL, ODD
BAD DATA
CN:0
G: 1111111
B: 1111011
CN:10
G: 0000000
B: 00001000

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

7. OPERATION

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

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

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

SINGLE TEST SELECTION: (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 TM03 LOGIC INDEPENDENT OF THE TU45. HOWEVER THERE ARE A FEW SIGNALS WHICH ARE REQUIRED FROM THE TU45 TO COMPLETE THE TESTS, AND AT LEAST ONE CASE WHERE TU45 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, NR2, 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 TCCM2
CRC GENERATOR CNR2

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 TCCM2
WRITE CONTROL TCPE3

LOGIC TEST 7: WRAP0, NORMAL, ODD

WRAP 0 IS THE MOST COMPLETE OF THE TM03 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 TM03 DOES NOT WRITE THE LRC CHARACTER.

LIKELY FAULT LOCATIONS: M8934, M8933

CIRCUITS -----	PRINT REFERENCES -----
CRC GENERATOR	CNR22
CRC CHECKOUT	CNR23
CRC, CRC STROBE	TCCM4
READ LINE MULTIPLEXERS	TCCM6
MM CLK	MRS
CRC READ TIMING	CNR24
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 MAJOP 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, M8939

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.

540

z

.LIST BIN,LOC,SEQ

541
542
543
544
545
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

```
.TITLE TM03-TU45 CONTROL LOGIC TEST-PART II
:CZTUPAO
:25 MAY 78
:J.G. ADAMS/R. J. COLLINS

:REVISED JUN 1977 BY J.G.ADAMS ;++B CHANGED MODULE TYPEOUTS TO
; ++B REFLECT TM03 REGISTER SET

.MCALL .SACT11,.SEOP,$CATCH,$SAVE,$RESTORE,$CHAIN,$SCHNMODE
.NLIST MC
.LIST ME
.ENABLE ABS,AMA

:CONSOLE SWITCHES*****
:
:SW15: 1=HALT ON ERROR
:      0=CONTINUE
:SW14: 1=LOOP ON ERROR
:      0=CONTINUE
:SW13: 1=DO NOT PRINT ERRORS
:      0=PRINT ERRORS
:SW12: 1=HALT AT END OF PASS
:      0=CONTINUOUS CYCLE
:SW11: 1=INHIBIT ITERATIONS
:      0=DO ITERATIONS
:SW10: 1=HALT AT END OF EACH TEST
:      0=CONTINUE
:SW8: 1=NO WRAP DATA CHECK
:      0=DO WRAP DATA CHECK
:SW7: 1=NO WRAP STATUS CHECK
:      0=DO WRAP STATUS CHECK
:SW6: 1=SELECTABLE WRAP DATA PATTERN (IN SINGLE TEST)
:      0=AUTO PATTERNS
:SW0-5: SELECT TEST NUMBER :: 00=ALL TESTS
:IF HARDWARE SWR <15::00> = 177777 OR NOT AVAILABLE USE SOFTWARE SWR
```



```
622 ;REGISTER EQUIVS*****
623
624 000000 R0=%0
625 000001 R1=%1
626 000002 R2=%2
627 000003 R3=%3
628 000004 R4=%4
629 000005 R5=%5
630 000006 SP=%6
631 000007 PC=%7
632
633
634
635 ;ACT11 HOOK *****
636 000764 $SVPC= ;SAVE CURRENT LOCATION CTR
637 000046 .=46
638 000046 002310 .WORD SENDAD ;SET LOCATION 46
639 000052 000052 .=52
640 000052 000000 .WORD 0 ;SET LOCATION 52 = 0
641 000764 000764 .=$SVPC ;RESTORE LOCATION CTR
642
643 ;TTY INTERRUPT VECTOR*****
644
645 000060 .=60
646 000060 012306 .WORD TTINT ;TTY INTERRUPT HANDLER ADDRESS
647 000062 000340 .WORD 340 ;PRIORITY LEVEL 7
648
649 ;SOFTWARE SWITCH REGISTER*****
650 ;USED IF HARDWARE SWR = 177777, OR NOT AVAIL.
651 000176 000176 .=176
652 000176 000000 SWREG: .WORD 0
653
654
655 ;START ADDRESS*****
656 000200 000200 .=200
657 000200 000137 001200 JMP START ;PROGRAM START
658
659 ;RESTART ADDRESS*****
660 000210 000210 .=210
661 000210 000137 001670 JMP ST2
662
663 ;TM03 INTERRUPT VECTOR*****
664
665 000224 000224 .=224
666 000224 012276 MTINT ;TAPE INTERRUPT HANDLER ADDRESS
667 000226 000340 340
668
```

```

669
670          000510          .=510
671                                     ;MASS BUS REGISTER EQUIVS*****
672
673 000510 172440          C1:   172440
674 000512 172442          WC:   172442
675 000514 172444          BA:   172444
676 000516 172446          FC:   172446
677 000520 172450          CS:   172450
678 000522 172452          DS:   172452
679 000524 172454          ER:   172454
680 000526 172456          AS:   172456
681 000530 172460          CC:   172460
682 000532 172462          DB:   172462
683 000534 172464          MR:   172464
684 000536 172466          DT:   172466
685 000540 172470          SN:   172470
686 000542 172472          TC:   172472
687
688                                     ;ILLEGAL FUNCTION CODES
689
690 000544 005405          ILFT:  5405
691 000546 007415          7415
692 000550 016423          16423
693 000552 020437          20437
694 000554 022443          22443
695 000556 025447          25447
696 000560 031455          31455
697 000562 033465          33465
698 000564 036473          36473
699
700                                     ;CONSTANTS*****
701
702 000566 177776          PSW:   177776          ;PROCESSOR STATUS
703 000570 177570          SWR:   177570          ;SWITCH REGISTER
704 000572 177560          TKS:   177560          ;TTY READER STATUS
705 000574 177562          TKB:   177562          ;TTY READ BUFFER
706 000576 177564          TPS:   177564          ;TTY PUNCH STATUS
707 000600 177566          TPB:   177566          ;TTY PUNCH BUFFER
708 000602 177777          SERNUM: 177777          ;SERIAL NUMBER
709 000604 000011          DRVTP: 011             ;DRIVE TYPE
710 000606 000020          ITAMT: 20             ;ITERATION AMOUNT
711 000610 000224          VECT:  224            ;INTERRUPT VECTOR(RH)
712 000612 172440          REGS:  172440         ;STARTING REGISTER ADDRESS
  
```

```
713 ;FLAGS AND COUNTERS*****
714 ;NOTE ALL FLAGS AND COUNTERS ARE CLEARED ON STARTUP. PUT ANY
715 ;ADDITIONAL FLAGS BETWEEN STFLGS (START OF FLAGS) AND ENDFLG
716 ;(END OF FLAGS)
717
718 000614 STFLGS:
719 000614 000000 TOB: 0
720 000616 000000 TIB: 0
721 000620 000000 HDRFL: 0
722 000622 000000 EMADDR: 0
723 000624 000000 DRVN: 0
724 000626 000000 TRO0: 0
725 000630 000000 TRO1: 0
726 000632 000000 TRO2: 0
727 000634 000000 TRO3: 0
728 000636 000000 TRO4: 0
729 000640 000000 TRO5: 0
730 000642 000000 TRO6: 0
731 000644 000000 TRO7: 0
732 000646 000000 TR10: 0
733 000650 000000 TR11: 0
734 000652 000000 TR12: 0
735 000654 000000 TR13: 0
736 000656 000000 TR14: 0
737 000660 000000 TR15: 0
738 000662 000000 NRZOF: 0
739 000664 000000 SLVN: 0
740 000666 000000 PFLG: 0
741 000670 000000 RTRN: 0
742 000672 000000 ERADD: 0
743 000674 000000 TEMP1: 0
744 000676 000000 TEMP2: 0
745 000700 000000 TEMP3: 0
746 000702 000000 ITCNT: 0
747 000704 000000 SAV1: 0
748 000706 000000 SAV2: 0
749 000710 000000 SAV3: 0
750 000712 000000 SCOLP: 0
751 000714 000000 ITRLP: 0
752 000716 000000 EXFL: 0
753 000720 000000 ATAF: 0
754 000722 000000 SLAF: 0
755 000724 000000 SSCF: 0
756 000726 000000 ERRF: 0
757 000730 000000 ASF: 0
758 000732 000000 SCF: 0
759 000734 000000 TREF: 0
760 000736 000000 PEXFL: 0
761 000740 000000 STFLG: 0
762 000742 000000 LTADD: 0
763 000744 000000 T24FL: 0
764 000746 000000 ADDFL: 0
765 000750 000000 WAM: 0
766 000752 000000 FUN: 0
767 000754 000000 DATC: 0
768 000756 000000 WTAD: 0
```

```
769 000760 000000 DATAD: 0
770 000762 000000 RDAD: 0
771 000764 000000 W2FLG: 0
772 000766 000000 DERFL: 0
773 000770 000000 PREFL: 0
774 000772 000000 SERFL: 0
775 000774 000000 CRCNT: 0
776 000776 000000 UDES: 0
777 001000 000000 WPGFL: 0
778 001002 000000 PATRN: 0
779 001004 000000 STATF: 0
780 001006 000000 RDRVF: 0
781 001010 000000 RCDP: 0
782 001012 000000 STATC: 0
783 001014 000000 SKAT: 0
784 001016 000000 PCNTR: 0 ;PASS COUNTER
785 001020 000000 DCHKFL: .WORD 0 ;DATA CHECK FLAG 0/1 = CHECK/DO NOT CHECK
786 001022 000000 CRCFLG: .WORD 0 ;CRC CORRECTION TEST IN PROGRESS
787 001024
788
789 ;EXPT WRAP STATUS*****
790
791 001024 000000 WCS1: 0
792 001026 000000 WCS2: 0
793 001030 000000 WDS: 0
794 001032 000000 WER: 0
795
796 ;DATA PATTERN GENERATORS*****
797
798 001034 DATBL:
799 001034 005076 DATA0: DAT1 ;ALL ONE BITS
800 001036 005116 DATA1: DAT2 ;ALL ZERO BITS
801 001040 005122 DATA2: DAT3 ;ALTERNATING ONE/ZERO BITS
802 001042 005130 DATA3: DAT4 ;ALTERNATING PARITY CHARACTERS
803
804 ;CORE DUMP PATTERNS*****
805
806 001044 000005 WCDP2: 5
807 001046 000005 5
808 001050 000012 12
809 001052 000012 12
810 001054 000000 0
811 001056 000017 WCDPO: 17
812 001060 000017 17
813 001062 000017 17
814 001064 000017 17
815 001066 000000 0
```

```
816  
817 ;LOGIC TEST ENTRY TABLE*****  
818  
819 001070 000000 TSTTBL: 0  
820 001072 000000 0  
821 001074 002352 LT1  
822 001076 002352 LT1  
823 001100 002464 LT2  
824 001102 002464 LT2  
825 001104 002536 LT3  
826 001106 002536 LT3  
827 001110 002644 LT4  
828 001112 002644 LT4  
829 001114 002716 LT5  
830 001116 002716 LT5  
831 001120 003024 LT6  
832 001122 003024 LT6  
833 001124 003102 LT7  
834 001126 003102 LT7  
835 001130 003210 LT10  
836 001132 003210 LT10  
837 001134 003262 LT11  
838 001136 003262 LT11  
839 001140 003374 LT12  
840 001142 003374 LT12  
841 001144 003522 LT13  
842 001146 003522 LT13  
843 001150 003572 LT14  
844 001152 003572 LT14  
845 001154 003642 LT15  
846 001156 003642 LT15  
847 001160 003724 LT16  
848 001162 003724 LT16  
849 001164 004312 LT17  
850 001166 004312 LT17  
851 001170 004624 LT20  
852 001172 004624 LT20  
853 001174 002244 TADX: .WORD TEND  
854 001176 000020 TLAST: .WORD 20 ;CONTAINS # OF TESTS
```

```
855 .EVEN
856 ;PROGRAM START AND HOUSEKEEPING*****
857
858 001200 012706 000500 START: MOV #500,SP ;SET STACK POINTER
859 001204 013746 000004 MOV @#4,-(SP) ;SAVE ERROR TRAP
860 001210 013746 000006 MOV @#6,-(SP)
861 001214 012737 001240 000004 MOV #1$,@#4 ;SET TIME OUT TRAP TO GO TO 1$
862 001222 005037 000006 CLR @#6
863 001226 022777 177777 177334 CMP #177777,@SWR ;USE SOFTARE SWITCH IF SWR = 177777
864 001234 001402 BEQ 2$ ;OR TIMES OUT
865 001236 000404 BR 3$ ;OTHERWISE USE HARDWARE SWR
866 001240 022626 1$: CMP (SP)+,(SP)+ ;RESET STACK
867 001242 012737 000176 000570 2$: MOV #SWREG,SWR ;SET SWR = TO ADDRESS OF SOFTARE SWR
868 001250 012637 000006 3$: MOV (SP)+,@#6 ;RESTORE ERROR TRAP
869 001254 012637 000004 MOV (SP)+,@#4
870 001260 005027 CLR (PC)+ ;:CLEAR CHAIN INDICATOR
871 001262 000000 CHNFLG: .WORD 0 ;:CHAIN MODE INDICATOR
872 ;:1/0 = CHAIN/NOT CHAIN MODE
873 001264 022737 002310 000042 CMP #SENDAD,@#42 ;:BRANCH IF LOADED VIA ACT11 CHAIN MODE
874 001272 001404 BEQ 50$
875 001274 005737 000042 TST @#42 ;:BRANCH IF IN DUMP MODE
876 001300 001413 BEQ 52$
877 001302 000406 BR 51$
878 001304 012737 000176 000570 50$: MOV #SWREG,SWR ;:INVOKE SOFTWARE SWR
879 001312 012777 100000 177250 MOV #100000,@SWR ;:WITH HALT ON ERROR SET
880 001320 005237 001262 51$: INC CHNFLG ;:SET CHNFLG = CHAIN MODE
881 001324 000137 001714 JMP TSCD ;:GO TO CHAIN ADDRESS
882 001330 52$:
883 001330 122737 000006 000041 4$: CMPB #6,@#41 ;:BRANCH IF NOT LOADED VIA TMDP
884 001336 001004 BNE 5$
885 001340 012704 014471 MOV #MSG62,R4 ;ADVISE USER TO REMOVE MEDIA FROM UUT
886 001344 004737 012752 JSR PC,TTOUT
887 001350 012704 013674 5$: MOV #MSG1,R4
888 001354 004737 012752 JSR PC,TTOUT ;PRINT TITLE
889 001360 112737 000043 013674 MOVB #'#,MSG1 ;DO NOT PRINT TITLE ON RESTART
890 001366 012704 014365 MOV #MSG44,R4
891 001372 004737 012752 JSR PC,TTOUT ;REQUEST REGISTER ADDRESS
892 001376 013703 000612 MOV REGS,R3
893 001402 004737 013100 JSR PC,OCTP ;PRINT CURRENT ADDRESS
894 001406 012705 000612 MOV #REGS,R5 ;SET ADDRESS SAVE LOC
895 001412 012701 000007 MOV #7,R1 ;SET SIZE OF RESPONSE
896 001416 012702 176400 MOV #176400,R2 ;SET UPPER LIMIT
897 001422 012703 172300 MOV #172300,R3 ;SET LOWER LIMIT
898 001426 004737 012430 JSR PC,TTR ;GO GET RESPONSE
899 001432 012704 014407 MOV #MSG45,R4
900 001436 004737 012752 JSR PC,TTOUT ;REQUEST VECTOR
901 001442 013703 000610 MOV VECT,R3
902 001446 004737 013100 JSR PC,OCTP ;PRINT CURRENT VECTOR
903 001452 012705 000610 MOV #VECT,R5 ;SET ADDRESS SAVE LOC
904 001456 012701 000004 MOV #4,R1 ;SET SIZE OF RESPONSE
905 001462 012702 000224 MOV #224,R2 ;SET UPPER LIMIT
906 001466 012703 000150 MOV #150,R3 ;SET LOWER LIMIT
907 001472 004737 012430 JSR PC,TTR ;GO GET RESPONSE
908 001476 013700 000610 MOV VECT,R0 ;GET VECTOR
909 001502 012720 012276 MOV #MTINT,(R0)+ ;LOAD INTERRUPT ADDRESS IN VECTOR
910 001506 012710 000340 MOV #340,(R0) ;LOAD PRIORITY
```

```

911 001512 013700 000612      MOV      REGS,R0          ;GET START OF REGS
912 001516 012701 000016      MOV      #16,R1         ;SET NUMBER OF REGS
913 001522 012702 000510      MOV      #C1,R2         ;GET START OF TABLE
914 001526 010022              6$:      MOV      R0,(R2)+      ;BUILD TABLE
915 001530 062700 000002      ADD      #2,R0          ;BUMP ADDRESS
916 001534 005301              DEC      R1              ;SEE IF DONE
917 001536 001373              BNE      6$              ;IF NOT: BR
918 001540 012702 000614      MOV      #STFLGS,R2     ;
919 001544 012700 000210      MOV      #ENDFLG-STFLGS,R0 ;GET # OF FLAGS TO CLEAR
920 001550 006200              ASR      R0              ;FORM COUNT
921 001552 005022              7$:      CLR      (R2)+          ;CLEAR FLAGS + COUNTERS
922 001554 005300              DEC      R0
923 001556 001375              BNE      7$
924 001560 012704 014431      MOV      #MSG57,R4      ;REQUEST TM03 DRIVE #
925 001564 004737 012752      JSR      PC,TTOUT
926 001570 013703 000624      MOV      DRVN,R3        ;GET CURRENT DRIVE
927 001574 004737 013100      JSR      PC,OCTP        ;AND TYPE IT
928 001600 012705 000624      MOV      #DRVN,R5      ;TTR ROUTINE RETURNS DRIVE TO (R5)
929 001604 012701 000002      MOV      #2,R1         ;LIMIT RESPONSE TO 1 CHARACTER
930 001610 012702 000007      MOV      #7,R2         ;BETWEEN 0 AND 7
931 001614 012703 000000      MOV      #0,R3
932 001620 004737 012430      JSR      PC,TTR         ;GET RESPONSE & PUT IN DRVN
933 001624 012704 014447      MOV      #MSG58,R4      ;REQUEST SLAVE #
934 001630 004737 012752      JSR      PC,TTOUT
935 001634 013703 000664      MOV      SLVN,R3        ;GET CURRENT SLAVE #
936 001640 004737 013100      JSR      PC,OCTP        ;AND TYPE IT
937 001644 012705 000664      MOV      #SLVN,R5      ;TTR ROUTINE RETURNS REPONSE TO (R5)
938 001650 012701 000002      MOV      #2,R1         ;LIMIT RESPONSE TO 1 CHARACTER
939 001654 012702 000007      MOV      #7,R2         ;BETWEEN 0-7
940 001660 012703 000000      MOV      #0,R3
941 001664 004737 012430      JSR      PC,TTR         ;GET RESPONSE & PUT IT IN SLVN
942
943
944 001670 012706 000500      ; START 210
945 001674 005037 001006      ST2:     MOV      #500,SP ;SET STACK PTR
946 001700 005037 001016      CLR      RDRVF         ;CLEAR READ REVERSE FLAG
947 001704 005037 001022      CLR      PCNTR        ;
948 001710 004737 013532      CLR      CRCFLG       ;SET CRC FLAG = CRC NOT IN PROGRESS
                          JSR      PC,GTSWR ;GET SOFTWARE SWITCHES

```

```

949
950           ;TEST SCHEDULAR*****
951
952 001714 052777 000100 176650 TSCD:  BIS   #100,@TKS   ;SET KEYBOARD IE BIT
953 001722 005037 001000           CLR   WPGFL     ;CLEAR WRAP PATRN FLAG
954 001726 005037 000740           CLR   STFLG     ;CLEAR SINGLE TEST FLAG
955 001732 017700 176632           MOV   @SWR,RO
956 001736 042700 177700           BIC   #177700,RO ;BRANCH IF SINGLE TEST SELECTED
957 001742 001122           BNE   STSCD     ;GO SELECT SINGLE TEST
958 001744 005737 001262           TST   CHNFLG   ;;BRANCH IF NOT IN CHAIN MODE
959 001750 001457           BEQ   TSCDA
960 001752 012737 177777 000624           MOV   #-1,DRVN ;;INITIALIZE DRIVE #
961 001760 012737 177777 000664 NXTDRV: MOV   #-1,SLVN ;;INITIALIZE SLAVE #
962 001766 012777 000040 176524 1$:   MOV   #40,@CS  ;;INIT CONTROLLER
963 001774 005237 000624           INC   DRVN     ;;STEP DRIVE #
964 002000 022737 000010 000624           CMP   #10,DRVN ;;EXIT IF ALL DRIVES TESTED
965 002006 001521           BEQ   $DONE    ;;FOR AVAILABILITY
966 002010 013777 000624 176502           MOV   DRVN,@CS ;;LOAD DRIVE #
967 002016 005777 176466           TST   @C1     ;;ACCESS DRIVE
968 002022 032777 010000 176470           BIT   #10000,@CS ;;BRANCH IF DRIVE NON EXISTANT
969 002030 001356           BNE   1$      ;;(NED = 1)
970 002032 005237 000664           NXTSLV: INC  SLVN ;;STEP SLAVE # AND BRANCH
971 002036 001011           BNE   1$      ;;IF NOT SLAVE 0
972 002040 005737 000624           TST   DRVN    ;;BRANCH IF NOT DRIVE # 0
973 002044 001006           BNE   1$
974 002046 122737 000006 000041           CMPB  #6,@#41  ;;BRANCH IF NOT TMDP
975 002054 001002           BNE   1$
976 002056 005237 000664           INC   SLVN    ;;STEP TO SLAVE # 1
977 002062 022737 000010 000664 1$:   CMP   #10,SLVN ;;BRANCH IF ALL SLAVES TESTED
978 002070 001733           BEQ   NXTDRV  ;;FOR AVAILABILITY
979 002072 013777 000664 176442           MOV   SLVN,@TC ;;LOAD SLAVE UNIT #
980 002100 032777 002000 176430           BIT   #2000,@DT ;;BRANCH IF SLAVE NOT
981 002106 001751           BEQ   NXTSLV ;;PRESENT (SPR = 0)
982 002110 012737 001070 000742 TSCDA: MOV   #TSTTBL,LTADD
983 002116 062737 000004 000742 TSCD0: ADD   #4,LTADD
984 002124 013737 000742 000714 TSCD1: MOV   LTADD,ITRLP
985 002132 062737 000002 000714           ADD   #2,ITRLP ;SET ITERATION ADDRESS
986 002140 005037 000620           CLR   HDRFL    ;CLEAR PRINT HEADER FLAG
987 002144 017700 176572           MOV   @LTADD,RO ;SET POINTER TO TEST
988 002150 000110           JMP   (RO)     ;GO TO TEST
989 002152 032777 002000 176410 TSCD2: BIT   #2000,@SWR ;SEE IF HALT ON TEST
990 002160 001403           BEQ   TSCD3   ;IF NOT: BR
991 002162 000000           HALT
992 002164 005037 001000           CLR   WPGFL    ;CLEAR WRAP DATA GENERATOR FLAG
993 002170 005737 000740           TSCD3: TST   STFLG   ;SE IF SINGLE TEST
994 002174 001750           BEQ   TSCD0   ;IF NOT: BR
995 002176 017700 176366           MOV   @SWR,RO
996 002202 042700 177700           BIC   #177700,RO ;MASK TEST NUMBER
997 002206 001642           BEQ   TSCD    ;IF SO: BR
998 002210 012737 000001 000740 STSCD: MOV   #1,STFLG ;SET SINGLE TEST FLAG
999 002216 023700 001176           CMP   TLAST,RO ;SEE IF EXCEEDED TESTS
1000 002222 002410           BLT   TEND    ;IF SO: BR
1001 002224 006300           ASL   RO
1002 002226 006100           ROL   RO      ;SET TABLE MODIFIER
1003 002230 012737 001070 000742           MOV   #TSTTBL,LTADD
1004 002236 060037 000742           ADD   RO,LTADD ;SET TEST POINTER
  
```


1005	002242	000730			BR	TSCD1		
1006	002244	005737	001262		TEND:	TST	CHNFLG	;BRANCH IF IN CHAIN MODE
1007	002250	001270				BNE	NXTSLV	
1008	002252	012704	014346		\$DONE:	MOV	#MSG41,R4	
1009	002256	004737	012752			JSR	PC,TTOUT	;PRINT END OF PASS
1010	002262	013703	001016			MOV	PCNTR,R3	
1011	002266	004737	013100			JSR	PC,OCTP	;PRINT PASS NUMBER
1012	002272	005000				CLR	R0	
1013	002274	005300			1\$:	DEC	R0	;DELAY WAITING FOR
1014	002276	001376				BNE	1\$;TTY TO FINISH
1015	002300	013700	000042			MOV	@#42,R0	;GET ACT11 RETURN ADDRESS
1016	002304	001405				BEQ	HERE	;BRANCH IF NOT ACT11
1017	002306	000005				RESET		
1018	002310	004710			\$ENDAD:	JSR	PC,(R0)	
1019	002312	000240				NOP		
1020	002314	000240				NOP		
1021	002316	000240				NOP		
1022	002320	000240			HERE:	NOP		
1023	002322	005737	001262			TST	CHNFLG	;BRANCH IF IN CHAIN MODE
1024	002326	001005				BNE	TENDX	
1025	002330	032777	010000	176232		BIT	#10000,@SWR	;SEE IF HALT ON PASS
1026	002336	001001				BNE	TENDX	;IF NOT: BR
1027	002340	000000				HALT		
1028	002342	005237	001016		TENDX:	INC	PCNTR	;BUMP PASS COUNTER
1029	002346	000137	001714			JMP	TSCD	;RESTART
1030								

```

1031                                     ;THESE TESTS CHECK DATA FORMATTING
1032                                     ;AND TRANSFER THROUGH THE TM03 WRAP AROUND MODES
1033
1034                                     ;LOGIC TEST 1: WRAP 3, NRZ, NORMAL ODD *****
1035
1036 002352 012737 004270 001024 LT1:  MOV    #4270,WCS1      ;SET EXPT CS1
1037 002360 012737 000100 001026      MOV    #100,WCS2      ;SET EXPT CS2
1038 002366 012737 010600 001030      MOV    #10600,WDS    ;SET EXPT DS
1039 002374 012737 000000 001032      MOV    #0,WER        ;SET EXPT ER
1040 002402 012737 014565 000622      MOV    #MSLT1,EMADDR ;SET HEADER
1041 002410 012737 001700 000776      MOV    #1700,UDES    ;SET NRZ,NORMAL, ODD
1042 002416 005037 001002                LT1A: CLR    PATRN      ;POINT TO PATTERN 0
1043 002422 012737 002430 000712      MOV    #LT1B,SCOLP   ;SET SCOPE ADDRESS
1044 002430 004737 005256                LT1B: JSR    PC,WAM3    ;GO DO WRAP 3
1045 002434 005237 001002                INC    PATRN         ;BUMP PATTERN POINTER
1046 002440 032737 000004 001002      BIT    #4,PATRN      ;SEE IF DONE
1047 002446 001770                BEQ    LT1B          ;IF NOT: BR
1048 002450 004737 012052                JSR    PC,ITER       ;GO SEE IF ITERATIONS
1049 002454 005037 001006                CLR    RDRVF        ;CLEAR READ REVERSE FLAG
1050 002460 000137 002152                JMP    TSCD2        ;RETURN TO SCHEDULAR
1051
1052                                     ;LOGIC TEST 2: WRAP 3, PE, NORMAL, ODD*****
1053
1054 002464 000240                LT2:  NOP
1055 002466 012737 004270 001024 LT2A:  MOV    #4270,WCS1      ;SET EXPT CS1
1056 002474 012737 000100 001026      MOV    #100,WCS2      ;SET EXPT CS2
1057 002502 012737 010640 001030      MOV    #10640,WDS    ;SET EXPT DS
1058 002510 012737 000000 001032      MOV    #0,WER        ;SET EXPT WER
1059 002516 012737 014633 000622      MOV    #MSLT2,EMADDR ;SET HEADER
1060 002524 012737 002300 000776      MOV    #2300,UDES    ;SET PE, NORMAL, ODD
1061 002532 000137 002416                JMP    LT1A         ;EXECUTE TEST SEQUENCE
1062
1063                                     ;LOGIC TEST 3: WRAP 2, NRZ, NORMAL, ODD*****
1064
1065 002536 012737 004260 001024 LT3:  MOV    #4260,WCS1      ;SET EXPT CS1
1066 002544 012737 000100 001026      MOV    #100,WCS2      ;SET EXPT CS2
1067 002552 012737 010600 001030      MOV    #10600,WDS    ;SET EXPT DS
1068 002560 012737 000000 001032      MOV    #0,WER        ;SET EXPT WER
1069 002566 012737 014700 000622      MOV    #MSLT3,EMADDR ;SET HEADER
1070 002574 012737 001700 000776      MOV    #1700,UDES    ;SET TO NRZ,NORMAL, ODD
1071 002602 005037 001002                LT3A: CLR    PATRN      ;POINT TO PATTERN 0
1072 002606 012737 002614 000712      MOV    #LT3B,SCOLP   ;SET SCOPE ADDRESS
1073 002614 004737 005212                LT3B: JSR    PC,WAM2    ;GO DO WRAP 2
1074 002620 005237 001002                INC    PATRN         ;BUMP POINTER
1075 002624 032737 000004 001002      BIT    #4,PATRN      ;SEE IF DONE
1076 002632 001770                BEQ    LT3B          ;IF NOT: BR
1077 002634 004737 012052                JSR    PC,ITER       ;GO SEE IF ITERATIONS
1078 002640 000137 002152                JMP    TSCD2        ;RETURN TO SCHEDULAR

```

```

1079
1080
1081
1082 002644 000240
1083 002646 012737 004260 001024
1084 002654 012737 000100 001026
1085 002662 012737 010640 001030
1086 002670 012737 000000 001032
1087 002676 012737 014746 000622
1088 002704 012737 002300 000776
1089 002712 000137 002602
1090
1091
1092
1093 002716 012737 004260 001024
1094 002724 012737 000100 001026
1095 002732 012737 010600 001030
1096 002740 012737 000000 001032
1097 002746 012737 015013 000622
1098 002754 012737 001700 000776
1099 002762 005037 001002
1100 002766 012737 002774 000712
1101 002774 004737 005202
1102 003000 005237 001002
1103 003004 032737 000004 001002
1104 003012 001770
1105 003014 004737 012052
1106 003020 000137 002152
1107
1108
1109
1110 003024 000240
1111 003026 004737 011550
1112 003032 012737 004260 001024
1113 003040 012737 000100 001026
1114 003046 012737 010640 001030
1115 003054 012737 000000 001032
1116 003062 012737 015061 000622
1117 003070 012737 002300 000776
1118 003076 000137 002762

;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

```

```
1119
1120
1121                ;LOGIC TEST 7: WRAP 0, NRZ,NORMAL, ODD*****
1122 003102 012737 144260 001024 LT7:  MOV    #144260,WCS1    ;SET EXPT CS1
1123 003110 012737 000100 001026      MOV    #100,WCS2      ;SET EXPT CS2
1124 003116 012737 150600 001030      MOV    #150600,WDS    ;SET EXPT DS
1125 003124 012737 000200 001032      MOV    #200,WER       ;SET EXPT ER
1126 003132 012737 015126 000622      MOV    #MSLT7,EMADDR  ;SET HEADER
1127 003140 012737 001700 000776      MOV    #1700,UDES     ;SET NRZ, NORMAL, ODD
1128 003146 005037 0010G2 LT7A:  CLR    PATRN        ;POINT TO PATTERN 0
1129 003152 012737 003160 000712      MOV    #LT7B,SCOLP    ;SET SCOPE ADDRESS
1130 003160 004737 005136 LT7B:  JSR    PC,WAMO      ;GO DO WRAP 0
1131 003164 005237 001002      INC    PATRN          ;BUMP POINTER
1132 003170 032737 000004 001002      BIT    #4,PATRN       ;SEE IF DONE
1133 003176 001770      BEQ    LT7B           ;IF NOT: BR
1134 003200 004737 012052      JSR    PC,ITER        ;GO SEE IF ITERATIONS
1135 003204 000137 002152      JMP    TSCD2          ;RETURN TO SCHEDULAR
1136
1137                ;LOGIC TEST 10: WRAP 0, PE, NORMAL, ODD*****
1138
1139 003210 000240 LT10:  NOP
1140 003212 012737 004260 001024 LT10A: MOV    #4260,WCS1    ;SET EXPT CS1
1141 003220 012737 000100 001026      MOV    #100,WCS2      ;SET EXPT CS2
1142 003226 012737 010640 001030      MOV    #10640,WDS     ;SET EXPT DS
1143 003234 012737 000000 001032      MOV    #0,WER         ;SET EXPT ER
1144 003242 012737 015174 000622      MOV    #MSLT10,EMADDR ;SET HEADER
1145 003250 012737 002300 000776      MOV    #2300,UDES     ;SET PE, NORMAL, ODD
1146 003256 000137 003146      JMP    LT7A           ;GO EXECUTE TEST SEQUENCE
```

```

1147                                     ;LOGIC TEST 11: CORE DUMP WRITE, WAM2*****
1148
1149 003262 012737 004260 001024 LT11:  MOV    #4260,WCS1      ;SET EXPT CS1
1150 003270 012737 000100 001026      MOV    #100,WCS2      ;SET EXPT CS2
1151 003276 012737 010600 001030      MOV    #10600,WDS     ;SET EXPT DS
1152 003304 012737 000000 001032      MOV    #0,WER         ;SET EXPT ER
1153 003312 012737 015242 000622      MOV    #MSLT11,EMADDR ;SET HEADER
1154 003320 012737 001720 000776      MOV    #1720,UDES     ;SET NRZ, CORE DUMP, ODD
1155 003326 005037 001002                CLR    PATRN          ;POINT TO PATTERN 0
1156 003332 012737 003340 000712      MOV    #LT11A,SCOLP   ;SET SCOPE ADDRESS
1157 003340 004737 005212                LT11A: JSR    PC,WAM2   ;GO DO WAM 2
1158 003344 022737 000002 001002      CMP    #2,PATRN       ;SEE IF DONE
1159 003352 001404                BEQ    LT11X          ;IF SO: BR
1160 003354 012737 000002 001002      MOV    #2,PATRN       ;SELECT PATTERN 2
1161 003362 000766                BR     LT11A          ;CONTINUE
1162 003364 004737 012052                LT11X: JSR    PC,ITER  ;GO SEE IF ITERATIONS
1163 003370 000137 002152                JMP    TSCD2         ;RETURN TO SCHEDULES
1164
1165                                     ;LOGIC TEST 12: CORE DUMP READ, WAM 3*****
1166
1167 003374 012737 004270 001024 LT12:  MOV    #4270,WCS1      ;SET EXPT CS1
1168 003402 012737 000100 001026      MOV    #100,WCS2      ;SET EXPT CS2
1169 003410 012737 010600 001030      MOV    #10600,WDS     ;SET EXPT DS
1170 003416 012737 000000 001032      MOV    #0,WER         ;SET EXPT ER
1171 003424 012737 015313 000622      MOV    #MSLT12,EMADDR ;SET HEADER
1172 003432 012737 001720 000776      MOV    #1720,UDES     ;SELECT NRZ, CORE DUMP, ODD
1173 003440 005037 001002                CLR    PATRN          ;SELECT PATTERN 0
1174 003444 012737 003460 000712      MOV    #LT12A,SCOLP   ;SET SCOPE ADDRESS
1175 003452 012737 001056 001010      MOV    #WCDP0,RCDP    ;POINT TO PATTERN 0
1176 003460 004737 005256                LT12A: JSR    PC,WAM3   ;GO DO WAM3
1177 003464 022737 000002 001002      CMP    #2,PATRN       ;SEE IF DONE
1178 003472 001407                BEQ    LT12X          ;IF SO: BR
1179 003474 012737 000002 001002      MOV    #2,PATRN       ;SELECT PATTERN 2
1180 003502 012737 001044 001010      MOV    #WCDP2,RCDP    ;POINT TO PATTERN 2
1181 003510 000763                BR     LT12A          ;CONTINUE
1182 003512 004737 012052                LT12X: JSR    PC,ITER  ;GO SEE IF ITERATION
1183 003516 000137 002152                JMP    TSCD2         ;RETURN TO SCHEDULE

```

```
1184  
1185 ;LOGIC TEST 13: EVEN PARITY WRITE: WAM 1(M8933)*****  
1186  
1187 003522 012737 004260 001024 LT13: MOV #4260,WCS1 ;SET EXPT CS1  
1188 003530 012737 000100 001026 MOV #100,WCS2 ;SET EXPT CS2  
1189 003536 012737 010600 001030 MOV #10600,WDS ;SET EXPT DS  
1190 003544 012737 000000 001032 MOV #0,WER ;SET EXPT ER  
1191 003552 012737 015363 000622 MOV #MSLT13,EMADDR ;SET HEADER  
1192 003560 012737 001710 000776 MOV #1710,UDES ;SET NRZ, NORMAL, EVEN  
1193 003566 000137 002762 JMP LT5A ;GO EXECUTE WAM 1
```

```
1194  
1195 ;LOGIC TEST 14: EVEN PARITY READ: WAM 0(M8933 M8934)*****  
1196  
1197 003572 012737 144260 001024 LT14: MOV #144260,WCS1 ;SET EXPT CS1  
1198 003600 012737 000100 001026 MOV #100,WCS2 ;SET EXPT CS2  
1199 003606 012737 150600 001030 MOV #150600,WDS ;SET EXPT DS  
1200 003614 012737 000200 001032 MOV #200,WER ;SET EXPT ER  
1201 003622 012737 015444 000622 MOV #MSLT14,EMADDR ;SET HEADER  
1202 003630 012737 001710 000776 MOV #1710,UDES ;SET NRZ, NORMAL, EVEN  
1203 003636 000137 003146 JMP LT7A ;GO DO WAM 0
```

```
1204  
1205 ;LOGIC TEST 15: READ REVERSE: WAM 3(M8906)*****  
1206  
1207 003642 012737 004276 001024 LT15: MOV #4276,WCS1 ;SET EXPT CS1  
1208 003650 012737 000100 001026 MOV #100,WCS2 ;SET EXPT CS2  
1209 003656 012737 010640 001030 MOV #10640,WDS ;SET EXPT DS  
1210 003664 012737 000000 001032 MOV #0,WER ;SET EXPT ER  
1211 003672 012737 015523 000622 MOV #MSLT15,EMADDR ;SET HEADER  
1212 003700 012737 002300 000776 MOV #2300,UDES ;SELECT PE,NORMAL,ODD  
1213 003706 000240 NOP  
1214 003710 000240 NOP  
1215 003712 012737 000001 001006 MOV #1,RDRVF ;SET READ REVERSE FLAG  
1216 003720 000137 002416 JMP LT1A ;GO DO WAM 3, REVERSE  
1217
```

```

1218
1219 ;LOGIC TEST 16: CRC CORRECTION-SINGLE TRACK,EVERY FRAME
1220 ;THIS IS A TEST OF THE CRC CORRECTION LOGIC. THE TEST WRITES
1221 ;A KNOWN PATTERN (ALL 1'S , ALL 0'S & 125252) WITH A DATA BIT
1222 ;ALTERED IN EACH OF THE DATA TRACKS. THIS TEST INSURES THAT A
1223 ;CRC CORRECTABLE ERROR IS CORRECTED.
1224 ;THE TEST PROCEEDS AS FOLLOWS:
1225
1226 ; STEP A WRITE A KNOWN PATTERN USING WRAP AROUND MODE 0
1227
1228 ; STEP B REWRITE THE PATTERN ABOVE WITH DATA BIT(S) MODIFIED
1229 ; IN TRACKS SPECIFIED BY CRCPAT USING WRAP AROUND MODE 4
1230 ; THIS WILL GENERATE A CRC ERROR
1231
1232 ; STEP C EXECUTE A READ REVERSE USING WRAP AROUND MODE 3
1233
1234 ; STEP D REPEAT STEP B ABOVE. UPON COMPLETION THE DATA READ
1235 ; BACK WILL MATCH THE DATA WRITTEN IN STEP A.
1236
1237
1238 003724 005037 001002 LT16: CLR PATRN ;SELECT PATTERN # 0 (ALL 1'S)
1239 003730 012737 000401 004302 MOV #401,CRCPAT ;SELECT BITS TO BE ALTERED (TRACK 1)
1240 003736 012737 015570 000622 MOV #MSLT16,EMADDR ;SET ERROR MESSAGE HEADER
1241 003744 012737 003752 000712 MOV #LT16A,SCOLP ;SET SCOPE LOOP
1242
1243 003752 012737 144260 001024 LT16A: MOV #144260,WCS1 ;SET EXPECTED VALUES UPON COMPLETION
1244 003760 012737 000100 001026 MOV #100,WCS2
1245 003766 012737 150600 001030 MOV #150600,WDS
1246 003774 012737 000200 001032 MOV #200,WER
1247 004002 012737 001700 000776 MOV #1700,UDES ;SET UNIT DESCRIPTION-NRZ,800BPI,ODD
1248 ;PARITY & PDP11 NORMAL MODE
1249 004010 005037 001022 CLR CRCFLG ;CLEAR CRC CORRECTION FLAG
1250 004014 004737 005136 JSR PC,WAMO ;DO A WRAP 0 --- STEP A
1251 004020 012737 000001 001022 MOV #1,CRCFLG ;SET CRC ERROR CORRECTION IN PROGRESS
1252 004026 013737 017334 004304 MOV WBUFF,CRCDAT ;GET DATA WRITTEN BY WRAP 0
1253 ;XOR CRCPAT WITH CRCDAT
1254 004034 013737 004304 004306 MOV CRCDAT,XORDAT ;GET DATA TO BE MODIFIED
1255 004042 013737 004302 004310 MOV CRCPAT,XORPAT ;GET MODIFIER
1256 004050 043737 004302 004306 BIC CRCPAT,XORDAT ;CLEAR SET BITS IN DATA TO BE MODIFIED
1257 004056 043737 004304 004310 BIC CRCDAT,XORPAT ;CLEAR SETTING BITS
1258 004064 053737 004310 004306 BIS XORPAT,XORDAT ;SET CLEAR BITS IN DATA TO BE MODIFIED
1259 004072 013737 004306 004304 MOV XORDAT,CRCDAT ;RESTORE MODIFIED DATA
1260
1261 004100 012737 144260 001024 MOV #144260,WCS1 ;SET EXPECTED VALUES UPON COMPLETION
1262 004106 012737 000110 001026 MOV #110,WCS2 ;OF WRAP 4
1263 004114 012737 150600 001030 MOV #150600,WDS
1264 004122 012737 100300 001032 MOV #100300,WER
1265 004130 004737 005364 JSR PC,WAM4 ;DO A WRAP 4 --- STEP B
1266 004134 000240 NOP
1267 004136 012737 000001 001006 MOV #1,RDRVF ;SET TO READ REVERSE
1268 004144 012737 144276 001024 MOV #144276,WCS1 ;SET EXPECTED VALUES UPON COMPLETION
1269 004152 012737 000110 001026 MOV #110,WCS2
1270 004160 012737 150600 001030 MOV #150600,WDS
1271 004166 012737 001000 001032 MOV #1000,WER
1272 004174 004737 005256 JSR PC,WAM3 ;DO A WRAP 3 --- STEP C
1273 004200 000240 NOP

```

```

1274 004202 012737 004260 001024      MOV    #4260,WCS1
1275 004210 012737 000110 001026      MOV    #110,WCS2
1276 004216 012737 010600 001030      MOV    #10600,WDS
1277 004224 012737 000000 001032      MOV    #0,WER
1278 004232 005037 001022          CLR    CRCFLG      ;CLEAR CRC CORRECTION IN PROGRESS FLAG
1279 004236 004737 005364          JSR    PC,WAM4     ;GO TO WRAP 4 --- STEP D
1280
1281 004242 006337 004302          ASL    CRCPAT      ;SELECT NEXT TRACK TO BE ALTERED
1282 004246 103241          BCC    LT16A       ;CONTINUE FOR ALL TRACKS
1283 004250 012737 000401 004302      MOV    #401,CRCPAT ;RESET BITS TO TRACK 1
1284 004256 005237 001002          INC    PATRN       ;SELECT NEXT PATTERN
1285 004262 022737 000003 001002      CMP    #3,PATRN    ;BRANCH IF NOT DONE
1286 004270 001230          BNE    LT16A
1287 004272 004737 012052          JSR    PC,ITER     ;ITERATION LOOP
1288 004276 000137 002152          JMP    TSCD2       ;RETURN TO SCHEDULER
1289
1290
1291 004302 000000          CRCPAT: .WORD 0    ;CONTAINS BITS TO BE ALTERED
1292 004304 000000          CRCDAT: .WORD 0    ;CONTAINS DATA TO BE WRITTEN BY WRAP4
1293 004306 000000          XORDAT: .WORD 0    ;TEMPRARY STORAGE FOR XOR
1294 004310 000000          XORPAT: .WORD 0    ;TEMPOARY STORAGE FOR XOR
1295

```



```

1296
1297
1298 ;LOGIC TEST 17:CRC CORRECTION - MULTIPLE BAD TRACKS
1299 ;THIS TEST CHECKS THAT CRC ERROR CORRECTION IS NOT PERFORMED WHEN
1300 ;MULTIPLE BAD TRACKS ARE DETECTED.
1301
1302 004312 012737 000002 001002 LT17: MOV #2,PATRN ;SELECT PATTERN #2 (125125)
1303 004320 012737 001001 004302 MOV #1001,CRCPAT ;SELECT 2 BAD TRACKS
1304 004326 012737 015660 000622 MOV #MSLT17,EMADDR ;SET ERROR MESSAGE HEADER
1305 004334 012737 004342 000712 MOV #LT17A,SCOLP ;SET SCOPE LOOP ADDRESS
1306 004342 012737 144260 001024 LT17A: MOV #144260,WCS1 ;SET EXPECTED VALUES ON COMPLETION
1307 004350 012737 000100 001026 MOV #100,WCS2 ;OF WRAP 0 BELOW
1308 004356 012737 150600 001030 MOV #150600,WDS
1309 004364 012737 000200 001032 MOV #200,WER
1310 004372 012737 001700 000776 MOV #1700,UDES ;SET UNIT DESCRIPTION
1311 004400 005037 001022 CLR CRCFLG ;SET CRC CORRECTION NOT IN PROGRESS
1312 004404 004737 005136 JSR PC,WAM0 ;DO A WRAP 0 --- STEP A
1313 004410 000240 NOP
1314 004412 012737 000002 001022 MOV #2,CRCFLG ;SET CRC CORRECTION IN PROGRESS
1315 004420 013737 017334 004304 MOV WBUFF,CRCDAT ;GET DATA TO BE WRITTEN
1316 004426 043737 004302 004304 BIC CRCPAT,CRCDAT ;MODIFY DATA TO BE WRITTEN
1317 004434 012737 144260 001024 MOV #144260,WCS1 ;SET EXPECTED VALUES ON COMPLETION
1318 004442 012737 000110 001026 MOV #110,WCS2 ;OF WRAP 4 BELOW
1319 004450 012737 150600 001030 MOV #150600,WDS
1320 004456 012737 100300 001032 MOV #100300,WER
1321 004464 004737 005364 JSR PC,WAM4 ;DO A WRAP 4 --- STEP B
1322 004470 000240 NOP
1323 004472 012737 000001 001006 MOV #1,RDRVF ;SET READ REVERSE FLAG
1324 004500 012737 144276 001024 MOV #144276,WCS1 ;SET EXPECTED VALUES ON COMPLETION
1325 004506 012737 000110 001026 MOV #110,WCS2 ;OF WRAP 3 BELOW
1326 004514 012737 150600 001030 MOV #150600,WDS
1327 004522 012737 001000 001032 MOV #1000,WER
1328 004530 004737 005256 JSR PC,WAM3 ;DO A WRAP 3 --- STEP C
1329 004534 000240 NOP
1330 004536 012737 144260 001024 MOV #144260,WCS1 ;SET EXPECTED VALUES ON COMPLETION
1331 004544 012737 000110 001026 MOV #110,WCS2 ;OF WRAP 4 BELOW
1332 004552 012737 150600 001030 MOV #150600,WDS
1333 004560 012737 100100 001032 MOV #100100,WER
1334 004566 005037 001022 CLR CRCFLG ;CLEAR CRC IN PROGRESS FLAG
1335 004572 013701 004304 MOV CRCDAT,R1 ;GET DATA THAT WAS WRITTEN IN STEP B
1336 004576 012703 017334 MOV #WBUFF,R3 ;SET START OF WRITE BUFFER
1337 004602 004737 005102 JSR PC,DAT1A ;GO SET WRITE BUFFER
1338 004606 004737 005364 JSR PC,WAM4 ;GO DO A WRAP 4 --- STEP D
1339 004612 000240 NOP
1340 004614 004737 012052 JSR PC,ITER ;ITERATE TEST
1341 004620 000137 002152 JMP TSCD2 ;RETURN TO SCHEDULER

```

```
1342
1343 ;LOGIC TEST 20: READ REVERSE,NRZ,WRAP 3
1344 ;THIS TEST TESTS THAT A CRC ERROR OCCURS AFTER A READ REVERSE USING
1345 ;WRAP AROUND MODE 3 IN NRZ MODE
1346
1347 004624 005037 001002 LT20: CLR PATRN ;SET PATTERN # 0 (ALL 1'S)
1348 004630 012737 144276 001024 MOV #144276,WCS1 ;SET EXPECTED VALUES ON COMPLETION
1349 004636 012737 000100 001026 MOV #100,WCS2
1350 004644 012737 150600 001030 MOV #150600,WDS
1351 004652 012737 100000 001032 MOV #100000,WER
1352 004660 012737 015744 000622 MOV #MSLT20,EMADDR ;SET ERROR MESSAGE ADDRESS
1353 004666 012737 001700 000776 MOV #1700,UDES ;SET UNIT DESCRIPTION
1354 004674 012737 004702 000712 MOV #LT20A,SCOLP ;SET SCOPE LOOP ADDRESS
1355 004702 012737 000001 001006 LT20A: MOV #1,RDRVF ;SET READ REVERSE FLAG
1356 004710 012737 000001 001020 MOV #1,DCHKFL ;SET DATA CHECK FLAG TO NOT CHACK DATA
1357 004716 004737 005256 JSR PC,WAM3
1358 004722 005037 001006 CLR RDRVF ;CLEAR READ REVERSE FLAG
1359 004726 005037 001020 CLR DCHKFL ;CLEAR DATA CHECK FLAG
1360 004732 004737 012052 JSR PC,ITER
1361 004736 000137 002152 JMP TSCD2 ;RETURN TO SCHEDULER
1362
```

```

1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376 004742 005737 000740          DSUP:  TST      STFLG          ;SEE IF SINGLE TEST
1377 004746 001434                    BEQ      DSO          ;IF NOT: BR
1378 004750 032777 000100 173612    BIT      #100,@SWR    ;SEE IF SELECT PATTERN
1379 004756 001430                    BEQ      DSO          ;IF NOT: BR
1380 004760 012704 016454                    MOV      #WMSG3,R4
1381 004764 004737 012752                    JSR      PC,TTOUT     ;REQUEST PATTERN NUMBER
1382 004770 013703 001002                    MOV      PATRN,R3
1383 004774 004737 013100                    JSR      PC,OCTP     ;PRINT PATTERN NUMBER
1384 005000 012705 001002                    MOV      #PATRN,R5   ;GET ADDRESS OF PATRN ENTRY
1385 005004 012701 000002                    MOV      #2,R1       ;SET SIZE OF ENTRY
1386 005010 012702 000003                    MOV      #3,R2       ;SET UPPER LIMIT
1387 005014 012703 000000                    MOV      #0,R3       ;SET LOWER LIMIT
1388 005020 004737 012430                    JSR      PC,TTR      ;GO GET PATTERN NUMBER
1389 005024 112737 000001 001001    MOVVB   #1,WPGFL+1   ;SET FLAG
1390 005032 113737 001002 001000    MOVVB   PATRN,WPGFL  ;SET PATTERN NUMBER
1391 005040 012703 017334          DS0:  MOV      #WBUF,R3 ;R3 = ADDR OF WRITE BUFFER
1392 005044 013701 001002          MOV      PATRN,R1   ;R1 = PATTERN SELECTOR
1393 005050 006301                    ASL      R1         ;MAKE PATTERN SELECTOR EVEN
1394 005052 004771 001034                    JSR      PC,@DATBL(R1) ;GO GENERATE PATTERN
1395 005056 012702 000202          DS3:  MOV      #202,R2 ;R2=BUFFER SIZE +2
1396 005062 012701 017746          MOV      #RBUF,R1  ;R1=READ DATA START
1397 005066 005021          DS4:  CLR      (R1)+   ;CLEAR BUFFER
1398 005070 005302                    DEC      R2         ;SEE IF DONE ALL
1399 005072 001375                    BNE     DS4         ;IF NOT: BR
1400 005074 000207                    RTS      PC         ;EXIT
1401
1402
1403
1404 005076 012701 177777          DAT1:  MOV      #-1,R1 ;R1=DATA
1405 005102 012702 000202          DAT1A: MOV      #202,R2 ;R2=WORD COUNT +2
1406 005106 010123          1$:  MOV      R1,(R3)+ ;LOAD BUFFER
1407 005110 005302                    DEC      R2         ;SEE IF DONE
1408 005112 001375                    BNE     1$         ;IF NOT: BR
1409 005114 000207                    RTS      PC         ;RETURN TO CALLER
1410
1411
1412
1413 005116 005001          DAT2:  CLR      R1     ;R1=DATA
1414 005120 000770                    BR      DAT1A      ;LOAD BUFFER
1415
1416
1417
1418 005122 012701 125125          DAT3:  MOV      #125125,R1 ;R1=DATA

```

```
1419 005126 000765          BR    DAT1A          ;LOAD BUFFER
1420
1421                          ;ALTERNATING PARITY CHARACTERS*****
1422
1423 005130 012701 177377    DAT4:  MOV    #177377,R1    ;R1=ALTERNATING PARITY DATA
1424 005134 000762          BR    DAT1A          ;GO LOAD BUFFER
1425
```

```

1426
1427                                     ;WRAP AROUND MODE 0 GLOBAL*****
1428
1429 005136 012737 000006 000750 WAM0:  MOV    #6,WAM           ;SET WAM NUMBER
1430 005144 012737 000060 000752 WAM01: MOV    #60,FUN          ;
1431 005152 005037 000754          CLR    DATC
1432 005156 012737 017334 000760      MOV    #WBUFF,DATAD      ;SET BUFFER ADDRESS
1433 005164 012737 017746 000762      MOV    #RBUFF,RDAD      ;SET POINTER TO READ BUFFER
1434 005172 004737 005430          JSR    PC,SETUP         ;GO SET UP
1435 005176 000137 006010          JMP    EXEC
1436
1437                                     ;WRAP AROUND MODE 1 WRITE BUFFER*****
1438
1439 005202 012737 000010 000750 WAM1:  MOV    #10,WAM
1440 005210 000755          BR     WAM01
1441
1442                                     ;WRAP AROUND MODE 2 BIT FIDDLER WRITE*****
1443
1444 005212 012737 000012 000750 WAM2:  MOV    #12,WAM
1445 005220 012737 000060 000752      MOV    #60,FUN
1446 005226 005037 000754          CLR    DATC
1447 005232 012737 017334 000760      MOV    #WBUFF,DATAD
1448 005240 012737 017746 000762      MOV    #RBUFF,RDAD
1449 005246 004737 005430 WAM2A: JSR    PC,SETUP
1450 005252 000137 006010      JMP    EXEC
1451
1452                                     ;WRAP AROUND MODE 3 BIT FIDDLER READ*****
1453
1454 005256 012737 000014 000750 WAM3:  MOV    #14,WAM           ;SET WAM NUMBER
1455 005264 012737 000070 000752      MOV    #70,FUN          ;SET FUNCTION
1456 005272 012737 017746 000760      MOV    #RBUFF,DATAD     ;SET BUFFER ADDRESS
1457 005300 012737 017334 000756      MOV    #WBUFF,WTAD      ;SET POINTER TO WRITE BUFFER
1458 005306 005737 001006          TST    RDRVF
1459 005312 001411          BEQ    WAM3A
1460 005314 062737 000376 000760      ADD    #376,DATAD
1461 005322 062737 000377 000756      ADD    #377,WTAD
1462 005330 012737 000076 000752      MOV    #76,FUN          ;SET READ REVERSE CODE
1463 005336 032737 000020 000776 WAM3A: BIT    #20,UDES
1464 005344 001403          BEQ    WAM3B
1465 005346 013737 001010 000756      MOV    RCDP,WTAD
1466 005354 004737 005430 WAM3B: JSR    PC,SETUP         ;GO SET UP
1467 005360 000137 006010      JMP    EXEC             ;GO EXECUTE
1468
1469                                     ;WRAP AROUND MODE 4 CRC CORRECTION
1470                                     ;CALL: JSR    PC,WAM4
1471
1472 005364 012737 000030 000750 WAM4:  MOV    #30,WAM           ;SET MAINTENANCE MODE FUNCTION = WRAP 4
1473 005372 012737 000060 000752      MOV    #60,FUN          ;SET TAPE FUNCTION = WRITE FWD
1474 005400 005037 000754          CLR    DATC
1475 005404 012737 004304 000760      MOV    #CRCDAT,DATAD    ;SET ADRS OF WRITE BUFFER
1476 005412 012737 017746 000762      MOV    #RBUFF,RDAD      ;SET READ BUFFER ADDRESS
1477 005420 004737 005430          JSR    PC,SETUP         ;GO SETUP REGISTERS
1478 005424 000137 006010      JMP    EXEC             ;GO EXECUTE

```

```

1479                                     ;REGISTER SETUP ROUTINE*****
1480
1481 005430 005737 001022          SETUP: TST      CRCFLG          ;DO NOT DO DATA SETUP NOR INIT
1482 005434 001004                BNE      1$              ;IF CRC CORRECTION IS IN PROGRESS
1483 005436 022737 000030 000750  CMP      #30,WAM        ;DO NOT INIT IF DOING WAM 4 --- STEP D
1484 005444 001004                BNE      2$              ;DO DRIVE CLEAR IF WAM4---STEP D
1485 005446 012777 000011 173034 1$:  MOV      #11,@C1
1486 005454 000412                BR       SET1A
1487 005456 005737 000740          2$:  TST      STFLG          ;SEE IF SINGLE TEST
1488 005462 001403                BEQ      SET0            ;IF NOT: BR
1489 005464 005737 001000          TST      WPGFL          ;SEE IF HAVE SELECTED PATTERN
1490 005470 001002                BNE      SET1            ;IF SO: BR
1491 005472 004737 004742          SET0: JSR      PC,DSUP    ;GO DO DATA SETUP
1492 005476 004737 012122          SET1: JSR      PC,INIT    ;INIT CONTROLLER,SELECT UNIT & DRIVE
1493                                     ;LOAD SLAVE DESC AND MOVE OFF BOT
1494 005502 012777 177400 173006  SET1A: MOV      #-400,@FC    ;SET FC=WCX2
1495 005510 032737 000020 000776  BIT      #20,UDES        ;SEE IF CORE DUMP
1496 005516 001403                BEQ      SET2            ;IF NOT: BR
1497 005520 012777 177000 172770  MOV      #-1000,@FC      ;SET FC=WCX4
1498 005526 012777 177600 172756  SET2: MOV      #-200,@WC   ;SET WC
1499 005534 013777 000760 172752  MOV      DATAD,@BA       ;SET BUS ADDRESS
1500 005542 032777 010000 172750  BIT      #10000,@CS      ;ASSURE DRIVE THERE
1501 005550 001417                BEQ      SP1            ;IF SO: BR
1502 005552 032777 020000 173010  BIT      #20000,@SWR     ;SEE IF PRINT ERRORS
1503 005560 001004                BNE      SP01           ;IF NOT: BR
1504 005562 012704 016475          MOV      #WMSG4,R4
1505 005566 004737 012752          JSR      PC,TTOUT        ;PRINT NON-EXISTANT DRIVE
1506 005572 032777 100000 172770  SP01: BIT      #100000,@SWR ;SEE IF HALT ON ERROR
1507 005600 001401                BEQ      SPO            ;IF NOT: BR
1508 005602 000000                HALT
1509 005604 000137 005476          SPO:  JMP      SET1        ;RESETUP
1510 005610 022737 000014 000750  SP1:  CMP      #14,WAM      ;SEE IF WAM 3
1511 005616 001031                BNE      SP1B           ;IF NOT: BR
1512 005620 117737 173132 000754  MOVB     @WTAD,DATC      ;GET FIRST CHAR
1513 005626 042737 177400 000754  BIC      #177400,DATC
1514 005634 000337 000754          SWAB     DATC
1515 005640 052737 000200 000754  BIS      #200,DATC       ;SET PARITY
1516 005646 005737 001006          TST      RDRVF          ;SEE IF READ REVERSE
1517 005652 001403                BEQ      SP1A           ;IF NOT: BR
1518 005654 005337 000756          DEC      WTAD           ;DECREMENT POINTER
1519 005660 000410                BR       SP1B
1520 005662 005237 000756          SP1A: INC      WTAD        ;BUMP POINTER
1521 005666 032737 000020 000776  BIT      #20,UDES        ;SEE IF CORE DUMP
1522 005674 001402                BEQ      SP1B           ;IF NOT: BR
1523 005676 005237 000756          INC      WTAD           ;BUMP POINTER AGAIN
1524 005702 053777 000776 172632  SP1B: BIS      UDES,@TC    ;SET UNIT DESCRIPTION (DEN,PAR,FMT)
1525 005710 052777 000001 172616  BIS      #1,@MR          ;SET MAINT MODE
1526 005716 053777 000750 172610  BIS      WAM,@MR         ;SET WAM
1527 005724 053777 000754 172602  BIS      DATC,@MR        ;SET DATA
1528 005732 013777 000752 172550  MOV      FUN,@C1         ;SET FUNCTION
1529 005740 032777 040000 172554  BIT      #40000,@DS      ;ASSURE NO ERROR
1530 005746 001002                BNE      SP3            ;IF ERROR: BR
1531 005750 000240                NOP
1532 005752 000207                RTS      PC              ;RETURN
1533 005754 032777 020000 172606  SP3:  BIT      #20000,@SWR   ;SEE IF PRINT ERRORS
1534 005762 001004                BNE      SP4            ;IF NOT: BR
    
```

```
1535 005764 012704 016436            MOV    #WMSG2,R4
1536 005770 004737 012752            JSR    PC,TTOUT            ;PRINT SETUP ERROR
1537 005774 032777 100000 172566 SP4: BIT    #100000,@SWR            ;SEE IF HALT ON ERROR
1538 006002 001401            BEQ    SP5                ;IF NOT: BR
1539 006004 000000            HALT
1540 006006 000207            SP5:  RTS    PC            ;RETURN
```

```

1541                                     ;EXECUTE WAM ROUTINE*****
1542
1543 006010 000240          EXEC:  NOP
1544 006012 000240          NOP
1545 006014 032777 000040 172512  BIT    #40,@MR
1546 006022 001403          BEQ    EX0          ;ASSURE MAINT CLOCK IS ZERO
1547 006024 042777 000040 172502  BIC    #40,@MR      ;IF NOT: CLEAR IT
1548 006032 022737 000010 000750  EX0:  CMP    #10,WAM    ;SEE IF WAM 1 OR 2 OR 3
1549 006040 003402          BLE    2$
1550 006042 000137 006454          1$:  JMP    EXW2         ;GO DO WAM 0
1551 006046 022737 000030 000750  2$:  CMP    #30,WAM    ;BRANCH IF WRAP AROUND MODE 4
1552 006054 001772          BEQ    1$
1553 006056 052777 000001 172424  EX1:  BIS    #1,@C1      ;SET GO BIT
1554 006064 005000          CLR    R0
1555 006066 012701 000002          MOV    #2,R1        ;SET DELAY
1556 006072 032777 100000 172442  EX1A: BIT    #100000,@TC ;SEE IF ALPHA
1557 006100 001404          BEQ    EX2          ;IF SO: BR
1558 006102 005300          DEC    R0
1559 006104 001372          BNE    EX1A        ;AWAIT ALPHA
1560 006106 005301          DEC    R1
1561 006110 001370          BNE    EX1A
1562 006112 005077 172450          EX2:  CLR    @PSW
1563 006116 012701 000400          MOV    #400,R1      ;SET NUMBER OF CLKS
1564 006122 032737 000020 000776  BIT    #20,UDES    ;SEE IF CORE DUMP
1565 006130 001402          BEQ    EX3          ;IF NOT: BR
1566 006132 012701 001000          MOV    #1000,R1     ;SET CLOCKS LWCX4
1567 006136 022737 000014 000750  EX3:  CMP    #14,WAM    ;SEE IF WAM 3
1568 006144 001010          BNE    1$          ;IF NOT: BR
1569 006146 032737 002000 000776  BIT    #2000,UDES  ;IS IT PE?
1570 006154 001016          BNE    EX5A        ;IF YES: BR
1571 006156 052777 000040 172350  BIS    #40,@MR      ;CLOCK UP
1572 006164 000412          BR     EX5A
1573 006166 032737 002000 000776  1$:  BIT    #2000,UDES  ;SEE IF PE
1574 006174 001404          BEQ    EX5          ;IF NOT PE: BR
1575 006176 006301          ASL    R1
1576 006200 062701 000246          ADD    #246,R1     ;SET TO ALLOW FOR PRE/POSTAMBLE
1577 006204 000402          BR     EX5A
1578 006206 062701 000010          EX5:  ADD    #10,R1      ;ADD CLOCKS FOR CRC AND LRC
1579 006212 022737 000014 000750  EX5A: CMP    #14,WAM    ;SEE IF WAM 3
1580 006220 001053          BNE    EX5C        ;IF NOT: BR
1581 006222 117700 172530          MOVB  @WTAD,R0
1582 006226 042700 177400          BIC    #177400,R0
1583 006232 005737 001006          TST    RDRVF
1584 006236 001403          BEQ    EX5A1       ;SEE IF REVERSE
1585 006240 005337 000756          DEC    WTAD        ;DEC POINTER
1586 006244 000416          BR     EX5B
1587 006246 005237 000756          EX5A1: INC    WTAD
1588 006252 032737 000020 000776  BIT    #20,UDES    ;SEE IF CORE DUMP
1589 006260 001410          BEQ    EX5B        ;IF NOT: BR
1590 006262 005237 000756          INC    WTAD        ;BUMP POINTER
1591 006266 005777 172464          TST    @WTAD      ;SEE IF END
1592 006272 001003          BNE    EX5B        ;IF NOT: BR
1593 006274 162737 000010 000756  SUB    #10,WTAD    ;RESTORE POINTER
1594 006302 052777 000040 172224  EX5B: BIS    #40,@MR      ;CLOCK UP
1595 006310 017702 172220          MOV    @MR,R2      ;READ MR
1596 006314 042702 177600          BIC    #177600,R2  ;MASK OUT DATA
  
```


1597	006320	000300				SWAB	R0		; POSITION DATA
1598	006322	022700	177000			CMF	#177000,R0		; SEE IF PATTERN 3
1599	006326	001404				BEQ	2\$; IF SO: BR
1600	006330	000240				NOP			
1601	006332	000240				NOP			
1602	006334	052700	000200		1\$:	BIS	#200,R0		; SET ODD PARITY
1603	006340	050002			2\$:	BIS	R0,R2		; LOAD NEW DATA
1604	006342	010277	172166			MOV	R2,@MR		; CLOCK DOWN AND LOAD NEW DATA
1605	006346	000426				BR	EX5D		
1606	006350	052777	000040	172156	EX5C:	BIS	#40,@MR		; CLOCK UP
1607	006356	042777	000040	172150		BIC	#40,@MR		; CLOCK DOWN
1608	006364	017700	172144			MOV	@MR,R0		; GET MR
1609	006370	000300				SWAB	R0		
1610	006372	032737	000010	000776		BIT	#10,UDES		; SEE IF EVEN PAR
1611	006400	001405				BEQ	EX5C0		; IF NOT: BR
1612	006402	010077	172354			MOV	R0,@RDAD		
1613	006406	005237	000762			INC	RDAD		
1614	006412	000402				BR	EX5C1		
1615	006414	110077	172342		EX5C0:	MOVB	R0,@RDAD		; PUT CHAR IN CORE
1616	006420	005237	000762		EX5C1:	INC	RDAD		
1617	006424	000240			EX5D:	NOP			
1618	006426	005301				DEC	R1		; SEE IF DONE CLKS
1619	006430	001270				BNE	EX5A		; IF NOT: BR
1620	006432	032777	000040	172074		BIT	#40,@MR		; CLOCK UP?
1621	006440	001403				BEQ	1\$; IF NOT: BR
1622	006442	042777	000040	172064		BIC	#40,@MR		; CLOCK DOWN
1623	006450	000137	011626		1\$:	JMP	EORP		; GO DO EOR

```

1624                                     ;EXECUTE WAM 0*****
1625
1626 006454 000240                      EXW2:  NOP
1627 006456 012737 006642 000670      MOV    #EXW2H,RTRN      ;SET INTERRUPT RETURN ADDRESS
1628 006464 012701 000200                MOV    #200,R1          ;SET NUMBER OF CLOCKS = FC/2
1629 006470 037737 002000 000776      BIT    #2000,UDES      ;SEE IF PE
1630 006476 00402                        BEQ    EXW2A            ;IF NOT: BR
1631 006500 012701 000100                MOV    #100,R1         ;ELSE SET CLKS = FC/4
1632 006504 012702 017746                EXW2A: MOV   #RBUF,R2    ;SET BUFFER ADDRESS
1633 006510 022737 000030 000750      CMP    #30,WAM         ;BRANCH IF NOT WRAP 4
1634 006516 001003                        BNE    1$
1635 006520 052777 000010 171772        BIS    #10,@CS         ;SET INHIBIT BUS ADDRESS INCREMENT
1636 006526 012777 000161 171754      1$:  MOV    #161,@C1     ;SET WRITE COMMAND AND GO
1637 006534 005077 172026                CLR    @PSW            ;ALLOW INTERRUPT
1638 006540 032777 000040 171766      EXW2B: BIT    #40,@MR
1639 006546 001774                        BEQ    EXW2B            ;AWAIT CLOCK UP
1640 006550 017722 171760                MOV    @MR,(R2)+       ;GET DATA
1641 006554 032777 000040 171752      EXW2C: BIT    #40,@MR
1642 006562 001374                        BNE    EXW2C            ;AWAIT CLOCK DOWN
1643 006564 017722 171744                MOV    @MR,(R2)+       ;GET DATA
1644 006570 005301                        DEC    R1               ;SEE IF DONE ALL
1645 006572 001362                        BNE    EXW2B            ;IF NOT: BR
1646 006574 012701 000003                EXW2E: MOV   #3,R1
1647 006600 005000                        CLR    R0               ;SET DELAY
1648 006602 005300                        EXW2F: DEC   R0
1649 006604 001376                        BNE    EXW2F
1650 006606 005301                        DEC    R1
1651 006610 001374                        BNE    EXW2F            ;DELAY
1652 006612 032777 020000 171750        BIT    #20000,@SWR     ;SEE IF ERROR PRINT
1653 006620 001004                        BNE    EXW2G            ;IF NOT: BR
1654 006622 012704 016700                MOV    #WMSG24,R4
1655 006626 004737 012752                JSR    PC,TTOUT        ;PRINT NO INTERUPT
1656 006632 005777 171732                EXW2G: TST   @SWR
1657 006636 100001                        BPL    EXW2H            ;SEE IF HALT ON ERROR
1658 006640 000000                        HALT
1659 006642 000240                        EXW2H: NOP
1660 006644 012701 017746                MOV    #RBUF,R1        ;GET START OF READ BUFFER
1661 006650 012700 000400                MOV    #400,R0         ;SET SIZE
1662 006654 010102                        MOV    R1,R2
1663 006656 012203                        EXW2J: MOV   (R2)+,R3
1664 006660 000303                        SWAB   R3
1665 006662 032737 000010 000776      BIT    #10,UDES        ;SEE IF EVEN PAR
1666 006670 001402                        BEQ    EXW2J0           ;IF NOT: BR
1667 006672 010321                        MOV    R3,(R1)+        ;SAVE PAR + DATA
1668 006674 000401                        BR     EXW2J1
1669 006676 110321                        EXW2J0: MOVB  R3,(R1)+  ;ASSEMBLE DATA IN BYTES
1670 006700 005300                        EXW2J1: DEC   R0
1671 006702 001365                        BNE    EXW2J            ;CONTINUE FOR ALL
1672 006704 032777 000200 171656        BIT    #200,@SWR       ;SEE IF STATUS CHECK
1673 006712 001002                        BNE    EXW2K            ;IF NOT: BR
1674 006714 004737 006742                JSR    PC,WSTCK        ;ELSE GO CHECK STATUS
1675 006720 000240                        EXW2K: NOP
1676 006722 032777 000400 171640        BIT    #400,@SWR       ;SEE IF DATA CHECK
1677 006730 001002                        BNE    EXW2X            ;IF NOT: BR
1678 006732 004737 007332                JSR    PC,DCHK         ;ELSE GO CHECK DATA
1679 006736 000240                        EXW2X: NOP

```

1680 006740 000207

RTS PC :EXIT

```

1681
1682                                     ;WRAP AROUND STATUS CHECK ROUTINE*****
1683
1684 006742 000240          WSTCK:  NOP
1685 006744 005037 000772      CLR     SERFL          ;CLEAR ERROR FLAG
1686 006750 005037 000620      CLR     HDRFL          ;CLEAR HEADER FLAG
1687 006754 022737 014633 000622  CMP     #MSLT2,EMADDR  ;SEE IF TEST 2
1688 006762 001404          BEQ     2$              ;IF SO: BR
1689 006764 022737 015523 000622  CMP     #MSLT15,EMADDR ;SEE IF TEST 15
1690 006772 001025          BNE     5$              ;IF NOT: BR
1691 006774 005737 001000      2$:  TST     WPGFL          ;SEE IF SINGLE PATTERN
1692 007000 001405          BEQ     3$              ;IF NOT: BR
1693 007002 122737 000003 001000  CMPB   #3,WPGFL       ;SEE IF PATTERN 3
1694 007010 001016          BNE     5$              ;IF NOT: BR
1695 007012 000404          BR      4$              ;ELSE GO DO EXPT CHANGE
1696 007014 022737 000003 001002  3$:  CMP     #3,PATRN     ;SEE IF PATTERN 3
1697 007022 001011          BNE     5$              ;IF NOT: BR
1698 007024 052737 140000 001024  4$:  BIS     #140000,WCS1 ;
1699 007032 052737 140000 001030  BIS     #140000,WDS   ;
1700 007040 052737 000100 001032  BIS     #100,WER      ;SET EXPT PARITY ERROR
1701 007046 012737 016522 000672  5$:  MOV     #WMSG6,ERADD  ;SET CODE=CS1
1702 007054 017702 171430      MOV     @C1,R2        ;GET RCVD CS1
1703 007060 013705 001024      MOV     WCS1,R5       ;GET EXPT CS1
1704 007064 004737 007206      JSR    PC,WSTG        ;GO CHK
1705 007070 012737 016547 000672  MOV     #WMSG6D,ERADD ;SET CODE=CS2
1706 007076 017702 171416      MOV     @CS,R2        ;SET RCVD CS2
1707 007102 013705 001026      MOV     WCS2,R5       ;GET EXPT CS2
1708 007106 053705 000624      BIS     DRVN,R5       ;SET DRIVE NUMBER IN EXPT CS2
1709 007112 004737 007206      JSR    PC,WSTG        ;GO CHK
1710 007116 012737 016555 000672  MOV     #WMSG6E,ERADD ;SET CODE=DS
1711 007124 017702 171372      MOV     @DS,R2        ;SET RCVD DS
1712 007130 013705 001030      MOV     WDS,R5        ;GET EXPT DS
1713 007134 004737 007206      1$:  JSR    PC,WSTG        ;GO CHK
1714 007140 012737 016562 000672  MOV     #WMSG6F,ERADD ;SET CODE=ER
1715 007146 017702 171352      MOV     @ER,R2        ;GET RCVD ER
1716 007152 000240          NOP
1717 007154 000240          NOP
1718 007156 013705 001032      MOV     WER,R5        ;GET EXPT ER
1719 007162 004737 007206      JSR    PC,WSTG        ;GO CHK
1720 007166 005737 000772      TST     SERFL          ;SEE IF ANY ERRORS
1721 007172 001456          BEQ     WSTX           ;IF NOT: BR
1722 007174 005777 171370      TST     @SWR          ;SEE IF HALT ON ERROR
1723 007200 100053          BPL     WSTX           ;IF NOT: BR
1724 007202 000000          HALT
1725 007204 000451          BR      WSTX           ;CONTINUE
1726 007206 000240          WSTG:  NOP
1727 007210 020205          CMP     R2,R5         ;SEE IF EXPT=RCVD
1728 007212 001446          BEQ     WSTX           ;IF SO: BR
1729 007214 005237 000772      INC     SERFL          ;SET ERROR FLAG
1730 007220 032777 020000 171342  BIT     #20000,@SWR   ;SEE IF PRINT ERRORS
1731 007226 001040          BNE     WSTX           ;IF NOT: BR
1732 007230 005737 000620      TST     HDRFL          ;SEE IF DONE HEADER
1733 007234 001010          BNE     WSTGO          ;IF SO: BR
1734 007236 013704 000622      MOV     EMADDR,R4
1735 007242 004737 012752      JSR    PC,TTOUT       ;PRINT TEST HEADER
1736 007246 012704 016664      MOV     #WMSG23,R4

```

```
1737 007252 004737 012752 JSR PC,TTOUT ;PRINT STATUS TAG
1738 007256 012737 000001 000620 WSTG0: MOV #1,HDRFL ;SET HEADER FLAG
1739 007264 013704 000672 MOV ERADD,R4
1740 007270 004737 012752 JSR PC,TTOUT ;PRINT CODE
1741 007274 012704 014223 MOV #MSG12,R4
1742 007300 004737 012752 JSR PC,TTOUT ;PRINT EXPT TAG
1743 007304 010503 MOV R5,R3
1744 007306 004737 013100 JSR PC,OCTP ;PRINT EXPT STATUS
1745 007312 012704 014232 MOV #MSG13,R4
1746 007316 004737 012752 JSR PC,TTOUT ;PRINT RCVD TAG
1747 007322 010203 MOV R2,R3
1748 007324 004737 013100 JSR PC,OCTP ;PRINT RCVD STATUS
1749 007330 000207 WSTX: RTS ;RETURN
1750
```

```

1751
1752                                     ;DATA CHECK ROUTINE*****
1753
1754 007332 000240          DCHK:  NOP
1755 007334 005737 001022      TST    CRCFLG          ;DO NOT DO A DATA CHACK
1756 007340 001402          BEQ    2$              ;IF CRC CORRECTION IN PROGRESS
1757 007342 000137 010172      1$:  JMP    DCHKX1
1758 007346 005737 001020      2$:  TST    DCHKFL          ;BRANCH IF DATA IS NOT TO BE CHECKED
1759 007352 001373          BNE    1$
1760 007354 005037 000620      CLR    HDRFL          ;CLEAR HEADER FLAG
1761 007360 005037 000766      CLR    DERFL          ;CLEAR DATA ERROR FLAG
1762 007364 005037 000774      CLR    CRCNT          ;CLEAR CHAR CNTR
1763 007370 032737 000010 000776  BIT    #10,UDES        ;SEE IF EVEN PARITY
1764 007376 001402          BEQ    DCHKA0          ;IF NOT: BR
1765 007400 000137 010214          JMP    DCHKE          ;ELSE GO CHECK EVEN
1766 007404 022737 000010 000750  DCHKA0:  CMP    #10,WAM        ;SEE IF WAM 1
1767 007412 001006          BNE    DCHKA          ;IF NOT: BR
1768 007414 032737 002000 000776  BIT    #2000,UDES     ;SEE IF PE
1769 007422 001402          BEQ    DCHKA          ;IF NOT: BR
1770 007424 000137 010716          JMP    PRCHK          ;GO CHK DATA
1771 007430 012700 177400          DCHKA:  MOV    #-400,R0  ;SET NUMBER OF CHARACTERS
1772 007434 022737 000012 000750  CMP    #12,WAM
1773 007442 001006          BNE    DCHKA1          ;IF NOT WRAP 2: BR
1774 007444 032737 000020 000776  BIT    #20,UDES
1775 007452 001402          BEQ    DCHKA1          ;IF NOT CORE DUMP: BR
1776 007454 012700 177000          MOV    #-1000,R0
1777 007460 022737 000030 000750  DCHKA1:  CMP    #30,WAM        ;BRANCH IF WRAP 4
1778 007466 001404          BEQ    1$
1779 007470 022737 000006 000750  CMP    #6,WAM          ;SEE IF WRAP 0
1780 007476 001007          BNE    DCHKA2          ;IF NOT: BR
1781 007500 012700 177744          1$:  MOV    #-34,R0      ;SET NUMBER OF CHARACTERS READ
1782 007504 012701 017344          MOV    #WBUF+10,R1   ;SET POINTER
1783 007510 005037 000766          CLR    DERFL          ;CLEAR DATA ERROR FLAG
1784 007514 000431          BR     DCHKB0
1785 007516 022737 000012 000750  DCHKA2:  CMP    #12,WAM        ;SEE IF WRAP 2
1786 007524 001021          BNE    DCHKB          ;IF NOT: BR
1787 007526 032737 002000 000776  BIT    #2000,UDES     ;SEE IF PE
1788 007534 001415          BEQ    DCHKB          ;IF NOT: BR
1789 007536 012700 177653          MOV    #-125,R0      ;POINT TO START OF DATA
1790 007542 012737 000001 000764  MOV    #1,W2FLG       ;SET WRAP 2 FLAG
1791 007550 004737 007570          JSR    PC,DCHKB       ;GO CHECK DATA
1792 007554 004737 011216          JSR    PC,W1DCHK      ;GO CHECK WRAP 1 DATA
1793 007560 005037 000764          CLR    W2FLG
1794 007564 000137 010154          JMP    DCHKX
1795 007570 005037 000766          DCHKB:  CLR    DERFL
1796 007574 012701 017334          MOV    #WBUF,R1      ;SET GOOD POINTER
1797 007600 012702 017746          DCHKB0:  MOV    #RBUF,R2     ;SET READ POINTER
1798 007604 032737 000020 000776  BIT    #20,UDES        ;SEE IF CORE DUMP
1799 007612 001416          BEQ    DCHKO          ;IF NOT: BR
1800 007614 022737 000012 000750  CMP    #12,WAM        ;SEE IF WAM 2
1801 007622 001011          BNE    DCHKD          ;IF NOT: BR
1802 007624 005737 001002          TST    PATRN          ;SEE IF PATTERN 0
1803 007630 001003          BNE    DCHKC          ;IF NOT: BR
1804 007632 012701 001056          MOV    #WCDPO,R1     ;SET CORE DUMP PATTERN 0
1805 007636 000404          BR     DCKO           ;GO CHECK DATA
1806 007640 012701 001044          DCHKC:  MOV    #WCDP2,R1 ;SET CORE DUMP WRITE PATTERN 2

```

```

1807 007644 000401          BR      DCHK0          ;GO CHECK DATA
1808 007646 000240          DCHKD: NOP
1809 007650 121112          DCHK0: CMPB      (R1),(R2)      ;SEE IF DATA OK
1810 007652 001466          BEQ      DCHK2          ;IF SO: BR
1811 007654 032777 020000 170706 BIT      #20000,@SWR          ;SEE IF PRINT ERRORS
1812 007662 001062          BNE      DCHK2          ;IF NOT: BR
1813 007664 005737 000620 TST      HDRFL          ;SEE IF DONE HEADER
1814 007670 001004          BNE      DCHK1          ;IF SO: BR
1815 007672 013704 000622 MOV      EMADDR,R4
1816 007676 004737 012752 JSR      PC,TTOUT        ;PRINT HEADER
1817 007702 005737 000766 DCHK1: TST      DERFL          ;SEE IF FIRST ERROR
1818 007706 001014          BNE      DCHK1A         ;IF NOT: BR
1819 007710 012704 016632 MOV      #WMSG16,R4
1820 007714 004737 012752 JSR      PC,TTOUT        ;PRINT DATA ERROR TAG
1821 007720 012704 017057 MOV      #WMSG32,R4
1822 007724 004737 012752 JSR      PC,TTOUT        ;PRINT PATRN TAG
1823 007730 013703 001002 MOV      PATRN,R3
1824 007734 004737 013100 JSR      PC,OCTP         ;PRINT PATTERN NUMBER
1825 007740 012737 000001 000620 DCHK1A: MOV     #1,HDRFL       ;SET HEADER FLAG
1826 007746 012737 000001 000766 MOV      #1,DERFL       ;SET DATA ERROR FLAG
1827 007754 012704 016656 MOV      #WMSG21,R4
1828 007760 004737 012752 JSR      PC,TTOUT        ;PRINT CHARACTER NUMBER TAG
1829 007764 013703 000774 MOV      CRCNT,R3
1830 007770 004737 013100 JSR      PC,OCTP         ;PRINT CHARACTER NUMBER
1831 007774 012704 016644 MOV      #WMSG17,R4
1832 010000 004737 012752 JSR      PC,TTOUT        ;PRINT GOOD TAG
1833 010004 111103          MOVB     (R1),R3
1834 010006 004737 013326 JSR      PC,DOUT         ;PRINT GOOD DATA
1835 010012 012704 016651 MOV      #WMSG20,R4
1836 010016 004737 012752 JSR      PC,TTOUT        ;PRINT BAD TAG
1837 010022 111203          MOVB     (R2),R3
1838 010024 004737 013326 JSR      PC,DOUT         ;PRINT BAD DATA
1839 010030 005737 000764 DCHK2: TST      W2FLG        ;SEE IF WRAP 2 NRZ
1840 010034 001020          BNE      DCHK2B         ;IF SO: BR
1841 010036 005201          INC      R1             ;BUMP POINTER
1842 010040 032737 000020 000776 BIT      #20,UDES        ;SEE IF CORE DUMP
1843 010046 001413          BEQ      DCHK2B         ;IF NOT: BR
1844 010050 022737 000012 000750 CMP      #12,WAM        ;SEE IF WAM 2
1845 010056 001006          BNE      DCHK2A         ;IF NOT: BR
1846 010060 005201          INC      R1             ;BUMP POINTER
1847 010062 005711          TST      (R1)          ;SEE IF END OF PATTERN
1848 010064 001004          BNE      DCHK2B         ;IF NOT: BR
1849 010066 162701 000010 SUB      #10,R1         ;RESET POINTER TO START OF PATTERN
1850 010072 000401          BR      DCHK2B         ;CONTINUE CHECK
1851 010074 000240          DCHK2A: NOP
1852 010076 005202          DCHK2B: INC      R2
1853 010100 022737 000030 000750 CMP      #30,WAM        ;BRANCH IF WAM 4
1854 010106 001404          BEQ      1$
1855 010110 022737 000006 000750 CMP      #6,WAM        ;SEE IF WAM 0
1856 010116 001002          BNE      DCHK3         ;IF NOT: BR
1857 010120 062701 000010 1$: ADD      #10,R1        ;BUMP POINTER
1858 010124 005237 000774 DCHK3: INC      CRCNT     ;BUMP CHAR CNTR
1859 010130 032777 000400 170432 BIT      #400,@SWR      ;SEE IF CONT DATA CHK
1860 010136 001006          BNE      DCHKX         ;IF NOT: BR
1861 010140 005200          INC      R0             ;SEE IF DONE
1862 010142 001242          BNE      DCHK0         ;IF NOT: BR
    
```

```
1863 010144 005737 000764          TST      W2FLG
1864 010150 001401          BEQ      DCHKX
1865 010152 000207          RTS      PC
1866 010154 005777 170410      DCHKX:  TST      @SWR          ;SEE IF HALT ON ERROR
1867 010160 100004          BPL      DCHKX1         ;IF NOT: BR
1868 010162 005737 000766          TST      DERFL         ;SEE IF DATA ERROR OCCURED
1869 010166 001401          BEQ      DCHKX1         ;IF NOT: BR
1870 010170 000000          HALT
1871 010172 005037 000774      DCHKX1: CLR      CRCNT         ;CLEAR CHAR CNTR
1872 010176 005037 000620          CLR      HDRFL         ;CLEAR HEADER FLAG
1873 010202 005037 000766          CLR      DERFL         ;CLEAR DATA ERROR FLAG
1874 010206 005037 000770          CLR      PREFL         ;CLEAR PREAMBLE FLAG
1875 010212 000207          RTS      PC             ;RETURN
```



```

1876
1877
1878
1879 010214 000240
1880 010216 022737 000006 000750
1881 010224 001005
1882 010226 012700 177744
1883 010232 012701 017344
1884 010236 000404
1885 010240 012700 177400
1886 010244 012701 017334
1887 010250 012702 017746
1888 010254 111105
1889 010256 005003
1890 010260 012704 000010
1891 010264 032705 000001
1892 010270 001401
1893 010272 005203
1894 010274 005304
1895 010276 001402
1896 010300 006005
1897 010302 000770
1898 010304 000240
1899 010306 111105
1900 010310 042705 177400
1901 010314 005703
1902 010316 001003
1903 010320 012705 100020
1904 010324 000405
1905 010326 032703 000001
1906 010332 001402
1907 010334 052705 100000
1908 010340 042712 077400
1909 010344 020512
1910 010346 001477
1911 010350 032777 020000 170212
1912 010356 001073
1913 010360 005737 000620
1914 010364 001004
1915 010366 013704 000622
1916 010372 004737 012752
1917 010376 005737 000766
1918 010402 001014
1919 010404 012704 016632
1920 010410 004737 012752
1921 010414 012704 017057
1922 010420 004737 012752
1923 010424 013703 001002
1924 010430 004737 013100
1925 010434 000240
1926 010436 012737 000001 000766
1927 010444 012737 000001 000620
1928 010452 012704 016656
1929 010456 004737 012752
1930 010462 013703 000774
1931 010466 004737 013100

;EVEN PARITY DATA CHECK*****
DCHKE: NOP
      CMP #6,WAM ;SEE IF WRAP 0
      BNE 1$ ;IF NOT: BR
      MOV #-34,R0 ;SET NUMBER OF CHARACTERS READ
      MOV #WBUF+10,R1 ;SET POINTER
      BR 2$
1$: MOV #-400,R0 ;SET NUMBER OF CHARACTERS
   MOV #WBUF,R1 ;R1=START OF WRITE BUFFER
2$: MOV #RBUF,R2 ;R2=START OF READ BUFFER
DCKE0: MOVB (R1),R5 ;GET EXPT DATA
      CLR R3
      MOV #10,R4 ;SET NUMBER OF BITS
DCKE1: BIT #1,R5 ;SEE IF ONE BIT
      BEQ DCKE2 ;IF NOT: BR
      INC R3 ;COUNT ONE BITS FOR PARITY CHECK
DCKE2: DEC R4 ;SEE IF DONE
      BEQ DCKE3 ;IF SO: BR
      ROR R5 ;POINT TO NEXT BIT
      BR DCKE1
DCKE3: NOP
      MOVB (R1),R5 ;GET EXPT DATA
      BIC #177400,R5 ;MASK DATA FIELD
      TST R3
      BNE DCKE4 ;IF NO ONE BITS SET: BR
      MOV #100020,R5
      BR DCKE5
DCKE4: BIT #1,R3 ;SEE IF ODD NUMBER OF ONE BITS
      BEQ DCKE5 ;IF NOT: BR
      BIS #100000,R5 ;SET EVEN PARITY BIT=1
DCKE5: BIC #77400,(R2) ;MASK DATA FIELD
      CMP R5,(R2) ;SEE IF DATA + PARITY GOOD
      BEQ DCKE10 ;IF SO: BR
      BIT #20000,@SWR ;SEE IF ERROR PRINT
      BNE DCKE10 ;IF NOT: BR
      TST HDRFL ;SEE IF DONE HEADER
      BNE DCKE6 ;IF SO: BR
      MOV EMADDR,R4
      JSR PC,TTOUT ;PRINT HEADER
DCKE6: TST DERFL ;SEE IF FIRST BAD CHAR
      BNE DCKE7 ;IF NOT: BR
      MOV #WMSG16,R4
      JSR PC,TTOUT ;PRINT BAD DATA TAG
      MOV #WMSG32,R4
      JSR PC,TTOUT ;PRINT PATTERN TAG
      MOV PATRN,R3
      JSR PC,OCPT ;PRINT PATTERN NUMBER
DCKE7: NOP
      MOV #1,DERFL ;SET DATA ERROR FLAG
      MOV #1,HDRFL ;SET HEADER FLAG
      MOV #WMSG21,R4
      JSR PC,TTOUT ;PRINT CHAR NUMBER TAG
      MOV CRCNT,R3
      JSR PC,OCPT ;ORINT CHAR NUMBER

```

```

1932 010472 012704 016644      MOV      #WMSG17,R4
1933 010476 004737 012752      JSR      PC,TTOUT      ;PRINT GOOD DATA TAG
1934 010502 110503      MOV      R5,R3
1935 010504 004737 013326      JSR      PC,DOUT      ;PRINT EXPT DATA
1936 010510 010503      MOV      R5,R3
1937 010512 004737 010622      JSR      PC,DCKEP     ;GO PRINT PARITY BIT
1938 010516 000240      NOP
1939 010520 012704 016651      MOV      #WMSG20,R4
1940 010524 004737 012752      JSR      PC,TTOUT     ;PRINT BAD TAG
1941 010530 111203      MOV      (R2),R3
1942 010532 004737 013326      JSR      PC,DOUT     ;PRINT BAD DATA
1943 010536 011203      MOV      (R2),R3
1944 010540 004737 010622      JSR      PC,DCKEP     ;GO PRINT PARITY BIT
1945 010544 000240      NOP
1946 010546 005201      DCKE10: INC      R1
1947 010550 022737 000006 000750      CMP      #6,WAM      ;SEE IF WRAP 0
1948 010556 001002      BNE      1$          ;IF NOT: BR
1949 010560 062701 000010      ADD      #10,R1      ;BUMP POINTER
1950 010564 005722      1$:      TST      (R2)+       ;BUMP POINTERS
1951 010566 005237 000774      INC      CRCNT      ;BUMP CHAR CNTR
1952 010572 032777 000400 167770      BIT      #400,@SWR   ;SEE IF CONTINUE DATA CHECK
1953 010600 001402      BEQ      DCKE11     ;IF SO: BR
1954 010602 000137 010154      JMP      DCHKX      ;GO TO END OF DATA CHECK
1955 010606 005200      DCKE11: INC      R0      ;SEE IF DONE
1956 010610 001402      BEQ      DCKE12     ;IF SO: BR
1957 010612 000137 010254      JMP      DCKE0      ;ELSE CONTINUE
1958 010616 000137 010154      DCKE12: JMP      DCHKX   ;GO TO END OF DATA CHECK
1959 010622 000240      DCKEP:  NOP
1960 010624 012737 000240 000614      MOV      #240,TOB
1961 010632 004737 013052      JSR      PC,TOG      ;SPACE
1962 010636 012737 000260 000614      MOV      #260,TOB
1963 010644 005703      TST      R3          ;SEE IF PARITY REALLY=0
1964 010646 100002      BPL      DCKEPO     ;IF SO: BR
1965 010650 005237 000614      INC      TOB        ;ELSE SET TO 1
1966 010654 004737 013052      DCKEPO: JSR      PC,TOG ;PRINT PARITY BIT
1967 010660 000207      RTS      PC          ;RETURN
1968
    
```

```

1969
1970
1971
1972 010662 012700 000051 PSCHK: MOV #51,R0 ;SET SIZE OF POSTAMBLE
1973 010666 012701 017212 MOV #POST,R1 ;SET POINTER TO POSTAMBLE
1974 010672 005037 000620 CLR HDRFL ;CLEAR HEADER FLAG
1975 010676 005037 000774 CLR CRCNT ;CLEAR CHAR CNTR
1976 010702 005037 000766 CLR DERFL ;CLEAR DATA ERROR FLAG
1977 010706 000240 NOP
1978 010710 000240 NOP
1979 010712 000137 010734 JMP PDO ;GO CHECK POSTAMPLE
1980
1981 010716 012700 000051 PRCHK: MOV #51,R0 ;SET SIZE OF PREAMBLE
1982 010722 012701 017070 MOV #PRE,R1 ;SET POINTER TO PREAMBLE
1983 010726 012702 017746 MOV #RBUF,R2 ;SET POINTER TO START OF READ BUFFER
1984 010732 022122 CMP (R1)+,(R2)+ ;BUMP ADDRESS POINTERS
1985 010734 121112 PDO: CMPB (R1),(R2) ;CHECK DATA
1986 010736 001004 BNE PD1 ;IF NOT GOOD: BR
1987 010740 126162 000001 000001 CMPB 1(R1),1(R2) ;COMPARE COMPLIMENT BYTE
1988 010746 001477 BEQ PD5 ;IF GOOD: BR
1989 010750 032777 020000 167612 PD1: BIT #20000,@SWR ;SEE IF PRINT INHIBIT
1990 010756 001073 BNE PD5 ;IF SO: BR
1991 010760 005737 000620 TST HDRFL ;SEE IF DONE HEADER
1992 010764 001020 BNE PD4 ;IF SO: BR
1993 010766 013704 000622 MOV EMADDR,R4
1994 010772 004737 012752 JSR PC,TTOUT ;PRINT TEST HEADER
1995 010776 005737 000770 TST PREFL ;SEE IF PREAMBLE CHECK
1996 011002 001403 BEQ PD2 ;IF NOT: BR
1997 011004 012704 017003 MOV #WMSG29,R4 ;SET POSTAMBLE HEADER
1998 011010 000402 BR PD3
1999 011012 012704 016765 PD2: MOV #WMSG28,R4 ;SET PREAMBLE HEADER
2000 011016 004737 012752 PD3: JSR PC,TTOUT ;PRINT HEADER
2001 011022 005237 000620 INC HDRFL
2002 011026 012704 016656 PD4: MOV #WMSG21,R4
2003 011032 004737 012752 JSR PC,TTOUT ;PRINT CHAR NUMBER TAG
2004 011036 013703 000774 MOV CRCNT,R3
2005 011042 004737 013100 JSR PC,OCTP ;PRINT CHAR NUMBER
2006 011046 012704 016644 MOV #WMSG17,R4
2007 011052 004737 012752 JSR PC,TTOUT ;PRINT GOOD TAG
2008 011056 116103 000001 MOV 1(R1),R3
2009 011062 004737 013326 JSR PC,DOUT ;PRINT GOOD CHAR
2010 011066 012737 000240 000614 MOV #240,TOB
2011 011074 004737 013052 JSR PC,TOG
2012 011100 111103 MOV (R1),R3
2013 011102 004737 013326 JSR PC,DOUT ;PRINT COMPLEMENT
2014 011106 012704 016651 MOV #WMSG20,R4
2015 011112 004737 012752 JSR PC,TTOUT ;PRINT BAD TAG
2016 011116 116203 000001 MOV 1(R2),R3
2017 011122 004737 013326 JSR PC,DOUT ;PRINT BAD CHAR
2018 011126 012737 000240 000614 MOV #240,TOB
2019 011134 004737 013052 JSR PC,TOG
2020 011140 111203 MOV (R2),R3
2021 011142 004737 013326 JSR PC,DOUT ;PRINT COMPLEMENT
2022 011146 022122 PD5: CMP (R1)+,(R2)+ ;BUMP ADDRESS POINTERS
2023 011150 005237 000774 INC CRCNT ;BUMP CHAR NUMBER
2024 011154 005300 DEC R0 ;SEE IF DONE
    
```

2025	011156	001266		BNE	PDO	; IF NOT: BR
2026	011160	005737	000770	TST	PREFL	; SEE IF PREAMBLE
2027	011164	001402		BEQ	PD6	; IF SO: BR
2028	011166	000137	010154	JMP	DCHKX	; GO TO EXIT ROUTINE
2029	011172	005237	000770	PD6:	INC	; SET PREAMBLE FLAG
2030	011176	005037	000620	CLR	HDRFL	; CLEAR HEADER FLAG
2031	011202	005037	000774	CLR	CRCNT	; CLEAR CHAR CNTR
2032	011206	005037	000766	CLR	DERFL	; CLEAR DATA ERROR FLAG
2033	011212	000137	011216	JMP	WIDCHK	; GO CHECK WRAP 1 DATA
2034						

```

2035
2036
2037
2038 011216 012700 177400
2039 011222 012701 017334
2040 011226 012702 017746
2041 011232 062702 000124
2042 011236 005737 000764
2043 011242 001401
2044 011244 005302
2045 011246 111105
2046 011250 120512
2047 011252 001007
2048 011254 005737 000764
2049 011260 001001
2050 011262 105105
2051 011264 120562 000001
2052 011270 001510
2053 011272 032777 020000 167270
2054 011300 001104
2055 011302 005737 000620
2056 011306 001020
2057 011310 013704 000622
2058 011314 004737 012752
2059 011320 012704 016632
2060 011324 004737 012752
2061 011330 012704 017057
2062 011334 004737 012752
2063 011340 013703 001002
2064 011344 004737 013100
2065 011350 012737 000001 000620
2066 011356 012704 016656
2067 011362 004737 012752
2068 011366 013703 000774
2069 011372 004737 013100
2070 011376 012704 016644
2071 011402 004737 012752
2072 011406 111105
2073 011410 110503
2074 011412 005737 000764
2075 011416 001001
2076 011420 105103
2077 011422 004737 013326
2078 011426 012737 000240 000614
2079 011434 004737 013052
2080 011440 110503
2081 011442 004737 013326
2082 011446 012704 016651
2083 011452 004737 012752
2084 011456 116203 000001
2085 011462 004737 013326
2086 011466 012737 000240 000614
2087 011474 004737 013052
2088 011500 111203
2089 011502 004737 013326
2090 011506 005237 000766

;WAM 1 PE DATA CHECK*****

W1DCHK: MOV #-400,R0 ;SET NUMBER OF CHAR TO CHECK
MOV #WBUFF,R1 ;SET WRITE DATA POINTER
MOV #RBUFF,R2 ;SET READ DATA POINTER
ADD #124,R2 ;POINT TO START OF DATA
TST W2FLG ;SEE IF WRAP 2
BEQ W1D0 ;IF NOT WAM 2: BR
DEC R2 ;RESET POINTER

W1D0: MOVB (R1),R5
CMPB R5,(R2) ;CHECK DATA
BNE W1D1 ;IF NOT GOOD:BR
TST W2FLG ;SEE IF WRAP 2
BNE W1D0A ;IF SO: BR
COMB R5 ;COMPLIMENT EXPT DATA

W1D0A: CMPB R5,1(R2) ;CHECK COMPLIMENT DATA
BEQ W1D3 ;IF GOOD: BR

W1D1: BIT #20000,@SWR ;SEE IF PRINT INHIBIT
BNE W1D3 ;IF SO: BR
TST HDRFL ;SEE IF DONE HEADER
BNE W1D2 ;IF SO: BR

MOV EMADDR,R4
JSR PC,TTOUT ;PRINT TEST HEADER

MOV #WMSG16,R4
JSR PC,TTOUT ;PRINT BAD DATA TAG

MOV #WMSG32,R4
JSR PC,TTOUT ;PRINT PATRN TAG

MOV PATRN,R3
JSR PC,OCTP ;PRINT PATTERN NUMBER

W1D2: MOV #1,HDRFL ;SET HEADER FLAG
MOV #WMSG21,R4
JSR PC,TTOUT ;PRINT CHAR NUMBER TAG

MOV CRCNT,R3
JSR PC,OCTP ;PRINT CHAR NUMBER

MOV #WMSG17,R4
JSR PC,TTOUT ;PRNT GOOD TAG

MOVB (R1),R5
MOVB R5,R3 ;GET GOOD CHAR
TST W2FLG ;SEE IF WRAP 2
BNE W1D2A ;IF SO: BR
COMB R3 ;ELSE COMPLIMENT CHAR

W1D2A: JSR PC,DOUT ;PRINT CHARACTER
MOV #240,TOB
JSR PC,TOG ;SPACE

MOVB R5,R3
JSR PC,DOUT ;PRINT CHAR

MOV #WMSG20,R4
JSR PC,TTOUT ;PRINT BAD TAG

MOVB 1(R2),R3
JSR PC,DOUT ;PRINT BAD CHAR

MOV #240,TOB
JSR PC,TOG ;SPACE

MOVB (R2),R3
JSR PC,DOUT ;PRINT CHAR
INC DERFL ;SET DATA ERROR FLAG

```

2091	011512	122122		W1D3:	CMPB	(R1)+,(R2)+	:BUMP ADDRESS
2092	011514	105722			TSTB	(R2)+	:BUMP ADDRESS
2093	011516	005237	000774		INC	CRCNT	:BUMP CHAR CNTR
2094	011522	000406			BR	W1D5	
2095	011524	005737	000764	W1D4:	TST	W2FLG	:SEE IF WRAP 2
2096	011530	001401			BEQ	W1D4A	:IF NOT: BR
2097	011532	000207			RTS	PC	:ELSE RETURN
2098	011534	000137	010662	W1D4A:	JMP	PSCHK	:GO CHECK POSTAMBLE
2099	011540	005200		W1D5:	INC	R0	
2100	011542	001770			BEQ	W1D4	
2101	011544	000137	011246		JMP	W1D0	
2102							
2103							
2104							:PREAMBLE/POSTAMBLE GENERATE SUBROUTINE*****
2105	011550	000240		PPGEN:	NOP		
2106	011552	012700	000050		MOV	#50,R0	:SET SIZE OF PREAMBLE
2107	011556	012701	017070		MOV	#PRE,R1	
2108	011562	005721			TST	(R1)+	:SET ADDRESS OF PRE
2109	011564	012721	177400	1\$:	MOV	#177400,(R1)+	:FILL TABLE
2110	011570	005300			DEC	R0	:SEE IF DONE
2111	011572	001374			BNE	1\$:IF NOT: BR
2112	011574	012701	017212		MOV	#POST,R1	:SET ADDRESS OF POST
2113	011600	012700	000050		MOV	#50,R0	:SET SIZE OF POST
2114	011604	012721	000377		MOV	#377,(R1)+	:SET SYNC CHAR
2115	011610	012721	177400	2\$:	MOV	#177400,(R1)+	:FILL TABLE
2116	011614	005300			DEC	R0	:SEE IF DONE
2117	011616	001374			BNE	2\$:IF NOT: BR
2118	011620	000207			RTS	PC	:RETURN

```
2119
2120 ;END OF RECORD FORCE SUBROUTINE*****
2121
2122 011622 005237 000676 EORPA: INC TEMP2 ;SET WRAP FLAG
2123 011626 017700 166702 EORP: MOV @MR,R0 ;GET MAINT REG
2124 011632 042700 000036 BIC #36,R0 ;CLEAR CURRENT OP CODE
2125 011636 052700 000024 BIS #24,R0 ;SET EOR CLEAR OP CODE
2126 011642 010077 166666 MOV RO,@MR ;DO EOR
2127 011646 042777 000037 166660 BIC #37,@MR ;CLEAR EOR AND MM
2128 011654 005000 CLR RO
2129 011656 012701 000002 MOV #2,R1
2130 011662 032777 000001 166620 EORP1: BIT #1,@C1 ;SEE IF GO GONE
2131 011670 001427 BEQ EORP2 ;IF SO: BR
2132 011672 005300 DEC RO
2133 011674 001372 BNE EORP1 ;AWAIT GO RESET
2134 011676 005301 DEC R1
2135 011700 001370 BNE EORP1
2136 011702 032777 020000 166660 BIT #20000,@SWR ;SEE IF ERROR PRINT INHIBIT
2137 011710 001017 BNE EORP2 ;IF SO: BR
2138 011712 005737 000620 TST HDRFL ;SEE IF DONE HEADER
2139 011716 001004 BNE EORP1A ;IF SO: BR
2140 011720 013704 000622 MOV EMADDR,R4
2141 011724 004737 012752 JSR PC,TTOUT ;PRINT HEADER
2142 011730 012704 017022 EORP1A: MOV #WMSG31,R4
2143 011734 004737 012752 JSR PC,TTOUT ;PRINT EOR GO BIT ERROR
2144 011740 005777 166624 TST @SWR ;SEE IF HALT ON ERROR
2145 011744 100001 BPL EORP2 ;IF NOT: BR
2146 011746 000000 HALT
2147 011750 000240 EORP2: NOP
2148 011752 005737 000676 TST TEMP2 ;SEE IF WAM
2149 011756 001015 BNE EORPX ;IF NOT: BR
2150 011760 032777 000200 166602 BIT #200,@SWR ;SEE IF STATUS CHECK
2151 011766 001002 BNE EORP3 ;IF NOT: BR
2152 011770 004737 006742 JSR PC,WSTCK ;ELSE GO CHECK STATUS
2153 011774 000240 EORP3: NOP
2154 011776 032777 000400 166564 BIT #400,@SWR ;SEE IF DATA CHECK
2155 012004 001002 BNE EORPX ;IF NOT: BR
2156 012006 004737 007332 JSR PC,DCHK ;ELSE GO CHECK DATA
2157 012012 000240 EORPX: NOP
2158 012014 005037 000676 CLR TEMP2 ;CLEAR FLAG
2159 012020 000207 RTS PC ;RETURN
2160
```

```

2161
2162           ;SCOPE LOOP ON ERROR SUBROUTINE*****
2163
2164 012022 000240          SCOPE:  NOP
2165 012024 032777 040000 166536  BIT    #40000,@SWR    ;SEE IF LOOP ON ERROR
2166 012032 001001          BNE    1$           ;IF SO: BR
2167 012034 000207          RTS    PC             ;ELSE EXIT
2168 012036 000240          1$:   NOP
2169 012040 005726          TST    (SP)+         ;RESET STACK
2170 012042 000240          NOP
2171 012044 000240          NOP
2172 012046 000177 166640  JMP    @SCOLP        ;LOOP ON ERROR
2173
2174           ;TEST ITERATION SUBROUTINE*****
2175
2176 012052 032777 004000 166510  ITER:  BIT    #4000,@SWR    ;SEE IF ITERATIONS
2177 012060 001403          BEQ    2$           ;IF SO: BR
2178 012062 005037 000702          1$:   CLR    ITCNT        ;CLEAR ITERATION COUNTER
2179 012066 000207          RTS    PC             ;ELSE EXIT
2180 012070 005737 001016          2$:   TST    @#PCNTR    ;DO SINGLE SUBTEST ITERATION
2181 012074 001772          BEQ    1$           ;ON FIRST PASS
2182 012076 005237 000702          INC    ITCNT        ;BUMP COUNTER
2183 012102 023737 000702 000606  CMP    ITCNT,ITAMT  ;SEE IF DONE ALL
2184 012110 001764          BEQ    1$           ;IF SO: BR
2185 012112 005726          TST    (SP)+         ;RESET STACK
2186 012114 017700 166574  MOV    @ITRLP,RO    ;SET ITERATION POINTER
2187 012120 000110          JMP    (RO)         ;GO ITERATE
2188
  
```



```

2189
2190           ;INITIALIZE SUBROUTINE*****
2191
2192 012122 012777 000040 166370 INIT:  MOV    #40,@CS           ;INIT
2193 012130 013777 000624 166362      MOV    DRVN,@CS         ;SELECT DRIVE
2194 012136 013777 000664 166376      MOV    SLVN,@TC        ;SELECT SLAVE
2195 012144 013746 000776             MOV    UDES,-(SP)      ;GET TEST'S UNIT DESCRIPTION
2196 012150 042716 174377             BIC    #174377,(SP)   ;CLEAR ALL BUT DENSITY SELECT BITS
2197 012154 022726 001400             CMP    #1400,(SP)+    ;BRANCH IF NOT NRZ (800 BPI)
2198 012160 001005                     BNE    1$
2199 012162 032777 000040 166332      BIT    #40,@DS        ;BRANCH IF SLAVE IS IN NRZ MODE
2200 012170 001420                     BEQ    4$              ;(PES = 0)
2201 012172 000404                     BR     2$              ;GO CHANGE DENSITY
2202 012174 032777 000040 166320 1$:  BIT    #40,@DS        ;BRANCH IF SLAVE IS IN PE MODE
2203 012202 001013                     BNE    4$              ;(PES = 1)
2204 012204 012777 000007 166276 2$:  MOV    #7,@C1         ;REWIND SLAVE
2205 012212 032777 000200 166302 20$: BIT    #200,@DS       ;WAIT FOR READY
2206 012220 001774                     BEQ    20$
2207 012222 032777 020000 166272 3$:  BIT    #20000,@DS    ;LOOP UNTIL REWIND IS COMPLETE
2208 012230 001374                     BNE    3$              ;(PIP = 0)
2209 012232 053777 000776 166302 4$:  BIS    UDES,@TC       ;LOAD UNIT DESCRIPTION
2210 012240 032777 000002 166254      BIT    #2,@DS         ;BRANCH IF NOT AT BOT
2211 012246 001407                     BEQ    6$
2212 012250 012777 000025 166232      MOV    #25,@C1        ;ERASE TO GET OFF BOT
2213 012256 032777 000200 166236 5$:  BIT    #200,@DS       ;LOOP UNTIL DONE
2214 012264 001774                     BEQ    5$
2215 012266 012777 000011 166214 6$:  MOV    #11,@C1        ;DO A DRIVE CLEAR
2216 012274 000207                     RTS     PC             ;RETURN TO CALLER
2217
2218           ;MAG TAPE INTERRUPT HANDLER*****
2219
2220 012276 000240             MTINT: NOP
2221 012300 013716 000670      MOV    RTRN,(SP)     ;SET RETURN TO (RTRN)
2222 012304 000002             RTI     ;RETURN
2223
2224           ;TTY INTERRUPT HANDLER*****
2225
2226 012306 017746 166262      TTINT: MOV    @TKB,-(SP)    ;GET CHARACTER
2227 012312 042716 000200      BIC    #200,(SP)     ;CLEAR PARITY BIT
2228 012316 122716 000003      CMPB   #3,(SP) ;BRANCH IF NOT CONTROL C
2229 012322 001006             BNE    1$
2230 012324 005737 001262      TST    CHNFLG        ;INHIBIT C IF CHAIN MODE
2231 012330 001003             BNE    1$
2232 012332 000005             RESET
2233 012334 000137 000200      JMP    @#200          ;RESTART PROGRAM
2234 012340 122716 000001 1$:  CMPB   #1,(SP)       ;BRANCH IF NOT A
2235 012344 001017             BNE    2$
2236 012346 022737 000176 000570    CMP    #SWREG,SWR    ;BRANCH IF HARDWARE SWR INVOKED
2237 012354 001016             BNE    3$
2238 012356 012737 177570 000570    MOV    #177570,SWR   ;INVOKE HARDWARE SWR
2239 012364 004737 013630      JSR    PC,,SAVE      ;SAVE REGISTERS ON THE STACK
2240 012370 012704 014537      MOV    #MSG63,R4     ;TYPE 'HARDWARE SWR IN USE'
2241 012374 004737 012752      JSR    PC,TTOUT
2242 012400 004737 013652      JSR    PC,,RESTORE   ;RESTORE REGISTERS
2243 012404 022716 000007 2$:  CMP    #7,(SP)       ;BRANCH IF NOT G
2244 012410 001005             BNE    4$

```

```
2245 012412 012737 000176 000570 3$: MOV #SWREG,SWR ; INVOKE SOFTWARE SWR
2246 012420 004737 013532 JSR PC,GTSWR ; GET SOFTWARE SWITCHES
2247 012424 005726 4$: TST (SP)+ ; POP CHARACTER OFF THE STACK
2248 012426 000002 RTI ; RETURN
2249
```

```

2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267 012430 010146
2268 012432 011601
2269 012434 005037 000674
2270 012440 005000
2271 012442 004737 012710
2272 012446 122737 000003 000616
2273 012454 001003
2274 012456 000005
2275 012460 000137 000200
2276 012464 122737 000015 000616
2277 012472 001004
2278 012474 005737 000674
2279 012500 001471
2280 012502 000457
2281 012504 122737 000025 000616
2282 012512 001005
2283 012514 012704 014465
2284 012520 004737 012752
2285 012524 000742
2286 012526 122737 000177 000616
2287 012534 001012
2288 012536 000241
2289 012540 006000
2290 012542 006200
2291 012544 006200
2292 012546 012704 014467
2293 012552 004737 012752
2294 012556 005201
2295 012560 000730
2296 012562 122737 000060 000616
2297 012570 101402
2298 012572 000137 012670
2299 012576 122737 000070 000616
2300 012604 101002
2301 012606 000137 012670
2302 012612 005237 000674
2303 012616 006300
2304 012620 006300
2305 012622 006300

```

```

:*****
:TTY ENTRY SUBROUTINE:
:
:THIS SUBROUTINE IS USED BY THE TEST CONDITION
:ENTRY ROUTINE TO READ THE RESPONSE ENTERED
:AT THE TTY AND CHECK THEM FOR LEGALITY AND
:LIMITS. ALL RESPONSE MUST BE TYPED IN OCTAL
:(0-7) AND MUST FALL WITHIN THE LIMITS SET BY
:THE CALLING ROUTINE.
:IF AN ENTRY IS ILLEGAL OR OUTSIDE THE LIMITS,
:A QUESTION MARK IS TYPED (?) AND THE RESPONSE
:MAY BE REENTERED.
:ENTRIES MAY NOT EXCEED SIX (6) CHARACTERS AND
:MAY BE TERMINATED AT LESS THAN SIX BY TYPING A
:CARRIAGE RETURN
:*****
TTR: MOV R1, -(SP) ;SAVE CHAR COUNT
10$: MOV (SP), R1 ;RESET CHAR COUNT (FOR U)
CLR TEMP1 ;CLEAR FIRST CHARACTER FLAG
CLR R0
1$: JSR PC, TTIN ;GO READ CHARACTER
CMPB #3, TIB ;BRANCH IF NOT C
BNE 11$
RESET
JMP @#200 ;RESTART AT 200
11$: CMPB #15, TIB ;SEE IF CR
BNE 2$ ;IF NOT: BR
TST TEMP1 ;SEE IF FIRST CHARACTER
BEQ 9$ ;IF SO: BR
BR 6$ ;ELSE GO LOAD VALUE
2$: CMPB #25, TIB ;BRANCH IF NOT CONTROL U
BNE 21$
MOV #MSG59, R4 ;TYPE<CR><LF>
JSR PC, TTOUT
BR 10$
21$: CMPB #177, TIB ;BRANCH IF NOT 'RUBOUT'
BNE 3$
CLC ;REMOVE LAST TYPED CHAR
ROR R0
ASR R0
ASR R0
MOV #MSG60, R4 ;TYPE ' '
JSR PC, TTOUT
INC R1 ;DECREMENT CHAR RECEIVED COUNT
BR 1$ ;GET NEXT CHAR
3$: CMPB #60, TIB ;SEE IF CHAR IS LESS THAN 0
BLOS 4$ ;IF NOT: BR
JMP T1NER ;ELSE GO TO ERROR
4$: CMPB #70, TIB ;SEE IF CHAR IS GREATER THAN 7
BHI 5$ ;IF NOT: BR
JMP T1NER ;ELSE GO TO ERROR
5$: INC TEMP1 ;SET FIRST CHARACTER FLAG
ASL R0
ASL R0 ;SHIFT 3 LEFT
ASL R0

```

2306	012624	042737	177770	000616		BIC	#177770,T1B	;STRIP ASCII
2307	012632	053700	000616			BIS	T1B,R0	;LOAD CHARACTER
2308	012636	005301				DEC	R1	;SEE IF DONE
2309	012640	001300				BNE	1\$;IF NOT: BR
2310	012642	020002			6\$:	CMP	R0,R2	;SEE IF EXCEEDED MAXIMUM LIMIT
2311	012644	101402				BLOS	7\$;IF NOT: BR
2312	012646	000137	012670			JMP	T1NER	;ELSE GO TO ERROR
2313	012652	020300			7\$:	CMP	R3,R0	;SEE IF BELOW MINIMUM LIMIT
2314	012654	101402				BLOS	8\$;IF NOT: BR
2315	012656	000137	012670			JMP	T1NER	;ELSE GO TO ERROR
2316	012662	010015			8\$:	MOV	R0,(R5)	;LOAD VALUE
2317	012664	005726			9\$:	TST	(SP)+	;POP CHAR COUNT OFF STACK
2318	012666	000207				RTS	PC	;EXIT

```

2319
2320 ;TTY ENTRY ERROR SUBROUTINE*****
2321
2322 012670 012704 014342 T1NER: MOV #MSG40,R4
2323 012674 004737 012752 JSR PC,TTOUT ;PRINT?
2324 012700 005726 TST (SP)+ ;POP CHAR COUNT OFF STACK
2325 012702 162716 000020 SUB #20,(SP) ;RESET SP TO START OF VALUE ROUTINE
2326 012706 000207 RTS PC ;REDO VALUE ENTRY
2327
2328 ;TTY READ SUBROUTINE*****
2329
2330 012710 005277 165656 TTIN: INC @TKS
2331 012714 105777 165652 1$: TSTB @TKS
2332 012720 100375 BPL 1$
2333 012722 017737 165646 000616 MOV @TKB,TIB
2334 012730 042737 000200 000616 BIC #200,TIB
2335 012736 013737 000616 000614 MOV TIB,TOB ;MOVE CHAR TO TTY OUPUT BFR
2336 012744 004737 013052 JSR PC,TOG ;ECHO CHARACTER
2337 012750 000207 RTS PC
2338
2339 ;TTY OU PUT SUBROUTINE*****
2340
2341 012752 112437 000614 TTOUT: MOVB (R4)+,TOB
2342 012756 122737 000043 000614 CMPB #43,TOB
2343 012764 001440 BEQ TEX
2344 012766 122737 000045 000614 CMPB #45,TOB
2345 012774 001403 BEQ 1$
2346 012776 004737 013052 JSR PC,TOG
2347 013002 000763 BR TTOUT
2348 013004 112737 000015 000614 1$: MOVB #15,TOB
2349 013012 004737 013052 JSR PC,TOG
2350 013016 012703 000004 MOV #4,R3
2351 013022 005037 000614 2$: CLR TOB
2352 013026 004737 013052 JSR PC,TOG
2353 013032 005303 DEC R3
2354 013034 001372 BNE 2$ ;DO FILLERS
2355 013036 112737 000012 000614 MOVB #12,TOB
2356 013044 004737 013052 JSR PC,TOG
2357 013050 000740 BR TTOUT
2358 013052 105777 165520 TOG: TSTB @TPS
2359 013056 100375 BPL TOG
2360 013060 113777 000614 165512 MOVB TOB,@TPB
2361 013066 000207 TEX: RTS PC
2362
2363
2364 ;OCTAL OUTPUT SUBROUTINE*****
2365
2366 013070 012737 000001 013324 OCTPE: MOV #1,OFL
2367 013076 000402 BR OCTPE1
2368 013100 005037 013324 OCTP: CLR OFL ;CLEAR FLAG FOR LEADING ZERO
2369 013104 010304 OCTPE1: MOV R3,R4 ;SEE IF NUMBER IS ZERO
2370 013106 001007 BNE OCTP0 ;IF NOT ZERO: BR
2371 013110 005737 013324 TST OFL ;SEE IF PRINT ALL 0
2372 013114 001004 BNE OCTP0 ;IF SO: BR
2373 013116 004737 013304 JSR PC,OCTPG1 ;ELSE PRINT ZERO
2374 013122 000137 013246 JMP OCTP3 ;SPACE AND EXIT

```

```

2375 013126 032704 100000      OCTP0: BIT      #100000,R4      ;SEE IF MSD = 1
2376 013132 001406              BEQ      OCTP1              ;IF NOT: BR
2377 013134 012704 000001      MOV      #1,R4
2378 013140 004737 013262      JSR      PC,OCTPG          ;PRINT 1
2379 013144 000137 013156      JMP      OCTP2
2380 013150 005004              OCTP1: CLR      R4
2381 013152 004737 013262      JSR      PC,OCTPG          ;PRINT 0
2382 013156 010304              OCTP2: MOV      R3,R4
2383 013160 006004              ROR      R4
2384 013162 006004              ROR      R4
2385 013164 006004              ROR      R4              ;POSITION DIGIT
2386 013166 006004              ROR      R4
2387 013170 000304              SWAB     R4
2388 013172 004737 013262      JSR      PC,OCTPG          ;PRINT DIGIT 2
2389 013176 010304              MOV      R3,R4
2390 013200 006004              ROR      R4
2391 013202 000304              SWAB     R4
2392 013204 004737 013262      JSR      PC,OCTPG          ;PRINT DIGIT 3
2393 013210 010304              MOV      R3,R4
2394 013212 006104              ROL      R4
2395 013214 006104              ROL      R4
2396 013216 000304              SWAB     R4
2397 013220 004737 013262      JSR      PC,OCTPG          ;PRINT DIGIT 4
2398 013224 010304              MOV      R3,R4
2399 013226 006004              ROR      R4
2400 013230 006004              ROR      R4
2401 013232 006004              ROR      R4
2402 013234 004737 013262      JSR      PC,OCTPG
2403 013240 010304              MOV      R3,R4
2404 013242 004737 013262      JSR      PC,OCTPG          ;PRINT DIGIT 5
2405 013246 012737 000240 000614 OCTP3: MOV      #240,TOB
2406 013254 004737 013052      JSR      PC,TOG            ;PRINT SPACE
2407 013260 000207              RTS      PC                ;EXIT
2408 013262 042704 177770      OCTPG: BIC      #177770,R4
2409 013266 001004              BNE     OCTPG0
2410 013270 005737 013324      TST     OFL
2411 013274 001001              BNE     OCTPG0
2412 013276 000207              RTS      PC
2413
2414 013300 005237 013324      OCTPG0: INC     OFL
2415 013304 052704 000260      OCTPG1: BIS     #260,R4
2416 013310 010437 000614      MOV      R4,TOB
2417 013314 004737 013052      JSR      PC,TOG
2418 013320 010304              MOV      R3,R4
2419 013322 000207              RTS      PC
2420 013324 000000      OFL:    0                  ;FIRST CHAR FLAG
2421

```

```

2422
2423                ;DATA CHARACTER OUTPUT SUBROUTINE*****
2424
2425 013326 012704 000010      DOUT:  MOV    #10,R4          ;SET NUMBER TO PRINT
2426 013332 110337 000614      MOVB   R3,TOB
2427 013336 105777 165234      1$:   TSTB   @TPS
2428 013342 100375              BPL    1$
2429 013344 132737 000200 000614 BITB   #200,TOB
2430 013352 001404              BEQ    2$
2431 013354 012777 000061 165216 MOV    #061,@TPB
2432 013362 000403              BR     3$
2433 013364 012777 000060 165206 2$:   MOV    #060,@TPB
2434 013372 006337 000614      3$:   ASL    TOB
2435 013376 005304              DEC    R4
2436 013400 001356              BNE   1$
2437 013402 000207              RTS    PC
2438
2439 013404 013703 000700      DOUTD: MOV   TEMP3,R3
2440 013410 000303              SWAB  R3
2441 013412 004737 013326      JSR   PC,DOUT
2442 013416 013703 000700      MOV   TEMP3,R3
2443 013422 004737 013326      JSR   PC,DOUT
2444 013426 000207              RTS    PC
2445
2446                ;TU45 SERIAL NUMBER PRINT SUBROUTINE*****
2447
2448 013430 010304      SNPT:  MOV    R3,R4
2449 013432 000304      SWAB  R4
2450 013434 006004      ROR   R4
2451 013436 006004      ROR   R4
2452 013440 006004      ROR   R4
2453 013442 006004      ROR   R4
2454 013444 004737 013506      JSR   PC,SNPG          ;GET FIRST DIGIT
2455 013450 010304      MOV    R3,R4          ;PRINT
2456 013452 000304      SWAB  R4
2457 013454 004737 013506      JSR   PC,SNPG          ;GET SECOND DIGIT
2458 013460 010304      MOV    R3,R4          ;PRINT
2459 013462 006004      ROR   R4
2460 013464 006004      ROR   R4
2461 013466 006004      ROR   R4
2462 013470 006004      ROR   R4
2463 013472 004737 013506      JSR   PC,SNPG          ;PRINT THIRD DIGIT
2464 013476 010304      MOV    R3,R4
2465 013500 004737 013506      JSR   PC,SNPG          ;PRINT FOURTH DIGIT
2466 013504 000207      RTS    PC             ;EXIT
2467 013506 012737 000260 000614 SNPG:  MOV    #260,TOB        ;SET BASE = 0
2468 013514 042704 177760      BIC   #177760,R4      ;MASK DIGIT
2469 013520 050437 000614      BIS   R4,TOB          ;SET ASCII
2470 013524 004737 013052      JSR   PC,TOG          ;TYPE DIGIT
2471 013530 000207      RTS    PC             ;RETURN

```

```

2472
2473           ;THIS ROUTINE GETS THE NEW VALUE FOR THE SOFTWARE SWITCH REG
2474
2475 013532 022737 000176 000570 GTSWR:  CMP    #SWREG,SWR    ;BRANCH IF SOFTWARE SWR NOT
2476 013540 001032                BNE    1$          ;INVOKED
2477 013542 004737 013630                JSR    PC,,SAVE   ;SAVE REGISTERS ON THE STACK
2478 013546 012704 016015                MOV    #SMSWR,R4  ;TYPE 'SWR = '
2479 013552 004737 012752                JSR    PC,TTOUT
2480 013556 017703 165006                MOV    @SWR,R3    ;GET CURRENT SETTING
2481 013562 004737 013070                JSR    PC,OCTPE   ;AND TYPE THEM
2482 013566 012704 016025                MOV    #SMNEW,R4  ;TYPE ' NEW = '
2483 013572 004737 012752                JSR    PC,TTOUT
2484 013576 013705 000570                MOV    SWR,R5    ;TTR ROUTINE RETURN NEW VALUE TO (R5)
2485 013602 012701 000007                MOV    #7,R1     ;LIMIT RESPONSE TO 7 CHARS
2486 013606 012702 177777                MOV    #177777,R2 ;BETWEEN 0 AND 177777
2487 013612 012703 000000                MOV    #0,R3
2488 013616 004737 012430                JSR    PC,TTR     ;GET RESPONSE
2489 013622 004737 013652                JSR    PC,,RESTORE ;RESTORE REGISTERS
2490 013626 000207                1$:   RTS    PC    ;RETURN TO CALLER
2491
2492           ;;ROUTINE TO SAVE REGISTERS ON THE STACK
2493 013630 010546                .SAVE: MOV    %5,-(SP) ;:R5 IS SAVED AT 12(SP)
2494 013632 010446                MOV    %4,-(SP) ;:R4 IS SAVED AT 10(SP)
2495 013634 010346                MOV    %3,-(SP) ;:R3 IS SAVED AT 6(SP)
2496 013636 010246                MOV    %2,-(SP) ;:R2 IS SAVED AT 4(SP)
2497 013640 010146                MOV    %1,-(SP) ;:R1 IS SAVED AT 2(SP)
2498 013642 010046                MOV    %0,-(SP) ;:R0 IS SAVED AT (SP)
2499 013644 016646 000014                MOV    14(SP),-(SP) ;:PUSH RETURN PC ON THE STACK
2500 013650 000207                RTS    PC        ;:RETURN TO CALLER
2501
2502           ;;ROUTINE TO RESTORE REGISTERS SAVED ON THE STACK
2503 013652 012666 000014                .RESTORE:MOV   (SP)+,14(SP) ;:STORE RETURN PC ON STACK
2504 013656 012600                MOV    (SP)+,%0
2505 013660 012601                MOV    (SP)+,%1
2506 013662 012602                MOV    (SP)+,%2
2507 013664 012603                MOV    (SP)+,%3
2508 013666 012604                MOV    (SP)+,%4
2509 013670 012605                MOV    (SP)+,%5
2510 013672 000207                RTS    PC
2511           ;;RETURN

```



```

2512
2513                                     ;MESSAGE TABLE*****
2514
2515 013674 022445 046524 031460 MSG1: .ASCII'%TM03-TU45 CONTROL LOGIC TEST PART II (CZTUPA0)';++B
2516 013702 052055 032125 020065
2517 013710 047503 052116 047522
2518 013716 020114 047514 044507
2519 013724 020103 042524 052123
2520 013732 050040 051101 020124
2521 013740 044511 020040 041450
2522 013746 052132 050125 030101
2523 013754      051
2524 013755      045 025052 040452 .ASCII /%***ASSURE TAPE IS AT BOT***/
2525 013762 051523 051125 020105
2526 013770 040524 042520 044440
2527 013776 020123 052101 041040
2528 014004 052117 025052      052
2529 014011      045 054524 042520 .ASCII /%TYPE <CR> TO TERMINATE RESPONSE & C TO RESTART%/
2530 014016 036040 051103 020076
2531 014024 047524 052040 051105
2532 014032 044515 040516 042524
2533 014040 051040 051505 047520
2534 014046 051516 020105 020046
2535 014054 041536 052040 020117
2536 014062 042522 052123 051101
2537 014070 022524      043
2538 014073      105 050130 026524 MSG6: .ASCII /EXPT-NOT RECVD#/
2539 014100 047516 020124 042522
2540 014106 053103 021504
2541 014112 041522 042126 047055 MSG7: .ASCII /RCVD-NOT EXPT#/
2542 014120 052117 042440 050130
2543 014126 021524
2544 014130 047045 047117 042455 MSG9: .ASCII /%NON-EXIST SLAVE #/
2545 014136 044530 052123 051440
2546 014144 040514 042526 021440
2547 014152 051045 040505 020104 MSG10: .ASCII /%READ CONT BUS PAR #/
2548 014160 047503 052116 041040
2549 014166 051525 050040 051101
2550 014174 021440
2551 014176 053445 044522 042524 MSG11: .ASCII /%WRITE CONT BUS PAR #/
2552 014204 041440 047117 020124
2553 014212 052502 020123 040520
2554 014220 020122      043
2555 014223      040 054105 052120 MSG12: .ASCII / EXPT #/
2556 014230 021440
2557 014232 051040 053103 020104 MSG13: .ASCII / RCVD #/
2558 014240      043
2559 014241      045 051115 041040 MSG14: .ASCII /%MR BITS 4-0#/
2560 014246 052111 020123 026464
2561 014254 021460
2562 014256 046445 020122 044502 MSG15: .ASCII /%MR BITS 15-7#/
2563 014264 051524 030440 026465
2564 014272 021467
2565 014274 044445 042524 035122 MSG16: .ASCII /%ITER: #/
2566 014302 021440
2567 014304 052045 020103 044502 MSG18: .ASCII /%TC BITS 12-0 #/

```

2568	014312	051524	030440	026462	
2569	014320	020060	043		
2570	014323	045	041506	041040	MSG19: .ASCII /%FC BITS 15-0 #/
2571	014330	052111	020123	032461	
2572	014336	030055	021440		
2573	014342	037440	021440		MSG40: .ASCII / ? #/
2574	014346	022445	047105	020104	MSG41: .ASCII /%END OF PASS #/
2575	014354	043117	050040	051501	
2576	014362	020123	043		
2577	014365	045	042522	044507	MSG44: .ASCII /%REGISTER START: #/
2578	014372	052123	051105	051440	
2579	014400	040524	052122	020072	
2580	014406	043			
2581	014407	045	042526	052103	MSG45: .ASCII /%VECTOR ADDRESS: #/
2582	014414	051117	040440	042104	
2583	014422	042522	051523	020072	
2584	014430	043			
2585	014431	045	046524	031460	MSG57: .ASCII /%TM03 DRIVE: #/
2586	014436	042040	044522	042526	
2587	014444	020072	043		
2588	014447	045	052524	032464	MSG58: .ASCII /%TU45 SLAVE: #/
2589	014454	051440	040514	042526	
2590	014462	020072	043		
2591	014465	045	043		MSG59: .ASCII /%#/
2592	014467	134	043		MSG60: .ASCII / #/
2593	014471	045	042522	047515	MSG62: .ASCII /%REMOVE TMDP FROM SLAVE TO BE TESTED%#/
2594	014476	042526	052040	042115	
2595	014504	020120	051106	046517	
2596	014512	051440	040514	042526	
2597	014520	052040	020117	042502	
2598	014526	052040	051505	042524	
2599	014534	022504	043		
2600	014537	045	040510	042122	MSG63: .ASCII /%HARDWARE SWR IN USE%#/
2601	014544	040527	042522	051440	
2602	014552	051127	044440	020116	
2603	014560	051525	022505	043	

```
2604                                     :TEST HEADER*****
2605
2606 014565      045 046045 043517 MSLT1: .ASCII /%%LOGIC TEST 1: WRAP 3,NRZ,NORMAL,ODD#/
2607 014572      041511 052040 051505
2608 014600      020124 035061 053440
2609 014606      040522 020120 026063
2610 014614      051116 026132 047516
2611 014622      046522 046101 047454
2612 014630      042104      043
2613 014633      045 046045 043517 MSLT2: .ASCII /%%LOGIC TEST 2: WRAP 3,PE,NORMAL,ODD#/
2614 014640      041511 052040 051505
2615 014646      020124 035062 053440
2616 014654      040522 020120 026063
2617 014662      042520 047054 051117
2618 014670      040515 026114 042117
2619 014676      021504
2620 014700      022445 047514 044507 MSLT3: .ASCII /%%LOGIC TEST 3: WRAP 2,NRZ,NORMAL,ODD#/
2621 014706      020103 042524 052123
2622 014714      031440 020072 051127
2623 014722      050101 031040 047054
2624 014730      055122 047054 051117
2625 014736      040515 026114 042117
2626 014744      021504
2627 014746      022445 047514 044507 MSLT4: .ASCII /%%LOGIC TEST 4: WRAP 2,PE,NORMAL,ODD#/
2628 014754      020103 042524 052123
2629 014762      032040 020072 051127
2630 014770      050101 031040 050054
2631 014776      026105 047516 046522
2632 015004      046101 047454 042104
2633 015012      043
2634 015013      045 046045 043517 MSLT5: .ASCII /%%LOGIC TEST 5: WRAP 1,NRZ,NORMAL,ODD#/
2635 015020      041511 052040 051505
2636 015026      020124 035065 053440
2637 015034      040522 020120 026061
2638 015042      051116 026132 047516
2639 015050      046522 046101 047454
2640 015056      042104      043
2641 015061      045 046045 043517 MSLT6: .ASCII /%%LOGIC TEST 6: WRAP 1,PE,NORMAL,ODD#/
2642 015066      041511 052040 051505
2643 015074      020124 035066 053440
2644 015102      040522 020120 026061
2645 015110      042520 047054 051117
2646 015116      040515 026114 042117
2647 015124      021504
2648 015126      022445 047514 044507 MSLT7: .ASCII /%%LOGIC TEST 7: WRAP 0,NRZ,NORMAL,ODD#/
2649 015134      020103 042524 052123
2650 015142      033440 020072 051127
2651 015150      050101 030040 047054
2652 015156      055122 047054 051117
2653 015164      040515 026114 042117
2654 015172      021504
2655 015174      022445 047514 044507 MSLT10: .ASCII /%%LOGIC TEST 10: WRAP 0,PE,NORMAL,ODD#/
2656 015202      020103 042524 052123
2657 015210      030440 035060 053440
2658 015216      040522 020120 026060
2659 015224      042520 047054 051117
```

2660	015232	040515	026114	042117	
2661	015240	021504			
2662	015242	022445	047514	044507	MSLT11: .ASCII /%%LOGIC TEST 11: CORE DUMP WRITE (M8906)#/
2663	015250	020103	042524	052123	
2664	015256	030440	035061	041440	
2665	015264	051117	020105	052504	
2666	015272	050115	053440	044522	
2667	015300	042524	024040	034115	
2668	015306	030071	024466	043	
2669	015313	045	046045	043517	MSLT12: .ASCII /%%LOGIC TEST 12: CORE DUMP READ (M8906)#/
2670	015320	041511	052040	051505	
2671	015326	020124	031061	020072	
2672	015334	047503	042522	042040	
2673	015342	046525	020120	042522	
2674	015350	042101	024040	034115	
2675	015356	030071	024466	043	
2676	015363	045	046045	043517	MSLT13: .ASCII /%%LOGIC TEST 13: EVEN PARITY WRITE (M8933 M8934)#/
2677	015370	041511	052040	051505	
2678	015376	020124	031461	020072	
2679	015404	053105	047105	050040	
2680	015412	051101	052111	020131	
2681	015420	051127	052111	020105	
2682	015426	046450	034470	031463	
2683	015434	046440	034470	032063	
2684	015442	021451			
2685	015444	022445	047514	044507	MSLT14: .ASCII /%%LOGIC TEST 14: EVEN PARITY READ(M8933 M8934)#/
2686	015452	020103	042524	052123	
2687	015460	030440	035064	042440	
2688	015466	042526	020116	040520	
2689	015474	044522	054524	051040	
2690	015502	040505	024104	034115	
2691	015510	031471	020063	034115	
2692	015516	031471	024464	043	
2693	015523	045	046045	043517	MSLT15: .ASCII /%%LOGIC TEST 15: READ REVERSE(M8906)#/
2694	015530	041511	052040	051505	
2695	015536	020124	032461	020072	
2696	015544	042522	042101	051040	
2697	015552	053105	051105	042523	
2698	015560	046450	034470	033060	
2699	015566	021451			
2700	015570	022445	047514	044507	MSLT16: .ASCII /%%LOGIC TEST 16: CRC CORRECTION SINGLE TRACK,ALL FRAMES#/
2701	015576	020103	042524	052123	
2702	015604	030440	035066	041440	
2703	015612	041522	041440	051117	
2704	015620	042522	052103	047511	
2705	015626	020116	044523	043516	
2706	015634	042514	052040	040522	
2707	015642	045503	040454	046114	
2708	015650	043040	040522	042515	
2709	015656	021523			
2710	015660	022445	047514	044507	MSLT17: .ASCII /%%LOGIC TEST 17: CRC CORRECTION MULTIPLE BAD TRACKS#/
2711	015666	020103	042524	052123	
2712	015674	030440	035067	041440	
2713	015702	041522	041440	051117	
2714	015710	042522	052103	047511	
2715	015716	020116	052515	052114	

2716	015724	050111	042514	041040
2717	015732	042101	052040	040522
2718	015740	045503	021523	
2719	015744	022445	047514	044507
2720	015752	020103	042524	052123
2721	015760	031040	035060	051040
2722	015766	040505	020104	042522
2723	015774	042526	051522	026105
2724	016002	051116	026132	051127
2725	016010	050101	031440	043

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

```
2726
2727 ;TAG MESSAGE
2728
2729 016015 045 053523 020122 $MSWR: .ASCII /%SWR = #/
2730 016022 020075 043
2731 016025 040 042516 020127 $MNEW: .ASCII / NEW = #/
2732 016032 020075 043
2733
2734 016036 .EVEN
2735 ;WRITE BUFFER
2736
2737 016036 000100 WDATA:
2738 016036 177777 -1
2739 016040 177777 -1
2740 016042 177777 -1
2741 016044 177777 -1
2742 016046 177777 -1
2743 016050 177777 -1
2744 016052 177777 -1
2745 016054 177777 -1
2746 016056 177777 -1
2747 016060 177777 -1
2748 016062 177777 -1
2749 016064 177777 -1
2750 016066 177777 -1
2751 016070 177777 -1
2752 016072 177777 -1
2753 016074 177777 -1
2754 016076 177777 -1
2755 016100 177777 -1
2756 016102 177777 -1
2757 016104 177777 -1
2758 016106 177777 -1
2759 016110 177777 -1
2760 016112 177777 -1
2761 016114 177777 -1
2762 016116 177777 -1
2763 016120 177777 -1
2764 016122 177777 -1
2765 016124 177777 -1
2766 016126 177777 -1
2767 016130 177777 -1
2768 016132 177777 -1
2769 016134 177777 -1
2770 016136 177777 -1
2771 016140 177777 -1
2772 016142 177777 -1
2773 016144 177777 -1
2774 016146 177777 -1
2775 016150 177777 -1
2776 016152 177777 -1
2777 016154 177777 -1
2778 016156 177777 -1
2779 016160 177777 -1
2780 016162 177777 -1
2781 016164 177777 -1
```

2782	016166	177777	-1
2783	016170	177777	-1
2784	016172	177777	-1
2785	016174	177777	-1
2786	016176	177777	-1
2787	016200	177777	-1
2788	016202	177777	-1
2789	016204	177777	-1
2790	016206	177777	-1
2791	016210	177777	-1
2792	016212	177777	-1
2793	016214	177777	-1
2794	016216	177777	-1
2795	016220	177777	-1
2796	016222	177777	-1
2797	016224	177777	-1
2798	016226	177777	-1
2799	016230	177777	-1
2800	016232	177777	-1
2801	016234	177777	-1

2802
2803
2804 :READ BUFFER
2805

2806	016236	000100	RDATA:	0
2807	016236	000000		0
2808	016240	000000		0
2809	016242	000000		0
2810	016244	000000		0
2811	016246	000000		0
2812	016250	000000		0
2813	016252	000000		0
2814	016254	000000		0
2815	016256	000000		0
2816	016260	000000		0
2817	016262	000000		0
2818	016264	000000		0
2819	016266	000000		0
2820	016270	000000		0
2821	016272	000000		0
2822	016274	000000		0
2823	016276	000000		0
2824	016300	000000		0
2825	016302	000000		0
2826	016304	000000		0
2827	016306	000000		0
2828	016310	000000		0
2829	016312	000000		0
2830	016314	000000		0
2831	016316	000000		0
2832	016320	000000		0
2833	016322	000000		0
2834	016324	000000		0
2835	016326	000000		0
2836	016330	000000		0
2837	016332	000000		0

2838	016334	000000	0
2839	016336	000000	0
2840	016340	000000	0
2841	016342	000000	0
2842	016344	000000	0
2843	016346	000000	0
2844	016350	000000	0
2845	016352	000000	0
2846	016354	000000	0
2847	016356	000000	0
2848	016360	000000	0
2849	016362	000000	0
2850	016364	000000	0
2851	016366	000000	0
2852	016370	000000	0
2853	016372	000000	0
2854	016374	000000	0
2855	016376	000000	0
2856	016400	000000	0
2857	016402	000000	0
2858	016404	000000	0
2859	016406	000000	0
2860	016410	000000	0
2861	016412	000000	0
2862	016414	000000	0
2863	016416	000000	0
2864	016420	000000	0
2865	016422	000000	0
2866	016424	000000	0
2867	016426	000000	0
2868	016430	000000	0
2869	016432	000000	0
2870	016434	000000	0

2871
2872 ;WRAP AROUND MESSAGES*****

2873					
2874	016436	051445	052105	050125	WMSG2: .ASCII /%SETUP ERROR%#/ 042440 051122 051117
2875	016444	021445			
2876	016452	050045	052101	047122	WMSG3: .ASCII /%PATRN NUMBER = #/ 047040 046525 042502
2877	016454	020122	020075	043	
2878	016462	045	047516	026516	WMSG4: .ASCII /%NON-EXISTANT DRIVE%#/ 045 047516 026516
2879	016470	054105	051511	040524	
2880	016475	052116	042040	044522	
2881	016502	042526	021445		
2882	016510	041445	030523	021440	WMSG6: .ASCII /%CS1 #/ 053445 020103 043
2883	016516	040502	021440		WMSG6A: .ASCII /%WC #/ 045 040502 021440
2884	016522	043045	020103	043	WMSG6B: .ASCII /%BA #/ 043045 020103 043
2885	016530	045	051503	020062	WMSG6C: .ASCII /%FC #/ 045 051503 020062
2886	016535	043			WMSG6D: .ASCII /%CS2 #/ 043
2887	016542	051504	021440		WMSG6E: .ASCII /%DS #/ 045 051504 021440
2888	016547	042445	020122	043	WMSG6F: .ASCII /%ER #/ 042445 020122 043
2889	016554	045	051501	021440	WMSG6G: .ASCII /%AS #/ 045 051501 021440
2890	016555	041445	020103	043	WMSG6H: .ASCII /%CC #/ 041445 020103 043
2891	016562				
2892	016567				
2893	016574				

2894	016601	045	041104	021440	WMSG6I: .ASCII	/XDB #/
2895	016606	046445	020122	043	WMSG6J: .ASCII	/XMR #/
2896	016613	045	052104	021440	WMSG6K: .ASCII	/XDT #/
2897	016620	052045	020103	043	WMSG6L: .ASCII	/XTC #/
2898	016625	045	047123	021440	WMSG6M: .ASCII	/XSN #/
2899	016632	041045	042101	042040	WMSG16: .ASCII	/XBAD DATA#/
2900	016640	052101	021501			
2901	016644	043445	020072	043	WMSG17: .ASCII	/XG: #/
2902	016651	045	035102	021440	WMSG20: .ASCII	/XB: #/
2903	016656	041445	035116	021440	WMSG21: .ASCII	/XCN: #/
2904	016664	041045	042101	051440	WMSG23: .ASCII	/XBAD STATUS#/
2905	016672	040524	052524	021523		
2906	016700	047045	020117	047111	WMSG24: .ASCII	/XNO INTERRUPT#/
2907	016706	042524	051122	050125		
2908	016714	021524				
2909	016716	047045	020117	046103	WMSG25: .ASCII	/XNO CLOCK UP#/
2910	016724	041517	020113	050125		
2911	016732	043				
2912	016733	045	047516	041440	WMSG26: .ASCII	/XNO CLOCK DOWN#/
2913	016740	047514	045503	042040		
2914	016746	053517	021516			
2915	016752	042045	052101	020101	WMSG27: .ASCII	/XDATA PAT: #/
2916	016760	040520	035124	043		
2917	016765	045	040502	020104	WMSG28: .ASCII	/XBAD PREAMBLE#/
2918	016772	051120	040505	041115		
2919	017000	042514	043			
2920	017003	045	040502	020104	WMSG29: .ASCII	/XBAD POSTAMBLE#/
2921	017010	047520	052123	046501		
2922	017016	046102	021505			
2923	017022	042445	051117	041440	WMSG31: .ASCII	/XEOB CLEAR DID NOT CLEAR GOX#/
2924	017030	042514	051101	042040		
2925	017036	042111	047040	052117		
2926	017044	041440	042514	051101		
2927	017052	043440	022517	043		
2928	017057	040	040520	051124	WMSG32: .ASCII	/ PATRN #/
2929	017064	020116	043			
2930						
2931		017070				
2932	017070	000000			PRE:	.EVEN
2933	017072	000000				0
2934	017074	000000				0
2935	017076	000000				0
2936	017100	000000				0
2937	017102	000000				0
2938	017104	000000				0
2939	017106	000000				0
2940	017110	000000				0
2941	017112	000000				0
2942	017114	000000				0
2943	017116	000000				0
2944	017120	000000				0
2945	017122	000000				0
2946	017124	000000				0
2947	017126	000000				0
2948	017130	000000				0
2949	017132	000000				0

2950	017134	000000	0
2951	017136	000000	0
2952	017140	000000	0
2953	017142	000000	0
2954	017144	000000	0
2955	017146	000000	0
2956	017150	000000	0
2957	017152	000000	0
2958	017154	000000	0
2959	017156	000000	0
2960	017160	000000	0
2961	017162	000000	0
2962	017164	000000	0
2963	017166	000000	0
2964	017170	000000	0
2965	017172	000000	0
2966	017174	000000	0
2967	017176	000000	0
2968	017200	000000	0
2969	017202	000000	0
2970	017204	000000	0
2971	017206	000000	0
2972	017210	000000	0
2973	017212	000000	0
2974	017214	000000	0
2975	017216	000000	0
2976	017220	000000	0
2977	017222	000000	0
2978	017224	000000	0
2979	017226	000000	0
2980	017230	000000	0
2981	017232	000000	0
2982	017234	000000	0
2983	017236	000000	0
2984	017240	000000	0
2985	017242	000000	0
2986	017244	000000	0
2987	017246	000000	0
2988	017250	000000	0
2989	017252	000000	0
2990	017254	000000	0
2991	017256	000000	0
2992	017260	000000	0
2993	017262	000000	0
2994	017264	000000	0
2995	017266	000000	0
2996	017270	000000	0
2997	017272	000000	0
2998	017274	000000	0
2999	017276	000000	0
3000	017300	000000	0
3001	017302	000000	0
3002	017304	000000	0
3003	017306	000000	0
3004	017310	000000	0
3005	017312	000000	0

POST:

3006	017314	000000	0
3007	017316	000000	0
3008	017320	000000	0
3009	017322	000000	0
3010	017324	000000	0
3011	017326	000000	0
3012	017330	000000	0
3013	017332	000000	0
3014	017334	000000	WBUFF: 0
3015		017746	.+.410
3016	017746	000000	RBUFF: 0
3017			
3018		000001	.END

ADDFL	000746	DCKE2	010274	EX5C	006350	MSG10	014152	PATRN	001002
AS	000526	DCKE3	010304	EX5C0	006414	MSG11	014176	PCNTR	001016
ASF	000730	DCKE4	010326	EX5C1	006420	MSG12	014223	PDO	010734
ATAF	000720	DCKE5	010340	EX5D	006424	MSG13	014232	PD1	010750
BA	000514	DCKE6	010376	FC	000516	MSG14	014241	PD2	011012
CC	000530	DCKE7	010434	FUN	000752	MSG15	014256	PD3	011016
CHNFLG	001262	DERFL	000766	GTSWR	013532	MSG16	014274	PD4	011026
CRCDAT	004304	DOUT	013326	HDRFL	000620	MSG18	014304	PD5	011146
CRCFLG	001022	DOUTD	013404	HERE	002320	MSG19	014323	PD6	011172
CRCNT	000774	DRVN	000624	ILFT	000544	MSG40	014342	PEXFL	000736
CRCPAT	004302	DRVTP	000604	INIT	012122	MSG41	014346	PFLG	000666
CS	000520	DS	000522	ITAMT	000606	MSG44	014365	POST	017212
C1	000510	DSUP	004742	ITCNT	000702	MSG45	014407	PPGEN	011550
DATAD	000760	DSO	005040	ITER	012052	MSG57	014431	PRCHK	010716
DATA0	001034	DS3	005056	ITRLP	000714	MSG58	014447	PRE	017070
DATA1	001036	DS4	005066	LTADD	000742	MSG59	014465	PREFL	000770
DATA2	001040	DT	000536	LT1	002352	MSG6	014073	PSCHK	010662
DATA3	001042	EMADDR	000622	LT1A	002416	MSG60	014467	PSW	000566
DATBL	001034	ENDFLG	001024	LT1B	002430	MSG62	014471	RBUFF	017746
DATC	000754	EORP	011626	LT10	003210	MSG63	014537	RCDP	001010
DAT1	005076	EORPA	011622	LT10A	003212	MSG7	014112	RDAD	000762
DAT1A	005102	EORPX	012012	LT11	003262	MSG9	014130	RDATA	016236
DAT2	005116	EORP1	011662	LT11A	003340	MSLT1	014565	RDRVF	001006
DAT3	005122	EORP1A	011730	LT11X	003364	MSLT10	015174	REGS	000612
DAT4	005130	EORP2	011750	LT12	003374	MSLT11	015242	RTRN	000670
DB	000532	EORP3	011774	LT12A	003460	MSLT12	015313	SAV1	000704
DCHK	007332	ER	000524	LT12X	003512	MSLT13	015363	SAV2	000706
DCHKA	007430	ERADD	000672	LT13	003522	MSLT14	015444	SAV3	000710
DCHKA0	007404	ERRF	000726	LT14	003572	MSLT15	015523	SCF	000732
DCHKA1	007460	EXEC	006010	LT15	003642	MSLT16	015570	SCOLP	000712
DCHKA2	007516	EXFL	000716	LT16	003724	MSLT17	015660	SCOPE	012022
DCHKB	007570	EXW2	006454	LT16A	003752	MSLT2	014633	SERFL	000772
DCHKBO	007600	EXW2A	006504	LT17	004312	MSLT20	015744	SERNUM	000602
DCHKC	007640	EXW2B	006540	LT17A	004342	MSLT3	014700	SETUP	005430
DCHKD	007646	EXW2C	006554	LT2	002464	MSLT4	014746	SET0	005472
DCKHE	010214	EXW2E	006574	LT2A	002466	MSLT5	015013	SET1	005476
DCHKFL	001020	EXW2F	006602	LT20	004624	MSLT6	015061	SET1A	005502
DCHKX	010154	EXW2G	006632	LT20A	004702	MSLT7	015126	SET2	005526
DCHKX1	010172	EXW2H	006642	LT3	002536	MTINT	012276	SKAT	001014
DCHK0	007650	EXW2J	006656	LT3A	002602	NRZOF	000662	SLAF	000722
DCHK1	007702	EXW2JO	006676	LT3B	002614	NXTDRV	001760	SLVN	000664
DCHK1A	007740	EXW2J1	006700	LT4	002644	NXTSLV	002032	SN	000540
DCHK2	010030	EXW2K	006720	LT4A	002646	OCTP	013100	SNPG	013506
DCHK2A	010074	EXW2X	006736	LT5	002716	OCTPE	013070	SNPT	013430
DCHK2B	010076	EX0	006032	LT5A	002762	OCTPE1	013104	SPO	005604
DCHK3	010124	EX1	006056	LT5B	002774	OCTPG	013262	SP01	005572
DCKEP	010622	EX1A	006072	LT6	003024	OCTPG0	013300	SP1	005610
DCKEPO	010654	EX2	006112	LT6A	003026	OCTPG1	013304	SP1A	005662
DCKEO	010254	EX3	006136	LT7	003102	OCTP0	013126	SP1B	005702
DCKE1	010264	EX5	006206	LT7A	003146	OCTP1	013150	SP3	005754
DCKE10	010546	EX5A	006212	LT7B	003160	OCTP2	013156	SP4	005774
DCKE11	010606	EX5A1	006246	MR	000534	OCTP3	013246	SP5	006006
DCKE12	010616	EX5B	006302	MSG1	013674	OFL	013324	SSCF	000724

START	001200	TR00	000626	VECT	000610	WMSG25	016716	WSTX	007330
STATC	001012	TR01	000630	WAM	000750	WMSG26	016733	WTAD	000756
STATF	001004	TR02	000632	WAM0	005136	WMSG27	016752	W1DCHK	011216
STFLG	000740	TR03	000634	WAM01	005144	WMSG28	016765	W1D0	011246
STFLGS	000614	TR04	000636	WAM1	005202	WMSG29	017003	W1D0A	011264
STSCD	002210	TR05	000640	WAM2	005212	WMSG3	016454	W1D1	011272
ST2	001670	TR06	000642	WAM2A	005246	WMSG31	017022	W1D2	011350
SWR	000570	TR07	000644	WAM3	005256	WMSG32	017057	W1D2A	011422
SWREG	000176	TR10	000646	WAM3A	005336	WMSG4	016475	W1D3	011512
TADX	001174	TR11	000650	WAM3B	005354	WMSG6	016522	W1D4	011524
TC	000542	TR12	000652	WAM4	005364	WMSG6A	016530	W1D4A	011534
TEMP1	000674	TR13	000654	WBUFF	017334	WMSG6B	016535	W1D5	011540
TEMP2	000676	TR14	000656	WC	000512	WMSG6C	016542	W2FLG	000764
TEMP3	000700	TR15	000660	WCDP0	001056	WMSG6D	016547	XORDAT	004306
TEND	002244	TSCD	001714	WCDP2	001044	WMSG6E	016555	XORPAT	004310
TENDX	002342	TSCDA	002110	WCS1	001024	WMSG6F	016562	\$DONE	002252
TEX	013066	TSCD0	002116	WCS2	001026	WMSG6G	016567	\$ENDAD	002310
TIB	000616	TSCD1	002124	WDATA	016036	WMSG6H	016574	\$MNEW	016025
TINER	012670	TSCD2	002152	WDS	001030	WMSG6I	016601	\$MSWR	016015
TKB	000574	TSCD3	002170	WER	001032	WMSG6J	016606	\$SVPC =	000764
TKS	000572	TSTTBL	001070	WMSG16	016632	WMSG6K	016613	=	017750
TLAST	001176	TTIN	012710	WMSG17	016644	WMSG6L	016620	.RESTO	013652
TOB	000614	TTINT	012306	WMSG2	016436	WMSG6M	016625	.SAVE	013630
TOG	013052	TTOUT	012752	WMSG20	016651	WPGFL	001000		
TPB	000600	TTR	012430	WMSG21	016656	WSTCK	006742		
TPS	000576	T24FL	000744	WMSG23	016664	WSTG	007206		
TREF	000734	UDES	000776	WMSG24	016700	WSTG0	007256		

. ABS. 017750 000

ERRORS DETECTED: 0

,CZTUPA.SEQ/SOL_CZTUPA.P11
RUN-TIME: 24 42 2 SECONDS
RUN-TIME RATIO: 113/69=1.6
CORE USED: 14K (28 PAGES)