

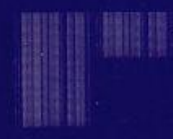
TM03, TU45

TM03/TU45 LGC PRT 2 AH-E491B-MC  
CZTUPBO FICHE 1 OF 1

JUN 1980  
COPYRIGHT © 75.80  
MADE IN USA



Microfiche grid containing multiple frames of data, including tables and text. The content is extremely faint and illegible due to the low resolution of the scan.



.REM %

IDENTIFICATION

PRODUCT CODE: AC-E490B-MC  
PRODUCT NAME: CZTUPB0 TM03/TU45 LGC PRT 2  
DATE CREATED: 25 MAY 1978  
MAINTAINER: CSS - NASHUA  
AUTHOR: J. G. ADAMS/R. J. COLLINS  
UPDATE INFORMATION: DATE 29-FEB-1980 AUTHOR VIJAY ANANDWALA

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 (©) 1975, 1980 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 PARENTESES (), AND  
MEMORY LOCATIONS CONTAINING THE DEFAULT ARE SHOWN IN  
SQUARE BRACKETS [].

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

TM03-TU45 CONTROL LOGIC TEST PART II (CZTUPB0)  
\*\*\*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 FOLLOWING ARE TWO EXAMPLES OF ERRORS DETECTED BY THE WRAP AROUND DATA TESTS. NOTE THAT EACH WRAP AROUND TEST MAY BE ACCOMPANIED BY EITHER A STATUS ERROR OF A DATA ERROR OR BOTH.

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

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

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

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

7. OPERATION

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

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

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

SINGLE TEST SELECTION: (SW0-SW5)

WHEN SW0-SW5 ARE SET TO ZERO (00), THE SCHEDULAR WILL EXECUTE ALL TESTS IN SEQUENCE. IF SW0-SW5 ARE SET TO SOME SPECIFIC TEST NUMBER THEN THAT PARTICULAR TEST ONLY WILL BE EXECUTED UNTIL THE TEST SELECT NUMBER IS CHANGED. WHEN YOU WISH TO SELECT A PARTICULAR TEST, SET SW10 TO A ONE (1) IN ORDER TO STOP AT THE END OF THE CURRENT TEST BEFORE SELECTING A DIFFERENT TEST NUMBER. YOU MAY SELECT THAT NUMBER IN ANY DIRECTION (HIGHER OR LOWER) BECAUSE EACH TEST IS SELF CONTAINED.

WRAP AROUND DATA PATTERNS MAY BE SELECTED VIA SW6 WHEN IN SINGLE TEST MODE. A TELETYPE REQUEST IS MADE FOR THE DESIRED DATA PATTERN WHENEVER SWITCH TEN (SW10) AND SWITCH SIX (SW6) ARE SET TO A ONE (1) WHILE ONE OF THE WRAP TESTS IS SELECTED IN SW0-SW5.

8. SUBTEST SUMMARIES

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

FOR THE MOST PART, THIS DIAGNOSTIC TESTS PARTICULAR AREAS OF THE TMO3 LOGIC INDEPENDENT OF THE 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 CLGCK(SB)L: USED IN WAMO TO GENERATE DATA AND REC(SB)L
4. RSDO(SB)L: SHOULD NOT OCCUR DURING WRAP AROUND TESTS, BUT WILL INTERFERE WITH THEIR OPERATION IF CAUSED BY A FAILURE SUCH AS A GROUNDED OUTPUT FROM THE G056.

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

PROGRAMMED SEQUENCE:

TAPE CONTROL REGISTER IS LOADED WITH DENSITY 3, FORMAT 14, ODD PARITY WRP3 IS LOADED IN MAINT. REGISTER. READ FUNCTION IS LOADED, EXECUTING WRAP3 CONSISTS LOADING DATA CHARACTERS INTO MAINT. REGISTER DATA FIELD, WHERE THERE ARE MULTI- PLEXERS TO BIT FIDDLER, MM CLK IS TOGGLED TO CREATE RDS. THE BIT FIDDLER TRANSMIT DATA ACCESS MASSBUS DATA LINES. WHEN ALL THE DATA HAS BEEN TRANSMITTED AN EOR CLK IS TRANSMITTED TO N REGISTER WHICH BRINGS OPERATION TO A CLOSE.

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

CIRCUITS

PRINT REFERENCES

MASSBUS CHAR. ASSEMBLE  
CLR. 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 APPEAR UNTIL THE PE WRITE CONTROL CIRCUIT HAS CLOCKED THROUGH THE PREAMBLE.

CIRCUITS  
-----

PRINT REFERENCES  
-----

(IN ADDITION TO TEST 44)  
PE WRITE CONTROL

TCPE3

LOGIC TEST 5: WRP1, NRZ, NORMAL, ODD  
-----

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

LIKELY FAULT LOCATIONS: M8933, M8934 (CRC GENERATOR)  
-----

CIRCUITS  
-----

PRINT REFERENCES  
-----

WRITE BUFFER  
CRC GENERATOR

TCCM2  
CNRZ2

LOGIC TEST 6: WRAP1, PE, NORMAL, ODD  
-----

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

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

CIRCUITS  
-----

PRINT REFERENCE  
-----

WRITE BUFFER  
WRITE CONTROL

TCCM2  
TCPE3

1

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	CNRZ2
CRC CHECKOUT	CNRZ3
CRC, CRC STROBE	TCCM4
READ LINE MULTIPLEXERS	TCCM6
MM CLK	MR5
CRC READ TIMING	CNRZ4
SHUTDOWN CIRCUITRY	TCCM5

LOGIC TEST 10: WRP0, PE, NORMAL, ODD  
-----

REPEAT OF TEST 7 IN PE MODE.

LIKELY FAULT LOCATIONS: M8901, M8932, M8933  
-----

CIRCUITS  
-----

PRINT REFERENCES  
-----

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

LOGIC TEST 11: CORE DUMP WRITE, WAM2

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

LIKELY FAULT LOCATION: M8906

LOGIC TEST 12: CORE DUMP READ, WAM3

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

LIKELY FAULT LOCATION: M8906

LOGIC TEST 13: EVEN PARITY WRITE - WAM1

REPEAT OF TEST 5 EXCEPT EVEN PARITY IS SPECIFIED.

LIKELY FAULT LOCATION: M8933

LOGIC TEST 14: EVEN PARITY READ: WAM0,

REPEAT OF TEST 7 EXCEPT EVEN PARITY IS USED.

LIKELY FAULT LOCATIONS: M8933, M8934

LOGIC TEST 15: READ REVERSE, WAM3 (M8906)  
-----

REPEAT OF TEST 2 EXCEPT READ REVERSE COMMAND IS ISSUED.

LIKELY FAULT LOCATIONS: M8908, 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.

1844

x

.LIST BIN,LOC,SEQ

1848  
1849  
1850  
1851  
1852  
1853  
1854  
1855  
1856  
1857  
1858  
1859  
1860  
1861  
1862  
1863  
1864  
1865  
1866  
1867  
1868  
1869  
1870  
1871  
1872  
1873  
1874  
1875  
1876  
1877  
1878  
1879  
1880  
1881  
1882

```
.TITLE TM03-TU45 CONTROL LOGIC TEST-PART II
:CZTUPB0
: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
```

1884  
1885  
1886  
1887  
1888  
1889  
1890  
1891  
1892  
1893  
1894  
1895  
1896  
1897  
1898  
1899  
1900  
1901  
1902  
1903  
1904  
1905  
1906  
1907  
1908  
1909  
1910  
1911  
1912  
1913  
1914  
1915  
1916  
1917  
1918  
1919  
1920  
1921  
1922  
1923  
1924  
1925  
1926  
1927  
1928  
1929

```
:TU45 REGISTER BITS*****  
:15 :14 :13 :12 :11 :10 :09 :08 :07 :06 :05 :04 :03 :02 :01 :00 :  
:CS1: :SC :TRE;MCP;SPR;DNA;PSL;A17;A16;RDY;IE ;FUN;FUN;FUN;FUN;FUN;GO ;  
:WC: :WORD  
:BA: :BUS  
:FC: :FRAME  
:CS2: :DLT;WCE;UPE;NED;NEM;PGE;MXF;MDP;OR ;IR ;CLR;PAT;BA1;U2 ;U1 ;U0 ;  
:DS: :ATA;ERR;PIP;MOL;WRL;EOT;SPR;DPR;DRY;SSC;PES;SDN;IDB;TMK;BOT;SLA;  
:ER: :CDE;UNS;OPI;DTE;NEF;CS ;FCE;NSG;PEF;INC;DAT;FMT;CNT;RMR;ILR;ILF;  
:ATS: :ATTENTION  
:CC: :CHECK  
:DB: :DATA  
:MR: :DB ;DB ;DB ;DB ;DB ;DB ;DB ;DB ;DB ;DB ;WRT;MM ;OP ;OP ;OP ;OP ;MM ;  
:DT: :DRIVE ;7 ; :SLV;  
:SN: :SERIAL  
:TC: :ACL;FCS;SAC;ENA;SPR;DEN;DEN;DEN;FMT;FMT;FMT;FMT;EVN;SSN;SSN;SSN;  
: : :DT : :4 ;2 ;1 ;8 ;4 ;2 ;1 ;PAR;4 ;2 ;1 ;
```

```

1931                                     ;REGISTER EQUIVS*****
1932
1933         000000                       R0=%0
1934         000001                       R1=%1
1935         000002                       R2=%2
1936         000003                       R3=%3
1937         000004                       R4=%4
1938         000005                       R5=%5
1939         000006                       SP=%6
1940         000007                       PC=%7
1941
1943
1944
(1)
(1)         000764                       ;ACT11 HOOK *****
(1)         000046                       $SVPC=.           ;SAVE CURRENT LOCATION CTR
(1) 000046 002374                       .=46
(1)         000052                       .WORD SENDAD      ;SET LOCATION 46
(1) 000052 000000                       .=52
(1)         000764                       .WORD 0           ;SET LOCATION 52 = 0
(1)                                     .=$SVPC          ;RESTORE LOCATION CTR
(1)
1945                                     ;TTY INTERRUPT VECTOR*****
1946
1947         000060                       .=60
1948 000060 012372                       .WORD TTINT      ;TTY INTERRUPT HANDLER ADDRESS
1949 000062 000340                       .WORD 340        ;PRIORITY LEVEL 7
1950
1951                                     ;SOFTWARE SWITCH REGISTER*****
1952                                     ;USED IF HARDWARE SWR = 177777, OR NOT AVAIL.
1953         000176                       .=176
1954 000176 000000 . SWREG: .WORD 0
1955
1956                                     ;START ADDRESS*****
1957
1958         000200                       .=200
1959 000200 000137 001200                 JMP START ;PROGRAM START
1960
1961                                     ;RESTART ADDRESS*****
1962         000210                       .=210
1963 000210 000137 001670                 JMP ST2
1964
1965                                     ;TM03 INTERRUPT VECTOR*****
1966
1967         000224                       .=224
1968 000224 012362                       MTINT           ;TAPE INTERRUPT HANDLER ADDRESS
1969 000226 000340                       340
1970

```

```
1972
1973      000510      . = 510
1974      : MASS BUS REGISTER EQUIVS*****
1975
1976 000510 172440  C1: 172440
1977 000512 172442  WC: 172442
1978 000514 172444  BA: 172444
1979 000516 172446  FC: 172446
1980 000520 172450  CS: 172450
1981 000522 172452  DS: 172452
1982 000524 172454  ER: 172454
1983 000526 172456  AS: 172456
1984 000530 172460  CC: 172460
1985 000532 172462  DB: 172462
1986 000534 172464  MR: 172464
1987 000536 172466  DT: 172466
1988 000540 172470  SN: 172470
1989 000542 172472  TC: 172472
1990
1991      : ILLEGAL FUNCTION CODES
1992
1993 000544 005405  ILFT: 5405
1994 000546 007415      7415
1995 000550 016423      16423
1996 000552 020437      20437
1997 000554 022443      22443
1998 000556 025447      25447
1999 000560 031455      31455
2000 000562 033465      33465
2001 000564 036473      36473
2002
2003      : CONSTANTS*****
2004
2005 000566 177776  PSW: 177776      ; PROCESSOR STATUS
2006 000570 177570  SWR: 177570      ; SWITCH REGISTER
2007 000572 177560  TKS: 177560      ; TTY READER STATUS
2008 000574 177562  TKB: 177562      ; TTY READ BUFFER
2009 000576 177564  TPS: 177564      ; TTY PUNCH STATUS
2010 000600 177566  TPB: 177566      ; TTY PUNCH BUFFER
2011 000602 177777  SERNUM: 177777   ; SERIAL NUMBER
2012 000604 000011  DRVTP: 011       ; DRIVE TYPE
2013 000606 000020  ITAMT: 20        ; ITERATION AMOUNT
2014 000610 000224  VECT: 224        ; INTERRUPT VECTOR(RH)
2015 000612 172440  REGS: 172440     ; STARTING REGISTER ADDRESS
```

2017  
2018  
2019  
2020  
2021  
2022 000614  
2023 000614 000000  
2024 000616 000000  
2025 000620 000000  
2026 000622 000000  
2027 000624 000000  
2028 000626 000000  
2029 000630 000000  
2030 000632 000000  
2031 000634 000000  
2032 000636 000000  
2033 000640 000000  
2034 000642 000000  
2035 000644 000000  
2036 000646 000000  
2037 000650 000000  
2038 000652 000000  
2039 000654 000000  
2040 000656 000000  
2041 000660 000000  
2042 000662 000000  
2043 000664 000000  
2044 000666 000000  
2045 000670 000000  
2046 000672 000000  
2047 000674 000000  
2048 000676 000000  
2049 000700 000000  
2050 000702 000000  
2051 000704 000000  
2052 000706 000000  
2053 000710 000000  
2054 000712 000000  
2055 000714 000000  
2056 000716 000000  
2057 000720 000000  
2058 000722 000000  
2059 000724 000000  
2060 000726 000000  
2061 000730 000000  
2062 000732 000000  
2063 000734 000000  
2064 000736 000000  
2065 000740 000000  
2066 000742 000000  
2067 000744 000000  
2068 000746 000000  
2069 000750 000000  
2070 000752 000000  
2071 000754 000000  
2072 000756 000000

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

STFLGS:  
TOB: 0  
TIB: 0  
HDRFL: 0  
EMADDR: 0  
DRVN: 0  
TR00: 0  
TR01: 0  
TR02: 0  
TR03: 0  
TR04: 0  
TR05: 0  
TR06: 0  
TR07: 0  
TR10: 0  
TR11: 0  
TR12: 0  
TR13: 0  
TR14: 0  
TR15: 0  
NRZOF: 0  
SLVN: 0  
PFLG: 0  
RTRN: 0  
ERADD: 0  
TEMP1: 0  
TEMP2: 0  
TEMP3: 0  
ITCNT: 0  
SAV1: 0  
SAV2: 0  
SAV3: 0  
SCOLP: 0  
ITRLP: 0  
EXFL: 0  
ATAF: 0  
SLAF: 0  
SSCF: 0  
ERRF: 0  
ASF: 0  
SCF: 0  
TREF: 0  
PEXFL: 0  
STFLG: 0  
LTADD: 0  
T24FL: 0  
ADDFL: 0  
WAM: 0  
FUN: 0  
DATC: 0  
WTAD: 0

2073 000760 000000  
2074 000762 000000  
2075 000764 000000  
2076 000766 000000  
2077 000770 000000  
2078 000772 000000  
2079 000774 000000  
2080 000776 000000  
2081 001000 000000  
2082 001002 000000  
2083 001004 000000  
2084 001006 000000  
2085 001010 000000  
2086 001012 000000  
2087 001014 000000  
2088 001016 000000  
2089 001020 000000  
2090 001022 000000  
2091 001024  
2092  
2093  
2094  
2095 001024 000000  
2096 001026 000000  
2097 001030 000000  
2098 001032 000000  
2099  
2100  
2101  
2102 001034  
2103 001034 005162  
2104 001036 005202  
2105 001040 005206  
2106 001042 005214  
2107  
2108  
2109  
2110 001044 000005  
2111 001046 000005  
2112 001050 000012  
2113 001052 000012  
2114 001054 000000  
2115 001056 000017  
2116 001060 000017  
2117 001062 000017  
2118 001064 000017  
2119 001066 000000

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

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

;EXPT WRAP STATUS\*\*\*\*\*

WCS1: 0  
WCS2: 0  
WDS: 0  
WER: 0

;DATA PATTERN GENERATORS\*\*\*\*\*

DATBL:  
DATA0: DAT1 ;ALL ONE BITS  
DATA1: DAT2 ;ALL ZERO BITS  
DATA2: DAT3 ;ALTERNATING ONE/ZERO BITS  
DATA3: DAT4 ;ALTERNATING PARITY CHARACTERS

;CORE DUMP PATTERNS\*\*\*\*\*

WCDP2: 5  
5  
12  
12  
0  
WCDP0: 17  
17  
17  
17  
0

2121  
2122  
2123  
2124 001070 000000  
2125 001072 000000  
2126 001074 002436  
2127 001076 002436  
2128 001100 002550  
2129 001102 002550  
2130 001104 002622  
2131 001106 002622  
2132 001110 002730  
2133 001112 002730  
2134 001114 003002  
2135 001116 003002  
2136 001120 003110  
2137 001122 003110  
2138 001124 003166  
2139 001126 003166  
2140 001130 003274  
2141 001132 003274  
2142 001134 003346  
2143 001136 003346  
2144 001140 003460  
2145 001142 003460  
2146 001144 003606  
2147 001146 003606  
2148 001150 003656  
2149 001152 003656  
2150 001154 003726  
2151 001156 003726  
2152 001160 004010  
2153 001162 004010  
2154 001164 004376  
2155 001166 004376  
2156 001170 004710  
2157 001172 004710  
2158 001174 002330  
2159 001176 000020

:LOGIC TEST ENTRY TABLE\*\*\*\*\*

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

TADX: .WORD TEND  
TLAST: .WORD 20

;CONTAINS # OF TESTS

```

2161 .EVEN
2162 ;PROGRAM START AND HOUSEKEEPING*****
2163
2164 001200 012706 000500 START: MOV #500,SP ;SET STACK POINTER
2165 001204 013746 000004 MOV @#4,-(SP) ;SAVE ERROR TRAP
2166 001210 013746 000006 MOV @#6,-(SP)
2167 001214 012737 001240 000004 MOV #18,@#4 ;SET TIME OUT TRAP TO GO TO 18
2168 001222 005037 000006 CLR @#6
2169 001226 022777 177777 177334 CMP #177777,@SWR ;USE SOFTWARE SWITCH,IF SWR - 177777
2170 001234 001402 BEQ 2$ ;OR TIMES OUT
2171 001236 000404 BR 3$ ;OTHERWISE USE HARDWARE SWR
2172 001240 022626 1$: CMP (SP)+,(SP)+ ;RESET STACK
2173 001242 012737 000176 000570 2$: MOV #SWREG,SWR ;SET SWR = TO ADDRESS OF SOFTWARE SWR
2174 001250 012637 000006 3$: MOV (SP)+,@#6 ;RESTORE ERROR TRAP
2175 001254 012637 000004 MOV (SP)+,@#4
2176 001260 005027 CLR (PC)+ ;:CLEAR CHAIN INDICATOR
(1) 001262 000000 CHNFLG: .WORD 0 ;:CHAIN MODE INDICATOR
(1) ;:1/0 = CHAIN/NOT CHAIN MODE
(1) 001264 022737 002374 000042 CMP #SENDAD,@#42 ;:BRANCH IF LOADED VIA ACT11 CHAIN MODE
(1) 001272 001404 BEQ 50$
(1) 001274 005737 000042 TST @#42 ;:BRANCH IF IN DUMP MODE
(1) 001300 001413 BEQ 52$
(1) 001302 000406 BR 51$
(1) 001304 012737 000176 000570 50$: MOV #SWREG,SWR ;:INVOKE SOFTWARE SWR
(1) 001312 012777 100000 177250 MOV #100000,@SWR ;:WITH HALT ON ERROR SET
(1) 001320 005237 001262 51$: INC CHNFLG ;:SET CHNFLG = CHAIN MODE
(1) 001324 000137 001714 JMP TSCD ;:GO TO CHAIN ADDRESS
(1) 001330 52$:
2177 001330 122737 000006 000041 4$: CMPB #6,@#41 ;:BRANCH IF NOT LOADED VIA TMDP
2178 001336 001004 BNE 5$
2179 001340 012704 014573 MOV #MSG62,R4 ;:ADVISE USER TO REMOVE MEDIA FROM UUT
2180 001344 004737 013054 JSR PC,TTOUT
2181 001350 012704 013776 5$: MOV #MSG1,R4
2182 001354 004737 013054 JSR PC,TTOUT ;:PRINT TITLE
2183 001360 112737 000043 013776 MOVB #'#,MSG1 ;:DO NOT PRINT TITLE ON RESTART
2184 001366 012704 014467 MOV #MSG44,R4
2185 001372 004737 013054 JSR PC,TTOUT ;:REQUEST REGISTER ADDRESS
2186 001376 013703 000612 MOV REGS,R3
2187 001402 004737 013202 JSR PC,OCIP ;:PRINT CURRENT ADDRESS
2188 001406 012705 000612 MOV #REGS,R5 ;:SET ADDRESS SAVE LOC
2189 001412 012701 000007 MOV #7,R1 ;:SET SIZE OF RESPONSE
2190 001416 012702 176400 MOV #176400,R2 ;:SET UPPER LIMIT
2191 001422 012703 172300 MOV #172300,R3 ;:SET LOWER LIMIT
2192 001426 004737 012514 JSR PC,TTR ;:GO GET RESPONSE
2193 001432 012704 014511 MOV #MSG45,R4
2194 001436 004737 013054 JSR PC,TTOUT ;:REQUEST VECTOR
2195 001442 013703 000610 MOV VECT,R3
2196 001446 004737 013202 JSR PC,OCIP ;:PRINT CURRENT VECTOR
2197 001452 012705 000610 MOV #VECT,R5 ;:SET ADDRESS SAVE LOC
2198 001456 012701 000004 MOV #4,R1 ;:SET SIZE OF RESPONSE
2199 001462 012702 000224 MOV #224,R2 ;:SET UPPER LIMIT
2200 001466 012703 000150 MOV #150,R3 ;:SET LOWER LIMIT
2201 001472 004737 012514 JSR PC,TTR ;:GO GET RESPONSE
2202 001476 013700 000610 MOV VECT,R0 ;:GET VECTOR
2203 001502 012720 012362 MOV #MTINT,(R0)+ ;:LOAD INTERRUPT ADDRESS IN VECTOR
2204 001506 012710 000340 MOV #340,(R0) ;:LOAD PRIORITY

```

2205	001512	013700	000612	MOV	REGS,R0	;GET START OF REGS
2206	001516	012701	000016	MOV	#16,R1	;SET NUMBER OF REGS
2207	001522	012702	000510	MOV	#C1,R2	;GET START OF TABLE
2208	001526	010022		6\$: MOV	R0,(R2)+	;BUILD TABLE
2209	001530	062700	000002	ADD	#2,R0	;BUMP ADDRESS
2210	001534	005301		DEC	R1	;SEE IF DONE
2211	001536	001373		BNE	6\$	;IF NOT: BR
2212	001540	012702	000614	MOV	#STFLGS,R2	
2213	001544	012700	000210	MOV	#ENDFLG-STFLGS,R0	;GET # OF FLAGS TO CLEAR
2214	001550	006200		ASR	R0	;FORM COUNT
2215	001552	005022		7\$: CLR	(R2)+	;CLEAR FLAGS + COUNTERS
2216	001554	005300		DEC	R0	
2217	001556	001375		BNE	7\$	
2218	001560	012704	014533	MOV	#MSG57,R4	;REQUEST TM03 DRIVE #
2219	001564	004737	013054	JSR	PC,TTOUT	
2220	001570	013703	000624	MOV	DRVN,R3	;GET CURRENT DRIVE
2221	001574	004737	013202	JSR	PC,OCTP	;AND TYPE IT
2222	001600	012705	000624	MOV	#DRVN,R5	;TTR ROUTINE RETURNS DRIVE TO (R5)
2223	001604	012701	000002	MOV	#2,R1	;LIMIT RESPONSE TO 1 CHARACTER
2224	001610	012702	000007	MOV	#7,R2	;BETWEEN 0 AND 7
2225	001614	012703	000000	MOV	#0,R3	
2226	001620	004737	012514	JSR	PC,TTR	;GET RESPONSE & PUT IN DRVN
2227	001624	012704	014551	MOV	#MSG58,R4	;REQUEST SLAVE #
2228	001630	004737	013054	JSR	PC,TTOUT	
2229	001634	013703	000664	MOV	SLVN,R3	;GET CURRENT SLAVE #
2230	001640	004737	013202	JSR	PC,OCTP	;AND TYPE IT
2231	001644	012705	000664	MOV	#SLVN,R5	;TTR ROUTINE RETURNS REPONSE TO (R5)
2232	001650	012701	000002	MOV	#2,R1	;LIMIT RESPONSE TO 1 CHARACTER
2233	001654	012702	000007	MOV	#7,R2	;BETWEEN 0-7
2234	001660	012703	000000	MOV	#0,R3	
2235	001664	004737	012514	JSR	PC,TTR	;GET RESPONSE & PUT IT IN SLVN
2236						
2237				:START 210		
2238	001670	012706	000500	ST2: MOV	#500,SP	;SET STACK PTR
2239	001674	005037	001006	CLR	RDRVF	;CLEAR READ REVERSE FLAG
2240	001700	005037	001016	CLR	PCNTR	
2241	001704	005037	001022	CLR	CRCFLG	;SET CRC FLAG = CRC NOT IN PROGRESS
2242	001710	004737	013634	JSR	PC,GTSWR	;GET SOFTWARE SWITCHES

```

2244
2245 ;TEST SCHEDULAR*****
2246
2247 001714 052777 000100 176650 TSCD: BIS #100,@TKS ;SET KEYBOARD IE BIT
2248 001722 005037 001000 CLR WPGFL ;CLEAR WRAP PATRN FLAG
2249 001726 005037 000740 CLR STFLG ;CLEAR SINGLE TEST FLAG
2250 001732 017700 176632 MOV @SWR,RO
2251 001736 042700 177700 BIC #177700,RO ;BRANCH IF SINGLF TEST SELECTED
2252 001742 001154 BNE STSCD ;GO SELECT SINGLE TEST
2253 001744 005737 001262 TST CHNFLG ;:BRANCH IF NOT IN CHAIN MODE
(1) 001750 001511 BEQ TSCDA
(1) 001752 012737 177777 000624 MOV #-1,DRVN ;;INITIALIZE DRIVE #
(1) 001760 012737 177777 000664 NXTDRV: MOV #-1,SLVN ;;INITIALIZE SLAVE #
(1) 001766 012777 000040 176524 1$: MOV #40,@CS ;;INIT CONTROLLER
(1) 001774 005237 000624 INC DRVN ;;STEP DRIVE #
(1) 002000 022737 00C010 000624 CMP #10,DRVN ;;EXIT IF ALL DRIVES TESTED
(1) 002006 001553 BEQ $DONE ;;FOR AVAILABILITY
(1) 002010 013777 000624 176502 MOV DRVN,@CS ;;LOAD DRIVE #
(1) 002016 005777 176466 TST @C1 ;;ACCESS DRIVE
(1) 002022 032777 010000 176470 BIT #10000,@CS ;BRANCH IF DRIVE NON EXISTANT
(1) 002030 001356 BNE 1$ ;:(NED = 1)
(1) 002032 005237 000664 NXTSLV: INC SLVN ;;STEP SLAVE # AND BRANCH
(1) 002036 001011 BNE 1$ ;;IF NOT SLAVE 0
(1) 002040 005737 000624 TST DRVN ;;BRANCH IF NOT DRIVE # 0
(1) 002044 001006 BNE 1$
(1) 002046 122737 000006 000041 CMPB #6,@#41 ;:BRANCH IF NOT TMDP
(1) 002054 001002 BNE 1$
(1) 002056 005237 000664 INC SLVN ;;STEP TO SLAVE # 1
(1) 002062 022737 000010 000664 1$: CMP #10,SLVN ;:BRANCH IF ALL SLAVES TESTED
(1) 002070 001733 BEQ NXTDRV ;:FOR AVAILABILITY
(1) 002072 013777 000664 176442 MOV SLVN,@TC ;:LOAD SLAVE UNIT #
(1) 002100 032777 002000 176430 BIT #2000,@DT ;:BRANCH IF SLAVE NOT
(1) 002106 001751 BEQ NXTSLV ;:PRESENT (SPR = 0)
2254 002110 032777 010000 176404 BIT #10000,@DS ;TEST FOR MOL
2255 002116 001026 BNE TSCDA ;BRANCH IF MOL
2256 002120 012704 014667 MOV #MSG64,R4 ;ELSE PRINT NOT ON LINE
2257 002124 004737 013054 JSR PC,TTOUT ;PRINT MESSAGE
2258 002130 013703 000624 MOV DRVN,R3 ;DRIVE NUMBER
2259 002134 004737 013202 JSR PC,OCTP ;PRINT NUMBER
2260 002140 012704 014701 MOV #MSG65,R4 ;ADDRESS OF MESSAGE
2261 002144 004737 013054 JSR PC,TTOUT ;PRINT MESSAGE
2262 002150 013703 000664 MOV SLVN,R3 ;SLAVE NUMBER
2263 002154 004737 013202 JSR PC,OCTP ;PRINT NUMBER
2264 002160 012704 014711 MOV #MSG66,R4 ;ADDR OF MESSAGE
2265 002164 004737 013054 JSR PC,TTOUT ;PRINT IT
2266 002170 000137 002032 JMP NXTSLV
2267 002174 012737 001070 000742 TSCDA: MOV #TSTTBL,LTADD
2268 002202 062737 000004 000742 TSCD0: ADD #4,LTADD
2269 002210 013737 000742 000714 TSCD1: MOV LTADD,ITRLP
2270 002216 062737 000002 000714 ADD #2,ITRLP ;SET ITERATION ADDRESS
2271 002224 005037 000620 CLR HDRFL ;CLEAR PRINT HEADER FLAG
2272 002230 017700 176506 MOV @LTADD,RO ;SET POINTER TO TEST
2273 002234 000110 JMP (RO) ;GO TO TEST
2274 002236 032777 002000 176324 TSCD2: BIT #2000,@SWR ;SEE IF HALT ON TEST
2275 002244 001403 BEQ TSCD3 ;IF NOT: BR
2276 002246 000000 HALT

```

```

2277 002250 005037 001000          CLR      WPGFL      ;CLEAR WRAP DATA GENERATOR FLAG
2278 002254 005737 000740 TSCD3:  TST      STFLG      ;SE IF SINGLE TEST
2279 002260 001750          BEQ      TSCD0      ;IF NOT: BR
2280 002262 017700 176302          MOV      @SWR,RO
2281 002266 042700 177700          BIC      #177700,RO ;MASK TEST NUMBER
2282 002272 001610          BEQ      TSCD      ;IF SO: BR
2283 002274 012737 000001 000740 STSCD:  MOV      #1,STFLG   ;SET SINGLE TEST FLAG
2284 002302 023700 001176          CMP      TLAST,RO  ;SEE IF EXCEEDED TESTS
2285 002306 002410          BLT      TEND      ;IF SO: BR
2286 002310 006300          ASL      RO
2287 002312 006100          ROL      RO        ;SET TABLE MODIFIER
2288 002314 012737 001070 000742          MOV      #TSTTBL,LTADD
2289 002322 060037 000742          ADD      RO,LTADD  ;SET TEST POINTER
2290 002326 000730          BR      TSCD1
2291 002330 005737 001262          TEND:   TST      CHNFLG ;BRANCH IF IN CHAIN MODE
2292 002334 001236          BNE      NXTSLV
2293 002336 012704 014450 $DONE:  MOV      #MSG41,R4
2294 002342 004737 013054          JSR      PC,T1OUT  ;PRINT END OF PASS
2295 002346 013703 001016          MOV      PCNTR,R3
2296 002352 004737 013202          JSR      PC,OCIP  ;PRINT PASS NUMBER
2297 002356 005000          CLR      RO
2298 002360 005300          1$:    DEC      RO        ;DELAY WAITING FOR
2299 002362 001376          BNE      1$        ;TTY TO FINISH
2300 002364 013700 000042          MOV      @#42,RO  ;GET ACT11 RETURN ADDRESS
    (1) 002370 001405          BEQ      HERE      ;BRANCH IF NOT ACT11
    (1) 002372 000005          RESET
    (1) 002374 004710          $ENDAD: JSR      PC,(RO)
    (1) 002376 000240          NOP
    (1) 002400 000240          NOP
    (1) 002402 000240          NOP
    (1) 002404 000240          HERE:   NOP
2301 002406 005737 001262          TST      CHNFLG   ;BRANCH IF IN CHAIN MODE
2302 002412 001005          BNE      TENDX
2303 002414 032777 010000 176146          BIT      #10000,@SWR ;SEE IF HALT ON PASS
2304 002422 001001          BNE      TENDX    ;IF NOT: BR
2305 002424 000000          HALT
2306 002426 005237 001016          TENDX:  INC      PCNTR  ;BUMP PASS COUNTER
2307 002432 000137 001714          JMP      TSCD     ;RESTART
2308
    
```

```
2310 ;THESE TESTS CHECK DATA FORMATTING
2311 ;AND TRANSFER THROUGH THE TM03 WRAP AROUND MODFS
2312
2313 ;LOGIC TEST 1: WRAP 3, NRZ, NORMAL ODD *****
2314
2315 002436 012737 004270 001024 LT1: MOV #4270,WCS1 ;SET EXPT CS1
2316 002444 012737 000100 001026 MOV #100,WCS2 ;SET EXPT CS2
2317 002452 012737 010600 001030 MOV #10600,WDS ;SET EXPT DS
2318 002460 012737 000000 001032 MOV #0,WER ;SET EXPT ER
2319 002466 012737 014726 000622 MOV #MSLT1,EMADDR ;SET HEADER
2320 002474 012737 001700 000776 MOV #1700,UDES ;SET NRZ,NORMAL, ODD
2321 002502 005037 001002 LT1A: CLR PATRN ;POINT TO PATTERN 0
2322 002506 012737 002514 000712 MOV #LT1B,SCOLP ;SET SCOPE ADDRESS
2323 002514 004737 005342 LT1B: JSR PC,WAM3 ;GO DO WRAP 3
2324 002520 005237 001002 INC PATRN ;BUMP PATTERN POINTER
2325 002524 032737 000004 001002 BIT #4,PATRN ;SEE IF DONE
2326 002532 001770 BEQ LT1B ;IF NOT: BR
2327 002534 004737 012136 JSR PC,ITER ;GO SEE IF ITERATIONS
2328 002540 005037 001006 CLR RDRVF ;CLEAR READ REVERSE FLAG
2329 002544 000137 002236 JMP TSCD2 ;RETURN TO SCHEDULAR
2330
2331 ;LOGIC TEST 2: WRAP 3, PE, NORMAL, ODD*****
2332
2333 002550 000240 LT2: NOP
2334 002552 012737 004270 001024 LT2A: MOV #4270,WCS1 ;SET EXPT CS1
2335 002560 012737 000100 001026 MOV #100,WCS2 ;SET EXPT CS2
2336 002566 012737 010640 001030 MOV #10640,WDS ;SET EXPT DS
2337 002574 012737 000000 001032 MOV #0,WER ;SET EXPT WER
2338 002602 012737 014774 000622 MOV #MSLT2,EMADDR ;SET HEADER
2339 002610 012737 002300 000776 MOV #2300,UDES ;SET PE, NORMAL, ODD
2340 002616 000137 002502 JMP LT1A ;EXECUTE TEST SEQUENCE
2341
2342 ;LOGIC TEST 3: WRAP 2, NRZ, NORMAL, ODD*****
2343
2344 002622 012737 004260 001024 LT3: MOV #4260,WCS1 ;SET EXPT CS1
2345 002630 012737 000100 001026 MOV #100,WCS2 ;SET EXPT CS2
2346 002636 012737 010600 001030 MOV #10600,WDS ;SET EXPT DS
2347 002644 012737 000000 001032 MOV #0,WER ;SET EXPT WER
2348 002652 012737 015041 000622 MOV #MSLT3,EMADDR ;SET HEADER
2349 002660 012737 001700 000776 MOV #1700,UDES ;SET TO NRZ,NORMAL, ODD
2350 002666 005037 001002 LT3A: CLR PATRN ;POINT TO PATTERN 0
2351 002672 012737 002700 000712 MOV #LT3B,SCOLP ;SET SCOPE ADDRESS
2352 002700 004737 005276 LT3B: JSR PC,WAM2 ;GO DO WRAP 2
2353 002704 005237 001002 INC PATRN ;BUMP POINTER
2354 002710 032737 000004 001002 BIT #4,PATRN ;SEE IF DONE
2355 002716 001770 BEQ LT3B ;IF NOT: BR
2356 002720 004737 012136 JSR PC,ITER ;GO SEE IF ITERATIONS
2357 002724 000137 002236 JMP TSCD2 ;RETURN TO SCHEDULAR
```

```

2359
2360 ;LOGIC TEST 4: WRAP 2, PE, NORMAL, ODD*****
2361
2362 002730 000240 LT4: NOP
2363 002732 012737 004260 001024 LT4A: MOV #4260,WCS1 ;SET EXPT CS1
2364 002740 012737 000100 001026 MOV #100,WCS2 ;SET EXPT CS2
2365 002746 012737 010640 001030 MOV #10640,WDS ;SET EXPT DS
2366 002754 012737 000000 001032 MOV #0,WER ;SET EXPT WER
2367 002762 012737 015107 000622 MOV #MSLT4,EMADDR ;SET HEADER
2368 002770 012737 002300 000776 MOV #2300,UDES ;SET PE, NORMAL, ODD
2369 002776 000137 002666 JMP LT3A ;GO EXECUTE TEST SEQUENCES
2370
2371 ;LOGIC TEST 5: WRAP 1, NRZ, NORMAL, ODD*****
2372
2373 003002 012737 004260 001024 LT5: MOV #4260,WCS1 ;SET EXPT CS1
2374 003010 012737 000100 001026 MOV #100,WCS2 ;SET EXPT CS2
2375 003016 012737 010600 001030 MOV #10600,WDS ;SET EXPT DS
2376 003024 012737 000000 001032 MOV #0,WER ;SET EXPT WER
2377 003032 012737 015154 000622 MOV #MSLT5,EMADDR ;SET HEADER
2378 003040 012737 001700 000776 MOV #1700,UDES ;SET NRZ, NORMAL, ODD
2379 003046 005037 001002 LT5A: CLR PATRN ;POINT TO PATTERN ZERO
2380 003052 012737 003060 000712 MOV #LT5B,SCOLP ;SET SCOPE ADDRESS
2381 003060 004737 005266 LT5B: JSR PC,WAM1 ;GO DO WRAP 1
2382 003064 005237 001002 INC PATRN ;BUMP POINTER
2383 003070 032737 000004 001002 BIT #4,PATRN ;SEE IF DONE
2384 003076 001770 BEQ LT5B ;IF NOT: BR
2385 003100 004737 012136 JSR PC,ITER ;GO SEE IF ITERATIONS
2386 003104 000137 002236 JMP TSCD2 ;RETURN TO SCHEDULAR
2387
2388 ;LOGIC TEST 6: WRAP 1, PE, NORMAL, ODD*****
2389
2390 003110 000240 LT6: NOP
2391 003112 004737 011634 LT6A: JSR PC,PPGEN ;GO GENERATE PRE/POSTAMBLE
2392 003116 012737 004260 001024 MOV #4260,WCS1 ;SET EXPT CS1
2393 003124 012737 000100 001026 MOV #100,WCS2 ;SET EXPT CS2
2394 003132 012737 010640 001030 MOV #10640,WDS ;SET EXPT DS
2395 003140 012737 000000 001032 MOV #0,WER ;SET EXPT WER
2396 003146 012737 015222 000622 MOV #MSLT6,EMADDR ;SET HEADER
2397 003154 012737 002300 000776 MOV #2300,UDES ;SET PE, NORMAL, ODD
2398 003162 000137 003046 JMP LT5A ;GO EXECUTE TEST SEQUENCE

```

```
2400
2401 ;LOGIC TEST 7: WRAP 0, NRZ,NORMAL, ODD*****
2402
2403 003166 012737 144260 001024 LT7: MOV #144260,WCS1 ;SET EXPT CS1
2404 003174 012737 000100 001026 MOV #100,WCS2 ;SET EXPT CS2
2405 003202 012737 150600 001030 MOV #150600,WDS ;SET EXPT DS
2406 003210 012737 000200 001032 MOV #200,WER ;SET EXPT ER
2407 003216 012737 015267 000622 MOV #MSLT7,EMADDR ;SET HEADER
2408 003224 012737 001700 000776 MOV #1700,UDES ;SET NRZ, NORMAL, ODD
2409 003232 005037 001002 LT7A: CLR PATRN ;POINT TO PATTERN 0
2410 003236 012737 003244 000712 MOV #LT7B,SCOLP ;SET SCOPE ADDRESS
2411 003244 004737 005222 LT7B: JSR PC,WAMO ;GO DO WRAP 0
2412 003250 005237 001002 INC PATRN ;BUMP POINTER
2413 003254 032737 000004 001002 BIT #4,PATRN ;SEE IF DONE
2414 003262 001770 BEQ LT7B ;IF NOT: BR
2415 003264 004737 012136 JSR PC,ITER ;GO SEE IF ITERATIONS
2416 003270 000137 002236 JMP TSCD2 ;RETURN TO SCHEDULAR
2417
2418 ;LOGIC TEST 10: WRAP 0, PE, NORMAL, ODD*****
2419
2420 003274 000240 LT10: NOP
2421 003276 012737 004260 001024 LT10A: MOV #4260,WCS1 ;SET EXPT CS1
2422 003304 012737 000100 001026 MOV #100,WCS2 ;SET EXPT CS2
2423 003312 012737 010640 001030 MOV #10640,WDS ;SET EXPT DS
2424 003320 012737 000000 001032 MOV #0,WER ;SET EXPT ER
2425 003326 012737 015335 000622 MOV #MSLT10,EMADDR ;SET HEADER
2426 003334 012737 002300 000776 MOV #2300,UDES ;SET PE, NORMAL, ODD
2427 003342 000137 003232 JMP LT7A ;GO EXECUTE TEST SEQUENCE
```

```
2429 ;LOGIC TEST 11: CORE DUMP WRITE, WAM2*****
2430
2431 003346 012737 004260 001024 LT11: MOV #4260,WCS1 ;SET EXPT CS1
2432 003354 012737 000100 001026 MOV #100,WCS2 ;SET EXPT CS2
2433 003362 012737 010600 001030 MOV #10600,WDS ;SET EXPT DS
2434 003370 012737 000000 001032 MOV #0,WER ;SET EXPT ER
2435 003376 012737 015403 000622 MOV #MSLT11,EMADDR ;SET HEADER
2436 003404 012737 001720 000776 MOV #1720,UDES ;SET NRZ, CORE DUMP, ODD
2437 003412 005037 001002 CLR PATRN ;POINT TO PATTERN 0
2438 003416 012737 003424 000712 MOV #LT11A,SCOLP ;SET SCOPE ADDRESS
2439 003424 004737 005276 LT11A: JSR PC,WAM2 ;GO DO WAM 2
2440 003430 022737 000002 001002 CMP #2,PATRN ;SEE IF DONE
2441 003436 001404 BEQ LT11X ;IF SO: BR
2442 003440 012737 000002 001002 MOV #2,PATRN ;SELECT PATTERN 2
2443 003446 000766 BR LT11A ;CONTINUE
2444 003450 004737 012136 LT11X: JSR PC,ITER ;GO SEE IF ITERATIONS
2445 003454 000137 002236 JMP TSCD2 ;RETURN TO SCHEDULES
2446
2447 ;LOGIC TEST 12: CORE DUMP READ, WAM 3*****
2448
2449 003460 012737 004270 001024 LT12: MOV #4270,WCS1 ;SET EXPT CS1
2450 003466 012737 000100 001026 MOV #100,WCS2 ;SET EXPT CS2
2451 003474 012737 010600 001030 MOV #10600,WDS ;SET EXPT DS
2452 003502 012737 000000 001032 MOV #0,WER ;SET EXPT ER
2453 003510 012737 015454 000622 MOV #MSLT12,EMADDR ;SET HEADER
2454 003516 012737 001720 000776 MOV #1720,UDES ;SELECT NRZ, CORE DUMP, ODD
2455 003524 005037 001002 CLR PATRN ;SELECT PATTERN 0
2456 003530 012737 003544 000712 MOV #LT12A,SCOLP ;SET SCOPE ADDRESS
2457 003536 012737 001056 001010 MOV #WCDP0,RCDP ;POINT TO PATTERN 0
2458 003544 004737 005342 LT12A: JSR PC,WAM3 ;GO DO WAM3
2459 003550 022737 000002 001002 CMP #2,PATRN ;SEE IF DONE
2460 003556 001407 BEQ LT12X ;IF SO: BR
2461 003560 012737 000002 001002 MOV #2,PATRN ;SELECT PATTERN 2
2462 003566 012737 001044 001010 MOV #WCDP2,RCDP ;POINT TO PATTERN 2
2463 003574 000763 BR LT12A ;CONTINUE
2464 003576 004737 012136 LT12X: JSR PC,ITER ;GO SEE IF ITERATION
2465 003602 000137 002236 JMP TSCD2 ;RETURN TO SCHEDULE
```

```
2467
2468 ;LOGIC TEST 13: EVEN PARITY WRITE: WAM 1(M8933)*****
2469
2470 003606 012737 004260 001024 LT13: MOV #4260,WCS1 ;SET EXPT CS1
2471 003614 012737 000100 001026 MOV #100,WCS2 ;SET EXPT CS2
2472 003622 012737 010600 001030 MOV #10600,WDS ;SET EXPT DS
2473 003630 012737 000000 001032 MOV #0,WER ;SET EXPT ER
2474 003636 012737 015524 000622 MOV #MSLT13,EMADDR ;SET HEADER
2475 003644 012737 001710 000776 MOV #1710,UDES ;SET NRZ, NORMAL, EVEN
2476 003652 000137 003046 JMP LT5A ;GO EXECUTE WAM 1
2477
2478 ;LOGIC TEST 14: EVEN PARITY READ: WAM 0(M8933 M8934)*****
2479
2480 003656 012737 144260 001024 LT14: MOV #144260,WCS1 ;SET EXPT CS1
2481 003664 012737 000100 001026 MOV #100,WCS2 ;SET EXPT CS2
2482 003672 012737 150600 001030 MOV #150600,WDS ;SET EXPT DS
2483 003700 012737 000200 001032 MOV #200,WER ;SET EXPT ER
2484 003706 012737 015605 000622 MOV #MSLT14,EMADDR ;SET HEADER
2485 003714 012737 001710 000776 MOV #1710,UDES ;SET NRZ, NORMAL, EVEN
2486 003722 000137 003232 JMP LT7A ;GO DO WAM 0
2487
2488 ;LOGIC TEST 15: READ REVERSE: WAM 3(M8906)*****
2489
2490 003726 012737 004276 001024 LT15: MOV #4276,WCS1 ;SET EXPT CS1
2491 003734 012737 000100 001026 MOV #100,WCS2 ;SET EXPT CS2
2492 003742 012737 010640 001030 MOV #10640,WDS ;SET EXPT DS
2493 003750 012737 000000 001032 MOV #0,WER ;SET EXPT ER
2494 003756 012737 015664 000622 MOV #MSLT15,EMADDR ;SET HEADER
2495 003764 012737 002300 000776 MOV #2300,UDES ;SELECT PE,NORMAL,ODD
2496 003772 000240 NOP
2497 003774 000240 NOP
2498 003776 012737 000001 001006 MOV #1,RDRVF ;SET READ REVERSE FLAG
2499 004004 000137 002502 JMP LT1A ;GO DO WAM 3, REVERSE
2500
```

```

2502
2503      :LOGIC TEST 16: CRC CORRECTION-SINGLE TRACK,EVERY FRAME
2504      :THIS IS A TEST OF THE CRC CORRECTION LOGIC. THE TEST WRITES
2505      :A KNOWN PATTERN (ALL 1'S , ALL 0'S & 125252) WITH A DATA BIT
2506      :ALTERED IN EACH OF THE DATA TRACKS. THIS TEST INSURES THAT A
2507      :CRC CORRECTABLE ERROR IS CORRECTED.
2508      :THE TEST PROCEEDS AS FOLLOWS:
2509
2510      : STEP A  WRITE A KNOWN PATTERN USING WRAP AROUND MODE 0
2511
2512      : STEP B  REWRITE THE PATTERN ABOVE WITH DATA BIT(S) MODIFIED
2513      :          IN TRACKS SPECIFIED BY CRCPAT USING WRAP AROUND MODE 4
2514      :          THIS WILL GENERATE A CRC ERROR
2515
2516      : STEP C  EXECUTE A READ REVERSE USING WRAP AROUND MODE 3
2517
2518      : STEP D  REPEAT STEP B ABOVE. UPON COMPLETION THE DATA READ
2519      :          BACK WILL MATCH THE DATA WRITTEN IN STEP A.
2520
2521
2522 004010 005037 001002      LT16: CLR      PATRN      ;SELECT PATTERN # 0 (ALL 1'S)
2523 004014 012737 000401 004366  MOV      #401,CRCPAT  ;SELECT BITS TO BE ALTERED (TRACK 1)
2524 004022 012737 015731 000622  MOV      #MSLT16,EMADDR ;SET ERROR MESSAGE HEADER
2525 004030 012737 004036 000712  MOV      #LT16A,SCOLP  ;SET SCOPE LOOP
2526
2527 004036 012737 144260 001024  LT16A: MOV      #144260,WCS1 ;SET EXPECTED VALUES UPON COMPLETION
2528 004044 012737 000100 001026  MOV      #100,WCS2
2529 004052 012737 150600 001030  MOV      #150600,WDS
2530 004060 012737 000200 001032  MOV      #200,WER
2531 004066 012737 001700 000776  MOV      #1700,UDES      ;SET UNIT DESCRIPTION-NRZ,800BPI,ODD
2532                                ;PARITY & PDP11 NORMAL MODE
2533 004074 005037 001022      CLR      CRCFLG      ;CLEAR CRC CORRECTION FLAG
2534 004100 004737 005222      JSR      PC,WAMO      ;DO A WRAP 0 --- STEP A
2535 004104 012737 000001 001022  MOV      #1,CRCFLG    ;SET CRC ERROR CORRECTION IN PROGRESS
2536 004112 013737 017474 004370  MOV      WBUFF,CRCDAT ;GET DATA WRITTEN BY WRAP 0
2537                                ;XOR CRCPAT WITH CRCDAT
2538 004120 013737 004370 004372  MOV      CRCDAT,XORDAT ;GET DATA TO BE MODIFIED
2539 004126 013737 004366 004374  MOV      CRCPAT,XORPAT ;GET MODIFIER
2540 004134 043737 004366 004372  BIC      CRCPAT,XORDAT ;CLEAR SET BITS IN DATA TO BE MODIFIED
2541 004142 043737 004370 004374  BIC      CRCDAT,XORPAT ;CLEAR SETTING BITS
2542 004150 053737 004374 004372  BIS      XORPAT,XORDAT ;SET CLEAR BITS IN DATA TO BE MODIFIED
2543 004156 013737 004372 004370  MOV      XORDAT,CRCDAT ;RESTORE MODIFIED DATA
2544
2545 004164 012737 144260 001024  MOV      #144260,WCS1 ;SET EXPECTED VALUES UPON COMPLETION
2546 004172 012737 000110 001026  MOV      #110,WCS2    ;OF WRAP 4
2547 004200 012737 150600 001030  MOV      #150600,WDS
2548 004206 012737 100300 001032  MOV      #100300,WER
2549 004214 004737 005450      JSR      PC,WAM4      ;DO A WRAP 4 --- STEP B
2550 004220 000240      NOP
2551 004222 012737 000001 001006  MOV      #1,RDRVF     ;SET TO READ REVERSE
2552 004230 012737 144276 001024  MOV      #144276,WCS1 ;SET EXPECTED VALUES UPON COMPLETION
2553 004236 012737 000110 001026  MOV      #110,WCS2
2554 004244 012737 150600 001030  MOV      #150600,WDS
2555 004252 012737 001000 001032  MOV      #1000,WER
2556 004260 004737 005342      JSR      PC,WAM3      ;DO A WRAP 3 --- STEP C
2557 004264 000240      NOP

```

2558	004266	012737	004260	001024	MOV	#4260,WCS1	
2559	004274	012737	000110	001026	MOV	#110,WCS2	
2560	004302	012737	010600	001030	MOV	#10600,WDS	
2561	004310	012737	000000	001032	MOV	#0,WER	
2562	004316	005037	001022		CLR	CRCFLG	;CLEAR CRC CORRECTION IN PROGRESS FLAG
2563	004322	004737	005450		JSR	PC,WAM4	;GO TO WRAP 4 --- STEP D
2564							
2565	004326	006337	004366		ASL	CRCPAT	;SELECT NEXT TRACK TO BE ALTERED
2566	004332	103241			BCC	LT16A	;CONTINUE FOR ALL TRACKS
2567	004334	012737	000401	004366	MOV	#401,CRCPAT	;RESET BITS TO TRACK 1
2568	004342	005237	001002		INC	PATRN	;SELECT NEXT PATTERN
2569	004346	022737	000003	001002	CMP	#3,PATRN	;BRANCH IF NOT DONE
2570	004354	001230			BNE	LT16A	
2571	004356	004737	012136		JSR	PC,ITER	;ITERATION LOOP
2572	004362	000137	002236		JMP	TSCD2	;RETURN TO SCHEDULER
2573							
2574							
2575	004366	000000			CRCPAT: .WORD	0	;CONTAINS BITS TO BE ALTERED
2576	004370	000000			CRCDAT: .WORD	0	;CONTAINS DATA TO BE WRITTEN BY WRAP4
2577	004372	000000			XORDAT: .WORD	0	;TEMPRARY STORAGE FOR XOR
2578	004374	000000			XORPAT: .WORD	0	;TEMPOARY STORAGE FOR XOR
2579							

```

2581
2582
2583 :LOGIC TEST 17: CRC CORRECTION - MULTIPLE BAD TRACKS
2584 :THIS TEST CHECKS THAT CRC ERROR CORRECTION IS NOT PERFORMED WHEN
2585 :MULTIPLE BAD TRACKS ARE DETECTED.
2586
2587 004376 012737 000002 001002 LT17: MOV #2,PATRN ;SELECT PATTERN #2 (125125)
2588 004404 012737 001001 004366 MOV #1001,CRCPAT ;SELECT 2 BAD TRACKS
2589 004412 012737 016021 000622 MOV #MSLT17,EMADDR ;SET ERROR MESSAGE HEADER
2590 004420 012737 004426 000712 MOV #LT17A,SCOLP ;SET SCOPE LOOP ADDRESS
2591 004426 012737 144260 001024 LT17A: MOV #144260,WCS1 ;SET EXPECTED VALUES ON COMPLETION
2592 004434 012737 000100 001026 MOV #100,WCS2 ;OF WRAP 0 BELOW
2593 004442 012737 150600 001030 MOV #150600,WDS
2594 004450 012737 000200 001032 MOV #200,WER
2595 004456 012737 001700 000776 MOV #1700,UDES ;SET UNIT DESCRIPTION
2596 004464 005037 001022 CLR CRCFLG ;SET CRC CORRECTION NOT IN PROGRESS
2597 004470 004737 005222 JSR PC,WAMO ;DO A WRAP 0 --- STEP A
2598 004474 000240 NOP
2599 004476 012737 000002 001022 MOV #2,CRCFLG ;SET CRC CORRECTION IN PROGRESS
2600 004504 013737 017474 004370 MOV WBUFF,CRCDAT ;GET DATA TO BE WRITTEN
2601 004512 043737 004366 004370 BIC CRCPAT,CRCDAT ;MODIFY DATA TO BE WRITTEN
2602 004520 012737 144260 001024 MOV #144260,WCS1 ;SET EXPECTED VALUES ON COMPLETION
2603 004526 012737 000110 001026 MOV #110,WCS2 ;OF WRAP 4 BELOW
2604 004534 012737 150600 001030 MOV #150600,WDS
2605 004542 012737 100300 001032 MOV #100300,WER
2606 004550 004737 005450 JSR PC,WAM4 ;DO A WRAP 4 --- STEP B
2607 004554 000240 NOP
2608 004556 012737 000001 001006 MOV #1,RDRVF ;SET READ REVERSE FLAG
2609 004564 012737 144276 001024 MOV #144276,WCS1 ;SET EXPECTED VALUES ON COMPLETION
2610 004572 012737 000110 001026 MOV #110,WCS2 ;OF WRAP 3 BELOW
2611 004600 012737 150600 001030 MOV #150600,WDS
2612 004606 012737 001000 001032 MOV #1000,WER
2613 004614 004737 005342 JSR PC,WAM3 ;DO A WRAP 3 --- STEP C
2614 004620 000240 NOP
2615 004622 012737 144260 001024 MOV #144260,WCS1 ;SET EXPECTED VALUES ON COMPLETION
2616 004630 012737 000110 001026 MOV #110,WCS2 ;OF WRAP 4 BELOW
2617 004636 012737 150600 001030 MOV #150600,WDS
2618 004644 012737 100100 001032 MOV #100100,WER
2619 004652 005037 001022 CLR CRCFLG ;CLEAR CRC IN PROGRESS FLAG
2620 004656 013701 004370 MOV CRCDAT,R1 ;GET DATA THAT WAS WRITTEN IN STEP B
2621 004662 012703 017474 MOV #WBUFF,R3 ;SET START OF WRITE BUFFER
2622 004666 004737 005166 JSR PC,DAT1A ;GO SET WRITE BUFFER
2623 004672 004737 005450 JSR PC,WAM4 ;GO DO A WRAP 4 --- STEP D
2624 004676 000240 NOP
2625 004700 004737 012136 JSR PC,ITER ;ITERATE TEST
2626 004704 000137 002236 JMP TSCD2 ;RETURN TO SCHEDULER
  
```

2628  
2629  
2630  
2631  
2632  
2633  
2634  
2635  
2636  
2637  
2638  
2639  
2640  
2641  
2642  
2643  
2644  
2645  
2646  
2647  
2648

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

```
LT20:  CLR  PATRN          ;SET PATTERN # 0 (ALL 1'S)
        MOV  #144276,WCS1  ;SET EXPECTED VALUES ON COMPLETION
        MOV  #100,WCS2
        MOV  #150600,WDS
        MOV  #100000,WER
        MOV  #MSLT20,EMADDR ;SET ERROR MESSAGE ADDRESS
        MOV  #1700,UDES    ;SET UNIT DESCRIPTION
        MOV  #LT20A,SCOLP  ;SET SCOPE LOOP ADDRESS
LT20A: MOV  #1,RDRVF       ;SET READ REVERSE FLAG
        MOV  #1,DCHKFL     ;SET DATA CHECK FLAG TO NOT CHACK DATA
        JSR  PC,WAM3
        CLR  RDRVF        ;CLEAR READ REVERSE FLAG
        CLR  DCHKFL      ;CLEAR DATA CHECK FLAG
        JSR  PC,ITER
        JMP  TSCD2       ;RETURN TO SCHEDULER
```

2650  
 2651  
 2652  
 2653  
 2654  
 2655  
 2656  
 2657  
 2658  
 2659  
 2660  
 2661  
 2662  
 2663  
 2664  
 2665  
 2666  
 2667  
 2668  
 2669  
 2670  
 2671  
 2672  
 2673  
 2674  
 2675  
 2676  
 2677  
 2678  
 2679  
 2680  
 2681  
 2682  
 2683  
 2684  
 2685  
 2686  
 2687  
 2688  
 2689  
 2690  
 2691  
 2692  
 2693  
 2694  
 2695  
 2696  
 2697  
 2698  
 2699  
 2700  
 2701  
 2702  
 2703  
 2704  
 2705

005026 005737 000740  
 005032 001434  
 005034 032777 000100 173526  
 005042 001430  
 005044 012704 016614  
 005050 004737 013054  
 005054 013703 001002  
 005060 004737 013202  
 005064 012705 001002  
 005070 012701 000002  
 005074 012702 000003  
 005100 012703 000000  
 005104 004737 012514  
 005110 112737 000001 001001  
 005116 113737 001002 001000  
 005124 012703 017474  
 005130 013701 001002  
 005134 006301  
 005136 004771 001034  
 005142 012702 000202  
 005146 012701 020106  
 005152 005021  
 005154 005302  
 005156 001375  
 005160 000207  
 005162 012701 177777  
 005166 012702 000202  
 005172 010123  
 005174 005302  
 005176 001375  
 005200 000207  
 005202 005001  
 005204 000770  
 005206 012701 125125

DSUP:  
 173526  
 001001  
 001000  
 DS0:  
 DS3:  
 DS4:  
 DAT1:  
 DAT1A:  
 1\$:  
 DAT2:  
 DAT3:

```

:*****
:DATA SETUP ROUTINE:
:THIS ROUTINE IS USED TO GENERATE THE DATA PATTERNS
:THE PATTERN TO BE WRITTEN IS SPECIFIED BY:
:PATRN =0      ALL 1'S
:        =1      ALL 0'S
:        =2      ALTERNATE 1'S & 0'S (125125)
:        =3      ALTERNATING PARITY (177377)
:*****

DSUP:  TST      STFLG      ;SEE IF SINGLE TEST
      BEQ      DSO        ;IF NOT: BR
      BIT      #100,@SWR  ;SEE IF SELECT PATTERN
      BEQ      DSO        ;IF NOT: BR
      MOV      #WMSG3,R4
      JSR      PC,TTOUT   ;REQUEST PATTERN NUMBER
      MOV      PATRN,R3
      JSR      PC,OC1P    ;PRINT PATTERN NUMBER
      MOV      #PATRN,R5  ;GET ADDRESS OF PATRN ENTRY
      MOV      #2,R1      ;SET SIZE OF ENTRY
      MOV      #3,R2      ;SET UPPER LIMIT
      MOV      #0,R3      ;SET LOWER LIMIT
      JSR      PC,TTR     ;GO GET PATTERN NUMBER
      MOV      #1,WPGFL+1 ;SET FLAG
      MOV      PATRN,WPGFL ;SET PATTERN NUMBER
      MOV      #WBUFF,R3  ;R3 = ADDR OF WRITE BUFFER
      MOV      PATRN,R1   ;R1 = PATTERN SELECTOR
      ASL      R1         ;MAKE PATTERN SELECTOR EVEN
      JSR      PC,@DATBL(R1) ;GO GENERATE PATTERN
      MOV      #202,R2    ;R2=BUFFER SIZE +2
      MOV      #RBUFF,R1  ;R1=READ DATA START
      CLR      (R1)+      ;CLEAR BUFFER
      DEC      R2         ;SEE IF DONE ALL
      BNE     DS4        ;IF NOT: BR
      RTS      PC        ;EXIT

:ALL ONES*****
DAT1:  MOV      #-1,R1    ;R1=DATA
DAT1A: MOV      #202,R2   ;R2=WORD COUNT +2
1$:   MOV      R1,(R3)+  ;LOAD BUFFER
      DEC      R2        ;SEE IF DONE
      BNE     1$        ;IF NOT: BR
      RTS      PC        ;RETURN TO CALLER

:ALL ZEROS*****
DAT2:  CLR      R1        ;R1=DATA
      BR      DAT1A     ;LOAD BUFFER

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

2706 005212 000765  
2707  
2708  
2709  
2710 005214 012701 177377  
2711 005220 000762  
2712

BR DAT1A ;LOAD BUFFER  
;ALTERNATING PARITY CHARACTERS\*\*\*\*\*  
DAT4: MOV #177377,R1 ;R1=ALTERNATING PARITY DATA  
BR DAT1A ;GO LOAD BUFFER

```

2714
2715 ;WRAP AROUND MODE 0 GLOBAL*****
2716
2717 005222 012737 000006 000750 WAM0: MOV #6,WAM ;SET WAM NUMBER
2718 005230 012737 000060 000752 WAM01: MOV #60,FUN
2719 005236 005037 000754 CLR DATC
2720 005242 012737 017474 000760 MOV #WBUFF,DATAD ;SET BUFFER ADDRESS
2721 005250 012737 020106 000762 MOV #RBUFF,RDAD ;SET POINTER TO READ BUFFER
2722 005256 004737 005514 JSR PC,SETUP ;GO SET UP
2723 005262 000137 006074 JMP EXEC
2724
2725 ;WRAP AROUND MODE 1 WRITE BUFFER*****
2726
2727 005266 012737 000010 000750 WAM1: MOV #10,WAM
2728 005274 000755 BR WAM01
2729
2730 ;WRAP AROUND MODE 2 BIT FIDDLER WRITE*****
2731
2732 005276 012737 000012 000750 WAM2: MOV #12,WAM
2733 005304 012737 000060 000752 MOV #60,FUN
2734 005312 005037 000754 CLR DATC
2735 005316 012737 017474 000760 MOV #WBUFF,DATAD
2736 005324 012737 020106 000762 MOV #RBUFF,RDAD
2737 005332 004737 005514 WAM2A: JSR PC,SETUP
2738 005336 000137 006074 JMP EXEC
2739
2740 ;WRAP AROUND MODE 3 BIT FIDDLER READ*****
2741
2742 005342 012737 000014 000750 WAM3: MOV #14,WAM ;SET WAM NUMBER
2743 005350 012737 000070 000752 MOV #70,FUN ;SET FUNCTION
2744 005356 012737 020106 000760 MOV #RBUFF,DATAD ;SET BUFFER ADDRESS
2745 005364 012737 017474 000756 MOV #WBUFF,WTAD ;SET POINTER TO WRITE BUFFER
2746 005372 005737 001006 TST RDRVF
2747 005376 001411 BEQ WAM3A
2748 005400 062737 000376 000760 ADD #376,DATAD
2749 005406 062737 000377 000756 ADD #377,WTAD
2750 005414 012737 000076 000752 MOV #76,FUN ;SET READ REVERSE CODE
2751 005422 032737 000020 000776 WAM3A: BIT #20,UDES
2752 005430 001403 BEQ WAM3B
2753 005432 013737 001010 000756 WAM3B: MOV RCDP,WTAD
2754 005440 004737 005514 JSR PC,SETUP ;GO SET UP
2755 005444 000137 006074 JMP EXEC ;GO EXECUTE
2756
2757 ;WRAP AROUND MODE 4 CRC CORRECTION
2758 ;CALL: JSR PC,WAM4
2759
2760 005450 012737 000030 000750 WAM4: MOV #30,WAM ;SET MAINTENANCE MODE FUNCTION WRAP 4
2761 005456 012737 000060 000752 MOV #60,FUN ;SET TAPE FUNCTION = WRITE FWD
2762 005464 005037 000754 CLR DATC
2763 005470 012737 004370 000760 MOV #CRCDAT,DATAD ;SET ADRS OF WRITE BUFFER
2764 005476 012737 020106 000762 MOV #RBUFF,RDAD ;SET READ BUFFER ADDRESS
2765 005504 004737 005514 JSR PC,SETUP ;GO SETUP REGISTERS
2766 005510 000137 006074 JMP EXEC ;GO EXECUTE

```

```

2768 ;REGISTER SETUP ROUTINE*****
2769
2770 005514 005737 001022 SETUP: TST CRCFLG ;DO NOT DO DATA SETUP NOR INIT
2771 005520 001004 BNE 1$ ;IF CRC CORRECTION IS IN PROGRESS
2772 005522 022737 000030 000750 CMP #30,WAM ;DO NOT INIT IF DOING WAM 4 --- STEP D
2773 005530 001004 BNE 2$
2774 005532 012777 000011 172750 1$: MOV #11,@C1 ;DO DRIVE CLEAR IF WAM4---STEP D
2775 005540 000412 BR SET1A
2776 005542 005737 000740 2$: TST STFLG ;SEE IF SINGLE TEST
2777 005546 001403 BEQ SET0 ;IF NOT: BR
2778 005550 005737 001000 TST WPGFL ;SEE IF HAVE SELECTED PATTERN
2779 005554 001002 BNE SET1 ;IF SO: BR
2780 005556 004737 005026 SET0: JSR PC,DSUP ;GO DO DATA SETUP
2781 005562 004737 012206 SET1: JSR PC,INIT ;INIT CONTROLLER,SELECT UNIT & DRIVE
2782 ;LOAD SLAVE DESC AND MOVE OFF BOT
2783 005566 012777 177400 172722 SET1A: MOV #400,@FC ;SET FC=WGX2
2784 005574 032737 000020 000776 BIT #20,UDES ;SEE IF CORE DUMP
2785 005602 001403 BEQ SET2 ;IF NOT: BR
2786 005604 012777 177000 172704 MOV #1000,@FC ;SET FC=WGX4
2787 005612 012777 177600 172672 SET2: MOV #200,@WC ;SET WC
2788 005620 013777 000760 172666 MOV DATAD,@BA ;SET BUS ADDRESS
2789 005626 032777 010000 172664 BIT #10000,@CS ;ASSURE DRIVE THERE
2790 005634 001417 BEQ SP1 ;IF SO: BR
2791 005636 032777 020000 172724 BIT #20000,@SWR ;SEE IF PRINT ERRORS
2792 005644 001004 BNE SP01 ;IF NOT: BR
2793 005646 012704 016635 MOV #WMSG4,R4
2794 005652 004737 013054 JSR PC,TOUT ;PRINT NON-EXISTANT DRIVE
2795 005656 032777 100000 172704 SP01: BIT #100000,@SWR ;SEE IF HALT ON ERROR
2796 005664 001401 BEQ SPO ;IF NOT: BR
2797 005666 000000 HALT
2798 005670 000137 005562 SP0: JMP SET1 ;RESETUP
2799 005674 022737 000014 000750 SP1: CMP #14,WAM ;SEE IF WAM 3
2800 005702 001031 BNE SP1B ;IF NOT: BR
2801 005704 117737 173046 000754 MOVB @WTAD,DATC ;GET FIRST CHAR
2802 005712 042737 177400 000754 BIC #177400,DATC
2803 005720 000337 000754 SWAB DATC
2804 005724 052737 000200 000754 BIS #200,DATC ;SET PARITY
2805 005732 005737 001006 TST RDRVF ;SEE IF READ REVERSE
2806 005736 001403 BEQ SP1A ;IF NOT: BR
2807 005740 005337 000756 DEC WTAD ;DECREMENT POINTER
2808 005744 000410 BR SP1B
2809 005746 005237 000756 SP1A: INC WTAD ;BUMP POINTER
2810 005752 032737 000020 000776 BIT #20,UDES ;SEE IF CORE DUMP
2811 005760 001402 BEQ SP1B ;IF NOT: BR
2812 005762 005237 000756 INC WTAD ;BUMP POINTER AGAIN
2813 005766 053777 000776 172546 SP1B: BIS UDES,@TC ;SET UNIT DESCRIPTION (DEN,PAR,FMT)
2814 005774 052777 000001 172532 BIS #1,@MR ;SET MAINT MODE
2815 006002 053777 000750 172524 BIS WAM,@MR ;SET WAM
2816 006010 053777 000754 172516 BIS DATC,@MR ;SET DATA
2817 006016 013777 000752 172464 MOV FUN,@C1 ;SET FUNCTION
2818 006024 032777 040000 172470 BIT #40000,@DS ;ASSURE NO ERROR
2819 006032 001002 BNE SP3 ;IF ERROR: BR
2820 006034 000240 NOP
2821 006036 000207 RTS PC ;RETURN
2822 006040 032777 020000 172522 SP3: BIT #20000,@SWR ;SEE IF PRINT ERRORS
2823 006046 001004 BNE SP4 ;IF NOT: BR
  
```

2824	006050	012704	016576			MOV	#WMSG2,R4		
2825	006054	004737	013054			JSR	PC,TTOUT		:PRINT SETUP ERROR
2826	006060	032777	100000	172502	SP4:	BIT	#100000,@SWR		:SEE IF HALT ON ERROR
2827	006066	001401				BEQ	SP5		:IF NOT: BR
2828	006070	000000				HALT			
2829	006072	000207			SP5:	RTS	PC		:RETURN

```

2831                                     ;EXECUTE WAM ROUTINE*****
2832
2833 006074 000240 EXEC: NOP
2834 006076 000240 NOP
2835 006100 C32777 000040 172426 BIT #40,@MR
2836 006106 001403 BEQ EX0 ;ASSURE MAINT CLOCK IS ZERO
2837 006110 042777 000040 172416 BIC #40,@MR ;IF NOT: CLEAR IT
2838 006116 022737 000010 000750 EX0: CMP #10,WAM ;SEE IF WAM 1 OR 2 OR 3
2839 006124 003402 BLE 2$
2840 006126 000137 006540 1$: JMP EXW2 ;GO DO WAM 0
2841 006132 022737 000030 000750 2$: CMP #30,WAM ;BRANCH IF WRAP AROUND MODE 4
2842 006140 001772 BEQ 1$
2843 006142 052777 000001 172340 EX1: BIS #1,@C1 ;SET GO BIT
2844 006150 005000 CLR R0
2845 006152 012701 000002 MOV #2,R1 ;SET DELAY
2846 006156 032777 100000 172356 EX1A: BIT #100000,@TC ;SEE IF ALPHA
2847 006164 001404 BEQ EX2 ;IF SO: BR
2848 006166 005300 DEC R0
2849 006170 001372 BNE EX1A ;AWAIT ALPHA
2850 006172 005301 DEC R1
2851 006174 001370 BNE EX1A
2852 006176 005077 172364 EX2: CLR @PSW
2853 006202 012701 000400 MOV #400,R1 ;SET NUMBER OF CLKS
2854 006206 032737 000020 000776 BIT #20,UDES ;SEE IF CORE DUMP
2855 006214 001402 BEQ EX3 ;IF NOT: BR
2856 006216 012701 001000 MOV #1000,R1 ;SET CLOCKS LWCX4
2857 006222 022737 000014 000750 EX3: CMP #14,WAM ;SEE IF WAM 3
2858 006230 001010 BNE 1$ ;IF NOT: BR
2859 006232 032737 002000 000776 BIT #2000,UDES ;IS IT PE?
2860 006240 001016 BNE EX5A ;IF YES: BR
2861 006242 052777 000040 172264 BIS #40,@MR ;CLOCK UP
2862 006250 000412 BR EX5A
2863 006252 032737 002000 000776 1$: BIT #2000,UDES ;SEE IF PE
2864 006260 001404 BEQ EX5 ;IF NOT PE: BR
2865 006262 006301 ASL R1
2866 006264 062701 000246 ADD #246,R1 ;SET TO ALLOW FOR PRE/POSTAMBLE
2867 006270 000402 BR EX5A
2868 006272 062701 000010 EX5: ADD #10,R1 ;ADD CLOCKS FOR CRC AND LRC
2869 006276 022737 000014 000750 EX5A: CMP #14,WAM ;SEE IF WAM 3
2870 006304 001053 BNE EX5C ;IF NOT: BR
2871 006306 117700 172444 MOVB @WTAD,R0
2872 006312 042700 177400 BIC #177400,R0
2873 006316 005737 001006 TST RDRVF ;SEE IF REVERSE
2874 006322 001403 BEQ EX5A1 ;IF NOT: BR
2875 006324 005337 000756 DEC WTAD ;DEC POINTER
2876 006330 000416 BR EX5B
2877 006332 005237 000756 EX5A1: INC WTAD
2878 006336 032737 000020 000776 BIT #20,UDES ;SEE IF CORE DUMP
2879 006344 001410 BEQ EX5B ;IF NOT: BR
2880 006346 005237 000756 INC WTAD ;BUMP POINTER
2881 006352 005777 172400 TST @WTAD ;SEE IF END
2882 006356 001003 BNE EX5B ;IF NOT: BR
2883 006360 162737 000010 000756 SUB #10,WTAD ;RESTORE POINTER
2884 006366 052777 000040 172140 EX5B: BIS #40,@MR ;CLOCK UP
2885 006374 017702 172134 MOV @MR,R2 ;READ MR
2886 006400 042702 177600 BIC #177600,R2 ;MASK OUT DATA
    
```

2887	006404	000300				SWAB	RO	:POSITION DATA
2888	006406	022700	177000			CMP	#177000,RO	:SEE IF PATTERN 3
2889	006412	001404				BEQ	2\$	:IF SO: BR
2890	006414	000240				NOP		
2891	006416	000240				NOP		
2892	006420	052700	000200		1\$:	BIS	#200,RO	:SET ODD PARITY
2893	006424	050002			2\$:	BIS	RO,R2	:LOAD NEW DATA
2894	006426	010277	172102			MOV	R2,@MR	:CLOCK DOWN AND LOAD NEW DATA
2895	006432	000426				BR	EX5D	
2896	006434	052777	000040	172072	EX5C:	BIS	#40,@MR	:CLOCK UP
2897	006442	042777	000040	172064		BIC	#40,@MR	:CLOCK DOWN
2898	006450	017700	172060			MOV	@MR,RO	:GET MR
2899	006454	000300				SWAB	RO	
2900	006456	032737	000010	000776		BIT	#10,UDES	:SEE IF EVEN PAR
2901	006464	001405				BEQ	EX5C0	:IF NOT: BR
2902	006466	010077	172270			MOV	RO,@RDAD	
2903	006472	005237	000762			INC	RDAD	
2904	006476	000402				BR	EX5C1	
2905	006500	110077	172256		EX5C0:	MOVB	RO,@RDAD	:PUT CHAR IN CORE
2906	006504	005237	000762		EX5C1:	INC	RDAD	
2907	006510	000240			EX5D:	NOP		
2908	006512	005301				DEC	R1	:SEE IF DONE CLKS
2909	006514	001270				BNE	EX5A	:IF NOT: BR
2910	006516	032777	000040	172010		BIT	#40,@MR	:CLOCK UP?
2911	006524	001403				BEQ	1\$	:IF NOT: BR
2912	006526	042777	000040	172000		BIC	#40,@MR	:CLOCK DOWN
2913	006534	000137	011712		1\$:	JMP	EORP	:GO DO EOR

```

2915                                     :EXECUTE WAM 0*****
2916
2917 006540 000240 EXW2: NOP
2918 006542 012737 006726 000670 MOV #EXW2H,RTRN ;SET INTERRUPT RETURN ADDRESS
2919 006550 012701 000200 MOV #200,R1 ;SET NUMBER OF CLOCKS = FC/2
2920 006554 032737 002000 000776 BIT #2000,UDES ;SEE IF PE
2921 006562 001402 BEQ EXW2A ;IF NOT: BR
2922 006564 012701 000100 MOV #100,R1 ;ELSE SET CLKS = FC/4
2923 006570 012702 020106 EXW2A: MOV #RBUF,R2 ;SET BUFFER ADDRESS
2924 006574 022737 000030 000750 CMP #30,WAM ;BRANCH IF NOT WRAP 4
2925 006602 001003 BNE 1$
2926 006604 052777 000010 171706 BIS #10,@CS ;SET INHIBIT BUS ADDRESS INCREMENT
2927 006612 012777 000161 171670 1$: MOV #161,@C1 ;SET WRITE COMMAND AND GO
2928 006620 005077 171742 CLR @PSW ;ALLOW INTERRUPT
2929 006624 032777 000040 171702 EXW2B: BIT #40,@MR
2930 006632 001774 BEQ EXW2B ;AWAIT CLOCK UP
2931 006634 017722 171674 MOV @MR,(R2)+ ;GET DATA
2932 006640 032777 000040 171666 EXW2C: BIT #40,@MR
2933 006646 001374 BNE EXW2C ;AWAIT CLOCK DOWN
2934 006650 017722 171660 MOV @MR,(R2)+ ;GET DATA
2935 006654 005301 DEC R1 ;SEE IF DONE ALL
2936 006656 001362 BNE EXW2B ;IF NOT: BR
2937 006660 012701 000003 EXW2E: MOV #3,R1
2938 006664 005000 CLR R0 ;SET DELAY
2939 006666 005300 EXW2F: DEC R0
2940 006670 001376 BNE EXW2F
2941 006672 005301 DEC R1
2942 006674 001374 BNE EXW2F ;DELAY
2943 006676 032777 020000 171664 BIT #20000,@SWR ;SEE IF ERROR PRINT
2944 006704 001004 BNE EXW2G ;IF NOT: BR
2945 006706 012704 017040 MOV #WMSG24,R4
2946 006712 004737 013054 JSR PC,TTOUT ;PRINT NO INTERUPT
2947 006716 005777 171646 EXW2G: TST @SWR ;SEE IF HALT ON ERROR
2948 006722 100001 BPL EXW2H ;IF NOT: BR
2949 006724 000000 HALT
2950 006726 000240 EXW2H: NOP
2951 006730 012701 020106 MOV #RBUF,R1 ;GET START OF READ BUFFER
2952 006734 012700 000400 MOV #400,R0 ;SET SIZE
2953 006740 010102 MOV R1,R2
2954 006742 012203 EXW2J: MOV (R2)+,R3
2955 006744 000303 SWAB R3
2956 006746 032737 000010 000776 BIT #10,UDES ;SEE IF EVEN PAR
2957 006754 001402 BEQ EXW2JO ;IF NOT: BR
2958 006756 010321 MOV R3,(R1)+ ;SAVE PAR + DATA
2959 006760 000401 BR EXW2J1
2960 006762 110321 EXW2JO: MOV R3,(R1)+ ;ASSEMBLE DATA IN BYTES
2961 006764 005300 EXW2J1: DEC R0
2962 006766 001365 BNE EXW2J ;CONTINUE FOR ALL
2963 006770 032777 000200 171572 BIT #200,@SWR ;SEE IF STATUS CHECK
2964 006776 001002 BNE EXW2K ;IF NOT: BR
2965 007000 004737 007026 JSR PC,WSTCK ;ELSE GO CHECK STATUS
2966 007004 000240 EXW2K: NOP
2967 007006 032777 000400 171554 BIT #400,@SWR ;SEE IF DATA CHECK
2968 007014 001002 BNE EXW2X ;IF NOT: BR
2969 007016 004737 007416 JSR PC,DCHK ;ELSE GO CHECK DATA
2970 007022 000240 EXW2X: NOP

```

TM03-TU45 CONTROL LOGIC TEST-PART II MACY11 30A(1052) 13-MAY-80<sup>D 6</sup> 15:09 PAGE 36-1  
CZTUPB.P11 13-MAY-80 15:06

SEG 14

2971 007024 000207

RTS PC :EXIT

```

2973
2974
2975
2976 007026 000240
2977 007030 005037 000772
2978 007034 005037 000620
2979 007040 022737 014774 000622
2980 007046 001404
2981 007050 022737 015664 000622
2982 007056 001025
2983 007060 005737 001000
2984 007064 001405
2985 007066 122737 000003 001000
2986 007074 001016
2987 007076 000404
2988 007100 022737 000003 001002
2989 007106 001011
2990 007110 052737 140000 001024
2991 007116 052737 140000 001030
2992 007124 052737 000100 001032
2993 007132 012737 016662 000672
2994 007140 017702 171344
2995 007144 013705 001024
2996 007150 004737 007272
2997 007154 012737 016707 000672
2998 007162 017702 171332
2999 007166 013705 001026
3000 007172 053705 000624
3001 007176 004737 007272
3002 007202 012737 016715 000672
3003 007210 017702 171306
3004 007214 013705 001030
3005 007220 004737 007272
3006 007224 012737 016722 000672
3007 007232 017702 171266
3008 007236 000240
3009 007240 000240
3010 007242 013705 001032
3011 007246 004737 007272
3012 007252 005737 000772
3013 007256 001456
3014 007260 005777 171304
3015 007264 100053
3016 007266 000000
3017 007270 000451
3018 007272 000240
3019 007274 020205
3020 007276 001446
3021 007300 005237 000772
3022 007304 032777 020000 171256
3023 007312 001040
3024 007314 005737 000620
3025 007320 001010
3026 007322 013704 000622
3027 007326 004737 013054
3028 007332 012704 017024

```

;WRAP AROUND STATUS CHECK ROUTINE\*\*\*\*\*

```

WSTCK: NOP
CLR SERFL ;CLEAR ERROR FLAG
CLR HDRFL ;CLEAR HEADER FLAG
CMP #MSLT2,EMADDR ;SEE IF TEST 2
BEQ 2$ ;IF SO: BR
CMP #MSLT15,EMADDR ;SEE IF TEST 15
BNE 5$ ;IF NOT: BR
2$: TST WPGFL ;SEE IF SINGLE PATTERN
BEQ 3$ ;IF NOT: BR
CMPB #3,WPGFL ;SEE IF PATTERN 3
BNE 5$ ;IF NOT: BR
BR 4$ ;ELSE GO DO EXPT CHANGE
3$: CMP #3,PATRN ;SEE IF PATTERN 3
BNE 5$ ;IF NOT: BR
4$: BIS #140000,WCS1
BIS #140000,WDS
BIS #100,WER ;SET EXPT PARITY ERROR
5$: MOV #WMSG6,ERADD ;SET CODE=CS1
MOV @C1,R2 ;GET RCVD CS1
MOV WCS1,R5 ;GET EXPT CS1
JSR PC,WSTG ;GO CHK
MOV #WMSG6D,ERADD ;SET CODE=CS2
MOV @CS,R2 ;SET RCVD CS2
MOV WCS2,R5 ;GET EXPT CS2
BIS DRVN,R5 ;SET DRIVE NUMBER IN EXPT CS2
JSR PC,WSTG ;GO CHK
MOV #WMSG6E,ERADD ;SET CODE=DS
MOV @DS,R2 ;SET RCVD DS
MOV WDS,R5 ;GET EXPT DS
1$: JSR PC,WSTG ;GO CHK
MOV #WMSG6F,ERADD ;SET CODE=ER
MOV @ER,R2 ;GET RCVD ER
NOP
NOP
MOV WER,R5 ;GET EXPT ER
JSR PC,WSTG ;GO CHK
TST SERFL ;SEE IF ANY ERRORS
BEQ WSTX ;IF NOT: BR
TST @SWR ;SEE IF HALT ON ERROR
BPL WSTX ;IF NOT: BR
HALT
BR WSTX ;CONTINUE
WSTG: NOP
CMP R2,R5 ;SEE IF EXPT=RCVD
BEQ WSTX ;IF SO: BR
I' C SERFL ;SET ERROR FLAG
E. I #20000,@SWR ;SEE IF PRINT ERRORS
BNE WSTX ;IF NOT: BR
TST HDRFL ;SEE IF DONE HEADER
BNE WSTGO ;IF SO: BR
MOV EMADDR,R4
JSR PC,TTOUT ;PRINT TEST HEADER
MOV #WMSG23,R4

```

3029	007336	004737	013054		JSR	PC,TTOUT	;PRINT STATUS TAG
3030	007342	012737	000001	000620	WSTGO:	MOV #1,HDRFL	;SET HEADER FLAG
3031	007350	013704	000672		MOV	ERADD,R4	
3032	007354	004737	013054		JSR	PC,TTOUT	;PRINT CODE
3033	007360	012704	014325		MOV	#MSG12,R4	
3034	007364	004737	013054		JSR	PC,TTOUT	;PRINT EXPT TAG
3035	007370	010503			MOV	R5,R3	
3036	007372	004737	013202		JSR	PC,OCTP	;PRINT EXPT STATUS
3037	007376	012704	014334		MOV	#MSG13,R4	
3038	007402	004737	013054		JSR	PC,TTOUT	;PRINT RCVD TAG
3039	007406	010203			MOV	R2,R3	
3040	007410	004737	013202		JSR	PC,OCTP	;PRINT RCVD STATUS
3041	007414	000207			WSTX:	RTS	;RETURN
3042							

```

3044
3045           ;DATA CHECK ROUTINE*****
3046
3047 007416 000240           DCHK:  NOP
3048 007420 005737 001022   TST     CRCFLG           ;DO NOT DO A DATA CHACK
3049 007424 001402           BEQ     2$              ;IF CRC CORRECTION IN PROGRESS
3050 007426 000137 010256   1$:    JMP     DCHKX1
3051 007432 005737 001020   2$:    TST     DCHKFL           ;BRANCH IF DATA IS NOT TO BE CHECKED
3052 007436 001373           BNE     1$
3053 007440 005037 000620   CLR     HDRFL           ;CLEAR HEADER FLAG
3054 007444 005037 000766   CLR     DERFL           ;CLEAR DATA ERROR FLAG
3055 007450 005037 000774   CLR     CRCNT           ;CLEAR CHAR CNTR
3056 007454 032737 000010 000776   BIT     #10,UDES        ;SEE IF EVEN PARITY
3057 007462 001402           BEQ     DCHKAO          ;IF NOT: BR
3058 007464 000137 010300   JMP     DCHKE           ;ELSE GO CHECK EVEN
3059 007470 022737 000010 000750 DCHKAO: CMP     #10,WAM         ;SEE IF WAM 1
3060 007476 001006           BNE     DCHKA           ;IF NOT: BR
3061 007500 032737 002000 000776   BIT     #2000,UDES      ;SEE IF PE
3062 007506 001402           BEQ     DCHKA           ;IF NOT: BR
3063 007510 000137 011002   JMP     PRCHK           ;GO CHK DATA
3064 007514 012700 177400   DCHKA: MOV     #400,R0    ;SET NUMBER OF CHARACTERS
3065 007520 022737 000012 000750   CMP     #12,WAM
3066 007526 001006           BNE     DCHKA1          ;IF NOT WRAP 2: BR
3067 007530 032737 000020 000776   BIT     #20,UDES
3068 007536 001402           BEQ     DCHKA1          ;IF NOT CORE DUMP: BR
3069 007540 012700 177000   MOV     #1000,R0
3070 007544 022737 000030 000750 DCHKA1: CMP     #30,WAM    ;BRANCH IF WRAP 4
3071 007552 001404           BEQ     1$
3072 007554 022737 000006 000750   CMP     #6,WAM          ;SEE IF WRAP 0
3073 007562 001007           BNE     DCHKA2          ;IF NOT: BR
3074 007564 012700 177744   1$:    MOV     #34,R0      ;SET NUMBER OF CHARACTERS READ
3075 007570 012701 017504   MOV     #WBUFF+10,R1   ;SET POINTER
3076 007574 005037 000766   CLR     DERFL           ;CLEAR DATA ERROR FLAG
3077 007600 000431           BR      DCHKBO
3078 007602 022737 000012 000750 DCHKA2: CMP     #12,WAM    ;SEE IF WRAP 2
3079 007610 001021           BNE     DCHKB           ;IF NOT: BR
3080 007612 032737 002000 000776   BIT     #2000,UDES      ;SEE IF PE
3081 007620 001415           BEQ     DCHKB           ;IF NOT: BR
3082 007622 012700 177653   MOV     #125,R0        ;POINT TO START OF DATA
3083 007626 012737 000001 000764   MOV     #1,W2FLG       ;SET WRAP 2 FLAG
3084 007634 004737 007654   JSR     PC,DCHKB        ;GO CHECK DATA
3085 007640 004737 011302   JSR     PC,W1DCHK       ;GO CHECK WRAP 1 DATA
3086 007644 005037 000764   CLR     W2FLG
3087 007650 000137 010240   JMP     DCHKX
3088 007654 005037 000766   DCHKB: CLR     DERFL
3089 007660 012701 017474   MOV     #WBUFF,R1      ;SET GOOD POINTER
3090 007664 012702 020106   DCHKB0: MOV     #RBUFF,R2 ;SET READ POINTER
3091 007670 032737 000020 000776   BIT     #20,UDES        ;SEE IF CORE DUMP
3092 007676 001416           BEQ     DCHKO           ;IF NOT: BR
3093 007700 022737 000012 000750   CMP     #12,WAM         ;SEE IF WAM 2
3094 007706 001011           BNE     DCHKD           ;IF NOT: BR
3095 007710 005737 001002   TST     PATRN           ;SEE IF PATTERN 0
3096 007714 001003           BNE     DCHKC           ;IF NOT: BR
3097 007716 012701 001056   MOV     #WCDPO,R1      ;SET CORE DUMP PATTERN 0
3098 007722 000404           BR      DCHKO           ;GO CHECK DATA
3099 007724 012701 001044   DCHKC: MOV     #WCDP2,R1 ;SET CORE DUMP WRITE PATTERN 2

```

3100	007730	000401				BR	DCHK0		;GO CHECK DATA
3101	007732	000240				DCHKD: NOP			
3102	007734	121112				DCHK0: CMPB	(R1), (R2)		;SEE IF DATA OK
3103	007736	001466				BEQ	DCHK2		;IF SO: BR
3104	007740	032777	020000	170622		BIT	#20000,@SWR		;SEE IF PRINT ERRORS
3105	007746	001062				BNE	DCHK2		;IF NOT: BR
3106	007750	005737	000620			TST	HDRFL		;SEE IF DONE HEADER
3107	007754	001004				BNE	DCHK1		;IF SO: BR
3108	007756	013704	000622			MOV	EMADDR,R4		
3109	007762	004737	013054			JSR	PC,TTOUT		;PRINT HEADER
3110	007766	005737	000766			DCHK1: TST	DERFL		;SEE IF FIRST ERROR
3111	007772	001014				BNE	DCHK1A		;IF NOT: BR
3112	007774	012704	016772			MOV	#WMSG16,R4		
3113	010000	004737	013054			JSR	PC,TTOUT		;PRINT DATA ERROR TAG
3114	010004	012704	017217			MOV	#WMSG32,R4		
3115	010010	004737	013054			JSR	PC,TTOUT		;PRINT PATRN TAG
3116	010014	013703	001002			MOV	PATRN,R3		
3117	010020	004737	013202			JSR	PC,OCIP		;PRINT PATTERN NUMBER
3118	010024	012737	000001	000620		DCHK1A: MOV	#1,HDRFL		;SET HEADER FLAG
3119	010032	012737	000001	000766		MOV	#1,DERFL		;SET DATA ERROR FLAG
3120	010040	012704	017016			MOV	#WMSG21,R4		
3121	010044	004737	013054			JSR	PC,TTOUT		;PRINT CHARACTER NUMBER TAG
3122	010050	013703	000774			MOV	CRCNT,R3		
3123	010054	004737	013202			JSR	PC,OCIP		;PRINT CHARACTER NUMBER
3124	010060	012704	017004			MOV	#WMSG17,R4		
3125	010064	004737	013054			JSR	PC,TTOUT		;PRINT GOOD TAG
3126	010070	111103				MOVB	(R1),R3		
3127	010072	004737	013430			JSR	PC,DOUT		;PRINT GOOD DATA
3128	010076	012704	017011			MOV	#WMSG20,R4		
3129	010102	004737	013054			JSR	PC,TTOUT		;PRINT BAD TAG
3130	010106	111203				MOVB	(R2),R3		
3131	010110	004737	013430			JSR	PC,DOUT		;PRINT BAD DATA
3132	010114	005737	000764			DCHK2: TST	W2FLG		;SEE IF WRAP 2 NRZ
3133	010120	001020				BNE	DCHK2B		;IF SO: BR
3134	010122	005201				INC	R1		;BUMP POINTER
3135	010124	032737	000020	000776		BIT	#20,UDES		;SEE IF CORE DUMP
3136	010132	001413				BEQ	DCHK2B		;IF NOT: BR
3137	010134	022737	000012	000750		CMP	#12,WAM		;SEE IF WAM 2
3138	010142	001006				BNE	DCHK2A		;IF NOT: BR
3139	010144	005201				INC	R1		;BUMP POINTER
3140	010146	005711				TST	(R1)		;SEE IF END OF PATTERN
3141	010150	001004				BNE	DCHK2B		;IF NOT: BR
3142	010152	162701	000010			SUB	#10,R1		;RESET POINTER TO START OF PATTERN
3143	010156	000401				BR	DCHK2B		;CONTINUE CHECK
3144	010160	000240				DCHK2A: NOP			
3145	010162	005202				DCHK2B: INC	R2		
3146	010164	022737	000030	000750		CMP	#30,WAM		;BRANCH IF WAM 4
3147	010172	001404				BEQ	1\$		
3148	010174	022737	000006	000750		CMP	#6,WAM		;SEE IF WAM 0
3149	010202	001002				BNE	DCHK3		;IF NOT: BR
3150	010204	062701	000010			1\$: ADD	#10,R1		;BUMP POINTER
3151	010210	005237	000774			DCHK3: INC	CRCNT		;BUMP CHAR CNTR
3152	010214	032777	000400	170346		BIT	#400,@SWR		;SEE IF CONT DATA CHK
3153	010222	001006				BNE	DCHKX		;IF NOT: BR
3154	010224	005200				INC	R0		;SEE IF DONE
3155	010226	001242				BNE	DCHK0		;IF NOT: BR

3156	010230	005737	000764	TST	W2FLG	
3157	010234	001401		BEQ	DCHKX	
3158	010236	000207		RTS	PC	
3159	010240	005777	170324	DCHKX: TST	@SWR	;SEE IF HALT CV ERROR
3160	010244	100004		BPL	DCHKX1	;IF NOT: BR
3161	010246	005737	000766	TST	DERFL	;SEE IF DATA EPROR OCCURED
3162	010252	001401		BEQ	DCHKX1	;IF NOT: BR
3163	010254	000000		HALT		
3164	010256	005037	000774	DCHKX1: CLR	CRCNT	;CLEAR CHAR CNTR
3165	010262	005037	000620	CLR	HDRFL	;CLEAR HEADER FLAG
3166	010266	005037	000766	CLR	DERFL	;CLEAR DATA ERROR FLAG
3167	010272	005037	000770	CLR	PREFL	;CLEAR PREAMBLE FLAG
3168	010276	000207		RTS	PC	;RETURN

```

3170
3171
3172
3173 010300 000240
3174 010302 022737 000006 000750
3175 010310 001005
3176 010312 012700 177744
3177 010316 012701 017504
3178 010322 000404
3179 010324 012700 177400
3180 010330 012701 017474
3181 010334 012702 020106
3182 010340 111105
3183 010342 005003
3184 010344 012704 000010
3185 010350 032705 000001
3186 010354 001401
3187 010356 005203
3188 010360 005304
3189 010362 001402
3190 010364 006005
3191 010366 000770
3192 010370 000240
3193 010372 111105
3194 010374 042705 177400
3195 010400 005703
3196 010402 001003
3197 010404 012705 100020
3198 010410 000405
3199 010412 032703 000001
3200 010416 001402
3201 010420 052705 100000
3202 010424 042712 077400
3203 010430 020512
3204 010432 001477
3205 010434 032777 020000 170126
3206 010442 001073
3207 010444 005737 000620
3208 010450 001004
3209 010452 013704 000622
3210 010456 004737 013054
3211 010462 005737 000766
3212 010466 001014
3213 010470 012704 016772
3214 010474 004737 013054
3215 010500 012704 017217
3216 010504 004737 013054
3217 010510 013703 001002
3218 010514 004737 013202
3219 010520 000240
3220 010522 012737 000001 000766
3221 010530 012737 000001 000620
3222 010536 012704 017016
3223 010542 004737 013054
3224 010546 013703 000774
3225 010552 004737 013202

;EVEN PARITY DATA CHECK*****
DCKE:  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
      TSI 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 ;PRINT BAD DATA TAG
      JSR PC,TTOUT
      MOV #WMSG32,R4 ;PRINT PATTERN TAG
      JSR PC,TTOUT
      MOV PATRN,R3 ;PRINT PATTERN NUMBER
      JSR PC,OCIP
DCKE7: NOP
      MOV #1,DERFL ;SET DATA ERROR FLAG
      MOV #1,HDRFL ;SET HEADER FLAG
      MOV #WMSG21,R4 ;PRINT CHAR NUMBER TAG
      JSR PC,TTOUT
      MOV CRCNT,R3 ;ORINT CHAR NUMBER
      JSR PC,OCIP

```

3226	010556	012704	017004		MOV	#WMSG17,R4	
3227	010562	004737	01304		JSR	PC,TTOUT	;PRINT GOOD DATA TAG
3228	010566	110503			MOVB	R5,R3	
3229	010570	004737	013430		JSR	PC,DOUT	;PRINT EXPT DATA
3230	010574	010503			MOV	R5,R3	
3231	010576	004737	010706		JSR	PC,DCKEP	;GO PRINT PARITY BIT
3232	010602	000240			NOP		
3233	010604	012704	017011		MOV	#WMSG20,R4	
3234	010610	004737	013054		JSR	PC,TTOUT	;PRINT BAD TAG
3235	010614	111203			MOVB	(R2),R3	
3236	010616	004737	013430		JSR	PC,DOUT	;PRINT BAD DATA
3237	010622	011203			MOV	(R2),R3	
3238	010624	004737	010706		JSR	PC,DCKEP	;GO PRINT PARITY BIT
3239	010630	000240			NOP		
3240	010632	005201			DCKE10: INC	R1	
3241	010634	022737	000006	000750	CMR	#6,WAM	;SEE IF WRAP 0
3242	010642	001002			BNE	1\$	;IF NOT: BR
3243	010644	062701	000010		ADD	#10,R1	;BUMP POINTER
3244	010650	005722			1\$: TST	(R2)+	;BUMP POINTERS
3245	010652	005237	000774		INC	CRCNT	;BUMP CHAR CNTR
3246	010656	032777	000400	167704	BIT	#400,@SWR	;SEE IF CONTINUE DATA CHECK
3247	010664	001402			BEQ	DCKE11	;IF SO: BR
3248	010666	000137	010240		JMP	DCHKX	;GO TO END OF DATA CHECK
3249	010672	005200			DCKE11: INC	R0	;SEE IF DONE
3250	010674	001402			BEQ	DCKE12	;IF SO: BR
3251	010676	000137	010340		JMP	DCKE0	;ELSE CONTINUE
3252	010702	000137	010240		DCKE12: JMP	DCHKX	;GO TO END OF DATA CHECK
3253	010706	000240			DCKEP: NOP		
3254	010710	012737	000240	000614	MOV	#240,TOB	
3255	010716	004737	013154		JSR	PC,TOG	;SPACE
3256	010722	012737	000260	000614	MOV	#260,TOB	;SET PAR=0
3257	010730	005703			TST	R3	;SEE IF PARITY REALLY=0
3258	010732	100002			BPL	DCKEPO	;IF SO: BR
3259	010734	005237	000614		INC	TOB	;ELSE SET TO 1
3260	010740	004737	013154		DCKEPO: JSR	PC,TOG	;PRINT PARITY BIT
3261	010744	000207			RTS	PC	;RETURN
3262							

```

3264
3265                                     ;PREAMBLE/POSTAMBLE CHECK*****
3266
3267 010746 012700 000051      PSCHK: MOV    #51,R0          ;SET SIZE OF POSTAMBLE
3268 010752 012701 017352      MOV    #POST,R1         ;SET POINTER TO POSTAMBLE
3269 010756 005037 000620      CLR    HDRFL           ;CLEAR HEADER FLAG
3270 010762 005037 000774      CLR    CRCNT          ;CLEAR CHAR CNTR
3271 010766 005037 000766      CLR    DERFL          ;CLEAR DATA ERROR FLAG
3272 010772 000240
3273 010774 000240      NOP
3274 010776 000137 011020      JMP    PDO            ;GO CHECK POSTAMBLE
3275
3276 011002 012700 000051      PRCHK: MOV    #51,R0          ;SET SIZE OF PREAMBLE
3277 011006 012701 017230      MOV    #PRE,R1         ;SET POINTER TO PREAMBLE
3278 011012 012702 020106      MOV    #RBUFF,R2       ;SET POINTER TO START OF READ BUFFER
3279 011016 022122              CMP    (R1)+,(R2)+     ;BUMP ADDRESS POINTERS
3280 011020 121112              PDO:  CMPB   (R1),(R2)   ;CHECK DATA
3281 011022 001004              BNE    PD1            ;IF NOT GOOD: BR
3282 011024 126162 000001 000001  CMPB   1(R1),1(R2)     ;COMPARE COMPLIMENT BYTE
3283 011032 001477              BEQ    PD5            ;IF GOOD: BR
3284 011034 032777 020000 167526 PD1:  BIT    #20000,@SWR    ;SEE IF PRINT INHIBIT
3285 011042 001073              BNE    PD5            ;IF SO: BR
3286 011044 005737 000620      TST    HDRFL          ;SEE IF DONE HEADER
3287 011050 001020              BNE    PD4            ;IF SO: BR
3288 011052 013704 000622      MOV    EMADDR,R4
3289 011056 004737 013054      JSR    PC,TTOUT        ;PRINT TEST HEADER
3290 011062 005737 000770      TST    PREFL          ;SEE IF PREAMBLE CHECK
3291 011066 001403              BEQ    PD2            ;IF NOT: BR
3292 011070 012704 017143      MOV    #WMSG29,R4     ;SET POSTAMBLE HEADER
3293 011074 000402              BR     PD3
3294 011076 012704 017125      PD2:  MOV    #WMSG28,R4 ;SET PREAMBLE HEADER
3295 011102 004737 013054      PD3:  JSR    PC,TTOUT   ;PRINT HEADER
3296 011106 005237 000620      INC    HDRFL
3297 011112 012704 017016      PD4:  MOV    #WMSG21,R4
3298 011116 004737 013054      JSR    PC,TTOUT        ;PRINT CHAR NUMBER TAG
3299 011122 013703 000774      MOV    CRCNT,R3
3300 011126 004737 013202      JSR    PC,OCIP         ;PRINT CHAR NUMBER
3301 011132 012704 017004      MOV    #WMSG17,R4
3302 011136 004737 013054      JSR    PC,TTOUT        ;PRINT GOOD TAG
3303 011142 116103 000001      MOVB   1(R1),R3
3304 011146 004737 013430      JSR    PC,DOUT         ;PRINT GOOD CHAR
3305 011152 012737 000240 000614  MOV    #240,TOB
3306 011160 004737 013154      JSR    PC,TOG
3307 011164 111103      MOVB   (R1),R3
3308 011166 004737 013430      JSR    PC,DOUT         ;PRINT COMPLEMENT
3309 011172 012704 017011      MOV    #WMSG20,R4
3310 011176 004737 013054      JSR    PC,TTOUT        ;PRINT BAD TAG
3311 011202 116203 000001      MOVB   1(R2),R3
3312 011206 004737 013430      JSR    PC,DOUT         ;PRINT BAD CHAR
3313 011212 012737 000240 000614  MOV    #240,TOB
3314 011220 004737 013154      JSR    PC,TOG
3315 011224 111203      MOVB   (R2),R3
3316 011226 004737 013430      JSR    PC,DOUT         ;PRINT COMPLEMENT
3317 011232 022122      PD5:  CMP    (R1)+,(R2)+   ;BUMP ADDRESS POINTERS
3318 011234 005237 000774      INC    CRCNT          ;BUMP CHAR NUMBER
3319 011240 005300      DEC    R0             ;SEE IF DONE

```

3320 011242 001266  
3321 011244 005737 000770  
3322 011250 001402  
3323 011252 000137 010240  
3324 011256 005237 000770  
3325 011262 005037 000620  
3326 011266 005037 000774  
3327 011272 005037 000766  
3328 011276 000137 011302  
3329

PD6:

BNE	PDO	;IF NOT: BR
TST	PREFL	;SEE IF PREAMBLE
BEQ	PD6	;IF SO: BR
JMP	DCHKX	;GO TO EXIT ROUTINE
INC	PREFL	;SET PREAMBLE FLAG
CLR	HDRFL	;CLEAR HEADER FLAG
CLR	CRCNT	;CLEAR CHAR CNTR
CLR	DERFL	;CLEAR DATA ERROR FLAG
JMP	WIDCHK	;GO CHECK WRAP 1 DATA

```

3331
3332
3333 ;WAM 1 PE DATA CHECK*****
3334 011302 012700 177400 W1DCHK: MOV #400,R0 ;SET NUMBER OF CHAR TO CHECK
3335 011306 012701 017474 MOV #WBUFF,R1 ;SET WRITE DATA POINTER
3336 011312 012702 020106 MOV #RBUFF,R2 ;SET READ DATA POINTER
3337 011316 062702 000124 ADD #124,R2 ;POINT TO START OF DATA
3338 011322 005737 000764 TST W2FLG ;SEE IF WRAP 2
3339 011326 001401 BEQ W1D0 ;IF NOT WAM 2: BR
3340 011330 005302 DEC R2 ;RESET POINTER
3341 011332 111105 W1D0: MOVB (R1),R5
3342 011334 120512 CMPB R5,(R2) ;CHECK DATA
3343 011336 001007 BNE W1D1 ;IF NOT GOOD:BR
3344 011340 005737 000764 TST W2FLG ;SEE IF WRAP 2
3345 011344 001001 BNE W1D0A ;IF SO: BR
3346 011346 105105 COMB R5 ;COMPLIMENT EXPT DATA
3347 011350 120562 000001 W1D0A: CMPB R5,1(R2) ;CHECK COMPLIMENT DATA
3348 011354 001510 BEQ W1D3 ;IF GOOD: BR
3349 011356 032777 020000 167204 W1D1: BIT #20000,@SWR ;SEE IF PRINT INHIBIT
3350 011364 001104 BNE W1D3 ;IF SO BR
3351 011366 005737 000620 TST HDRFL ;SEE IF DONE HEADER
3352 011372 001020 BNE W1D2 ;IF SO: BR
3353 011374 013704 000622 MOV EMADDR,R4
3354 011400 004737 013054 JSR PC,TTOUT ;PRINT TEST HEADER
3355 011404 012704 016772 MOV #WMSG16,R4
3356 011410 004737 013054 JSR PC,TTOUT ;PRINT BAD DATA TAG
3357 011414 012704 017217 MOV #WMSG32,R4
3358 011420 004737 013054 JSR PC,TTOUT ;PRINT PATRN TAG
3359 011424 013703 001002 MOV PATRN,R3
3360 011430 004737 013202 JSR PC,OC1P ;PRINT PATTERN NUMBER
3361 011434 012737 000001 000620 W1D2: MOV #1,HDRFL ;SET HEADER FLAG
3362 011442 012704 017016 MOV #WMSG21,R4
3363 011446 004737 013054 JSR PC,TTOUT ;PRINT CHAR NUMBER TAG
3364 011452 013703 000774 MOV CRCNT,R3
3365 011456 004737 013202 JSR PC,OC1P ;PRINT CHAR NUMBER
3366 011462 012704 017004 MOV #WMSG17,R4
3367 011466 004737 013054 JSR PC,TTOUT ;PRNT GOOD TAG
3368 011472 111105 MOVB (R1),R5
3369 011474 110503 MOVB R5,R3 ;GET GOOD CHAR
3370 011476 005737 000764 TST W2FLG ;SEE IF WRAP 2
3371 011502 001001 BNE W1D2A ;IF SO: BR
3372 011504 105103 COMB R3 ;ELSE COMPLIMENT CHAR
3373 011506 004737 013430 W1D2A: JSR PC,DOUT ;PRINT CHARACTER
3374 011512 012737 000240 000614 MOV #240,TOB
3375 011520 004737 013154 JSR PC,TOG ;SPACE
3376 011524 110503 MOVB R5,R3
3377 011526 004737 013430 JSR PC,DOUT ;PRINT CHAR
3378 011532 012704 017011 MOV #WMSG20,R4
3379 011536 004737 013054 JSR PC,TTOUT ;PRINT BAD TAG
3380 011542 116203 000001 MOVB 1(R2),R3
3381 011546 004737 013430 JSR PC,DOUT ;PRINT BAD CHAR
3382 011552 012737 000240 000614 MOV #240,TOB
3383 011560 004737 013154 JSR PC,TOG ;SPACE
3384 011564 111203 MOVB (R2),R3
3385 011566 004737 013430 JSR PC,DOUT ;PRINT CHAR
3386 011572 005237 000766 INC DERFL ;SET DATA ERROR FLAG
  
```

```

3387 011576 122122
3388 011600 105722
3389 011602 005237 000774
3390 011606 000406
3391 011610 005737 000764
3392 011614 001401
3393 011616 000207
3394 011620 000137 010746
3395 011624 005200
3396 011626 001770
3397 011630 000137 011332
3398
3399
3400
3401 011634 000240
3402 011636 012700 000050
3403 011642 012701 017230
3404 011646 005721
3405 011650 012721 177400
3406 011654 005300
3407 011656 001374
3408 011660 012701 017352
3409 011664 012700 000050
3410 011670 012721 000377
3411 011674 012721 177400
3412 011700 005300
3413 011702 001374
3414 011704 000207
  
```

```

W1D3:  CMPB  (R1)+,(R2)+ ;BUMP ADDRESS
        TSTB  (R2)+ ;BUMP ADDRESS
        INC   CRCNT ;BUMP CHAR CNTR
        BR    W1D5
W1D4:  TST   W2FLG ;SEE IF WRAP 2
        BEQ  W1D4A ;IF NOT: BR
        RTS  PC ;ELSE RETURN
W1D4A: JMP   PSCHK ;GO CHECK POSTAMBLE
W1D5:  INC   RO
        BEQ  W1D4
        JMP  W1D0

;PREAMBLE/POSTAMBLE GENERATE SUBROUTINE*****

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

```

3416
3417                               ;END OF RECORD FORCE SUBROUTINE*****
3418
3419 011706 005237 000676      EORPA: INC      TEMP2      ;SET WRAP FLAG
3420 011712 017700 166616      EORP: MOV      @MR,R0      ;GET MAINT REG
3421 011716 042700 000036      BIC      #36,R0      ;CLEAR CURRENT OP CODE
3422 011722 052700 000024      BIS      #24,R0      ;SET EOR CLEAR OP CODE
3423 011726 010077 166602      MOV      R0,@MR      ;DO EOR
3424 011732 042777 000037 166574 BIC      #37,@MR      ;CLEAR EOR AND MM
3425 011740 005000
3426 011742 012701 000002      MOV      #2,R1
3427 011746 032777 000001 166534 EORP1: BIT      #1,@C1      ;SEE IF GO GONE
3428 011754 001427      BEQ      EORP2      ;IF SO: BR
3429 011756 005300      DEC      R0
3430 011760 001372      BNE      EORP1      ;AWAIT GO RESET
3431 011762 005301      DEC      R1
3432 011764 001370      BNE      EORP1
3433 011766 032777 020000 166574 BIT      #20000,@SWR      ;SEE IF ERROR PRINT INHIBIT
3434 011774 001017      BNE      EORP2      ;IF SO: BR
3435 011776 005737 000620      TST      HDRFL      ;SEE IF DONE HEADER
3436 012002 001004      BNE      EORP1A      ;IF SO: BR
3437 012004 013704 000622      MOV      EMADDR,R4
3438 012010 004737 013054      JSR      PC,TTOUT      ;PRINT HEADER
3439 012014 012704 017162      EORP1A: MOV     #WMSG31,R4
3440 012020 004737 013054      JSR      PC,TTOUT      ;PRINT EOR GO BIT ERROR
3441 012024 005777 166540      TST      @SWR      ;SEE IF HALT ON ERROR
3442 012030 100001      BPL      EORP2      ;IF NOT: BR
3443 012032 000000      HALT
3444 012034 000240      EORP2: NOP
3445 012036 005737 000676      TST      TEMP2      ;SEE IF WAM
3446 012042 001015      BNE      EORPX      ;IF NOT: BR
3447 012044 032777 000200 166516 BIT      #200,@SWR      ;SEE IF STATUS CHECK
3448 012052 001002      BNE      EORP3      ;IF NOT: BR
3449 012054 004737 007026      JSR      PC,WSTCK      ;ELSE GO CHECK STATUS
3450 012060 000240      EORP3: NOP
3451 012062 032777 000400 166500 BIT      #400,@SWR      ;SEE IF DATA CHECK
3452 012070 001002      BNE      EORPX      ;IF NOT: BR
3453 012072 004737 007416      JSR      PC,DCHK      ;ELSE GO CHECK DATA
3454 012076 000240      EORPX: NOP
3455 012100 005037 000676      CLR      TEMP2      ;CLEAR FLAG
3456 012104 000207      RTS      PC          ;RETURN
3457

```

```

3459
3460           ;SCOPE LOOP ON ERROR SUBROUTINE*****
3461
3462 012106 000240 SCOPE: NOP
3463 012110 032777 040000 166452 BIT #40000,@SWR ;SEE IF LOOP ON ERROR
3464 012116 001001 BNE 1$ ;IF SO: BR
3465 012120 000207 RTS PC ;ELSE EXIT
3466 012122 000240 1$: NOP
3467 012124 005726 TST (SP)+ ;RESET STACK
3468 012126 000240 NOP
3469 012130 000240 NOP
3470 012132 000177 166554 JMP @SCOLP ;LOOP ON ERROR
3471
3472           ;TEST ITERATION SUBROUTINE*****
3473
3474 012136 032777 004000 166424 ITER: BIT #4000,@SWR ;SEE IF ITERATIONS
3475 012144 001403 BEQ 2$ ;IF SO: BR
3476 012146 005037 000702 1$: CLR ITCNT ;CLEAR ITERATION COUNTER
3477 012152 000207 RTS PC ;ELSE EXIT
3478 012154 005737 001016 2$: TST @#PCNTR ;DO SINGLE SUBTEST ITERATION
3479 012160 001772 BEQ 1$ ;ON FIRST PASS
3480 012162 005237 000702 INC ITCNT ;BUMP COUNTER
3481 012166 023737 000702 000606 CMP ITCNT,ITAMT ;SEE IF DONE ALL
3482 012174 001764 BEQ 1$ ;IF SO: BR
3483 012176 005726 TST (SP)+ ;RESET STACK
3484 012200 017700 166510 MOV @ITRLP,R0 ;SET ITERATION POINTER
3485 012204 000110 JMP (R0) ;GO ITERATE
3486
  
```

```

3488
3489
3490
3491 012206 012777 000040 166304 INIT: MOV #40,@CS ;INIT
3492 012214 013777 000624 166276 MOV DRVN,@CS ;SELECT DRIVE
3493 012222 013777 000664 166312 MOV SLVN,@TC ;SFLECT SLAVE
3494 012230 013746 000776 MOV UDES,-(SP) ;GET TEST'S UNIT DESCRIPTION
3495 012234 042716 174377 BIC #174377,(SP) ;CLEAR ALL BUT DENSITY SELECT BITS
3496 012240 022726 001400 CMP #1400,(SP)+ ;BRANCH IF NOT NRZ (800 BPI)
3497 012244 001005 BNE 1$
3498 012246 032777 000040 166246 BIT #40,@DS ;BRANCH IF SLAVE IS IN NRZ MODE
3499 012254 001420 BEQ 4$ ;(PES = 0)
3500 012256 000404 BR 2$ ;GO CHANGE DENSITY
3501 012260 032777 000040 166234 1$: BIT #40,@DS ;BRANCH IF SLAVE IS IN PE MODE
3502 012266 001013 BNE 4$ ;(PES = 1)
3503 012270 012777 000007 166212 2$: MOV #7,@C1 ;REWIND SLAVE
3504 012276 032777 000200 166216 20$: BIT #200,@DS ;WAIT FOR READY
3505 012304 001774 BEQ 20$
3506 012306 032777 020000 166206 3$: BIT #20000,@DS ;LOOP UNTIL REWIND IS COMPLETE
3507 012314 001374 BNE 3$ ;(PIP = 0)
3508 012316 053777 000776 166216 4$: BIS UDES,@TC ;LOAD UNIT DESCRIPTION
3509 012324 032777 000002 166170 BIT #2,@DS ;BRANCH IF NOT AT BOT
3510 012332 001407 BEQ 6$
3511 012334 012777 000025 166146 MOV #25,@C1 ;ERASE TO GET OFF BOT
3512 012342 032777 000200 166152 5$: BIT #200,@DS ;LOOP UNTIL DONE
3513 012350 001774 BEQ 5$
3514 012352 012777 000011 166130 6$: MOV #11,@C1 ;DO A DRIVE CLFAR
3515 012360 000207 RTS PC ;RETURN TO CALLER
3516
3517
3518
3519 012362 000240 MTINT: NOP
3520 012364 013716 000670 MOV RTRN,(SP) ;SET RETURN TO (RTRN)
3521 012370 000002 RTI ;RETURN
3522
3523
3524
3525 012372 017746 166176 TTINT: MOV @TKB,-(SP) ;GET CHARACTER
3526 012376 042716 000200 BIC #200,(SP) ;CLEAR PARITY BIT
3527 012402 122716 000003 CMPB #3,(SP) ;BRANCH IF NOT CONTROL C
3528 012406 001006 BNE 1$
3529 012410 005737 001262 TST CHNFLG ;INHIBIT ^C IF CHAIN MODE
3530 012414 001003 BNE 1$
3531 012416 000005 RESET
3532 012420 000137 000200 JMP @#200 ;RESTART PROGRAM
3533 012424 122716 000001 1$: CMPB #1,(SP) ;BRANCH IF NOT ^A
3534 012430 001017 BNE 2$
3535 012432 022737 000176 000570 CMP #SWREG,SWR ;BRANCH IF HARDWARE SWR INVOKED
3536 012440 001016 BNE 3$
3537 012442 012737 177570 000570 MOV #177570,SWR ;INVOKE HARDWARE SWR
3538 012450 004737 013732 JSR PC,.SAVE ;SAVE REGISTERS ON THE STACK
3539 012454 012704 014641 MOV #MSG63,R4 ;TYPE 'HARDWARE SWR IN USE'
3540 012460 004737 013054 JSR PC,TTOUT
3541 012464 004737 013754 JSR PC,.RESTORE ;RESTORE REGISTERS
3542 012470 022716 000007 2$: (MP #7,(SP) ;BRANCH IF NOT ^G
3543 012474 001005 BNE 4$

```

```
3544 012476 012737 000176 000570 3$: MOV #SWREG,SWR ;INVOKE SOFTWARE SWR
3545 012504 004737 013634 JSR PC,GTSWR ;GET SOFTWARE SWITCHES
3546 012510 005726 TST (SP)+ ;POP CHARACTER OFF THE STACK
3547 012512 000002 RTI ;RETURN
3548
```

```

3550
3551
3552
3553
3554
3555
3556
3557
3558
3559
3560
3561
3562
3563
3564
3565
3566
3567 012514 010146
3568 012516 011601
3569 012520 005037 000674
3570 012524 005000
3571 012526 004737 012774
3572 012532 122737 000003 000616
3573 012540 001003
3574 012542 000005
3575 012544 000137 000200
3576 012550 122737 000015 000616
3577 012556 001004
3578 012560 005737 000674
3579 012564 001471
3580 012566 000457
3581 012570 122737 000025 000616
3582 012576 001005
3583 012600 012704 014567
3584 012604 004737 013054
3585 012610 000742
3586 012612 122737 000177 000616
3587 012620 001012
3588 012622 000241
3589 012624 006000
3590 012626 006200
3591 012630 006200
3592 012632 012704 014571
3593 012636 004737 013054
3594 012642 005201
3595 012644 000730
3596 012646 122737 000060 000616
3597 012654 101402
3598 012656 000137 012754
3599 012662 122737 000070 000616
3600 012670 101002
3601 012672 000137 012754
3602 012676 005237 000674
3603 012702 006300
3604 012704 006300
3605 012706 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$
3586: 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
3596: 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
ASL R0 ;SHIFT 3 LEFT

```

3606	012710	042737	177770	000616		BIC	#177770,T1B	;STRIP ASCII
3607	012716	053700	000616			BIS	T1B,R0	;LOAD CHARACTER
3608	012722	005301				DEC	R1	;SEE IF DONE
3609	012724	001300				BNE	1\$	;IF NOT: BR
3610	012726	020002			6\$:	CMP	R0,R2	;SEE IF EXCEEDED MAXIMUM LIMIT
3611	012730	101402				BLOS	7\$	;IF NOT: BR
3612	012732	000137	012754			JMP	T1NER	;ELSE GO TO ERROR
3613	012736	020300			7\$:	CMP	R3,R0	;SEE IF BELOW MINIMUM LIMIT
3614	012740	101402				BLOS	8\$	;IF NOT: BR
3615	012742	000137	012754			JMP	T1NER	;ELSE GO TO ERROR
3616	012746	010015			8\$:	MOV	R0,(R5)	;LOAD VALUE
3617	012750	005726			9\$:	TST	(SP)+	;POP CHAR COUNT OFF STACK
3618	012752	000207				RTS	PC	;EXIT

```

3620
3621           ;TTY ENTRY ERROR SUBROUTINE*****
3622
3623 012754 012704 014444  T1NER: MOV   #MSG40,R4
3624 012760 004737 013054  JSR   PC,TTOUT      ;PRINT?
3625 012764 005726          TST   (SP)+          ;POP CHAR COUNT OFF STACK
3626 012766 162716 000020  SUB   #20,(SP)      ;RESET SP TO START OF VALUE ROUTINE
3627 012772 000207          RTS    PC            ;REDO VALUE ENTRY
3628
3629           ;TTY READ SUBROUTINE*****
3630
3631 012774 017746 165566  TTIN: MOV   @PSW, -(SP) ;SAVE CURRENT PSW
3632 013000 052777 000340 165560  BIS   #340, @PSW    ;SET TO BR7 TO PREVENT INTRPT
3633 013006 005277 165560          INC   @TKS
3634 013012 105777 165554  1$:  TSTB  @TKS
3635 013016 100375          BPL   1$
3636 013020 012677 165542  MOV   (SP)+, @PSW   ;RESTORE PSW ,OK TO INTRPT NOW
3637 013024 017737 165544 000616  MOV   @TKB,TIB
3638 013032 042737 000200 000616  BIC   #200,TIB
3639 013040 013737 000616 000614  MOV   TIB,TOB      ;MOVE CHAR TO TTY OUPUT BFR
3640 013046 004737 013154  JSR   PC,TOG       ;ECHO CHARACTER
3641 013052 000207          RTS    PC
3642
3643           ;TTY OUTPUT SUBROUTINE*****
3644
3645 013054 112437 000614  TTOUT: MOVB  (R4)+,TOB
3646 013060 122737 000043 000614  CMPB  #43,TOB
3647 013066 001440          BEQ   TEX
3648 013070 122737 000045 000614  CMPB  #45,TOB
3649 013076 001403          BEQ   1$
3650 013100 004737 013154  JSR   PC,TOG
3651 013104 000763          BR    TTOUT
3652 013106 112737 000015 000614  1$:  MOVB  #15,TOB
3653 013114 004737 013154  JSR   PC,TOG
3654 013120 012703 000004          MOV   #4,R3
3655 013124 005037 000614  2$:  CLR   TOB
3656 013130 004737 013154  JSR   PC,TOG
3657 013134 005303          DEC   R3
3658 013136 001372          BNE   2$           ;DO FILLERS
3659 013140 112737 000012 000614  MOVB  #12,TOB
3660 013146 004737 013154  JSR   PC,TOG
3661 013152 000740          BR    TTOUT
3662 013154 105777 165416  TOG:  TSTB  @TPS
3663 013160 100375          BPL   TOG
3664 013162 113777 000614 165410  MOVB  TOB,@TPB
3665 013170 000207          TEX:  RTS    PC
3666
3667           ;OCTAL OUTPUT SUBROUTINE*****
3668
3669
3670 013172 012737 000001 013426  OCTPE: MOV   #1,OFL
3671 013200 000402          BR    OCTPE1
3672 013202 005037 013426  OCTP:  CLR   OFL      ;CLEAR FLAG FOR LEADING ZERO
3673 013206 010304          OCTPE1: MOV  R3,R4    ;SEE IF NUMBER IS ZERO
3674 013210 001007          BNE   OCTPO       ;IF NOT ZERO: BR
3675 013212 005737 013426          TST   OFL        ;SEE IF PRINT ALL 0

```

3676	013216	001004			BNE	OCTP0		;IF SO: BR
3677	013220	004737	013406		JSR	PC,OCTPG1		;ELSE PRINT ZERO
3678	013224	000137	013350		JMP	OCTP3		;SPACE AND EXIT
3679	013230	032704	100000		OCTP0:	BIT	#100000,R4	;SEE IF MSD = 1
3680	013234	001406				BEQ	OCTP1	;IF NOT: BR
3681	013236	012704	000001		MOV	#1,R4		
3682	013242	004737	013364		JSR	PC,OCTPG		;PRINT 1
3683	013246	000137	013260		JMP	OCTP2		
3684	013252	005004			OCTP1:	CLR	R4	
3685	013254	004737	013364		JSR	PC,OCTPG		;PRINT 0
3686	013260	010304			OCTP2:	MOV	R3,R4	
3687	013262	006004				ROR	R4	
3688	013264	006004				ROR	R4	
3689	013266	006004				ROR	R4	;POSITION DIGIT
3690	013270	006004				ROR	R4	
3691	013272	000304				SWAB	R4	
3692	013274	004737	013364		JSR	PC,OCTPG		;PRINT DIGIT 2
3693	013300	010304			MOV	R3,R4		
3694	013302	006004			ROR	R4		
3695	013304	000304			SWAB	R4		
3696	013306	004737	013364		JSR	PC,OCTPG		;PRINT DIGIT 3
3697	013312	010304			MOV	R3,R4		
3698	013314	006104			ROL	R4		
3699	013316	006104			ROL	R4		
3700	013320	000304			SWAB	R4		
3701	013322	004737	013364		JSR	PC,OCTPG		;PRINT DIGIT 4
3702	013326	010304			MOV	R3,R4		
3703	013330	006004			ROR	R4		
3704	013332	006004			ROR	R4		
3705	013334	006004			ROR	R4		
3706	013336	004737	013364		JSR	PC,OCTPG		
3707	013342	010304			MOV	R3,R4		
3708	013344	004737	013364		JSR	PC,OCTPG		;PRINT DIGIT 5
3709	013350	012737	000240	000614	OCTP3:	MOV	#240,TOB	
3710	013356	004737	013154		JSR	PC,TOG		;PRINT SPACE
3711	013362	000207			RTS	PC		;EXIT
3712	013364	042704	177770		OCTPG:	BIC	#177770,R4	
3713	013370	001004				BNE	OCTPG0	
3714	013372	005737	013426			TST	OFL	
3715	013376	001001				BNE	OCTPG0	
3716	013400	000207				RTS	PC	
3717								
3718	013402	005237	013426		OCTPG0:	INC	OFL	
3719	013406	052704	000260		OCTPG1:	BIS	#260,R4	
3720	013412	010437	000614			MOV	R4,TOB	
3721	013416	004737	013154			JSR	PC,TOG	
3722	013422	010304				MOV	R3,R4	
3723	013424	000207				RTS	PC	
3724	013426	000000			OFL:	0		;FIRST CHAR FLAG
3725								

```

3727
3728 ;DATA CHARACTER OUTPUT SUBROUTINE*****
3729
3730 013430 012704 000010 DOUT: MOV #10,R4 ;SET NUMBER TO PRINT
3731 013434 110337 000614 MOVB R3,TOB
3732 013440 105777 165132 1$: TSTB @TPS
3733 013444 100375 BPL 1$
3734 013446 132737 000200 000614 BITB #200,TOB
3735 013454 001404 BEQ 2$
3736 013456 012777 000061 165114 MOV #061,@TPB
3737 013464 000403 BR 3$
3738 013466 012777 000060 165104 2$: MOV #060,@TPB
3739 013474 006337 000614 3$: ASL TOB
3740 013500 005304 DEC R4
3741 013502 001356 BNE 1$
3742 013504 000207 RTS PC
3743
3744 013506 013703 000700 DOUTD: MOV TEMP3,R3
3745 013512 000303 SWAB R3
3746 013514 004737 013430 JSR PC,DOUT
3747 013520 013703 000700 MOV TEMP3,R3
3748 013524 004737 013430 JSR PC,DOUT
3749 013530 000207 RTS PC
3750
3751 ;TU45 SERIAL NUMBER PRINT SUBROUTINE*****
3752
3753 013532 010304 SNPT: MOV R3,R4
3754 013534 000304 SWAB R4
3755 013536 006004 ROR R4
3756 013540 006004 ROR R4
3757 013542 006004 ROR R4
3758 013544 006004 ROR R4 ;GET FIRST DIGIT
3759 013546 004737 013610 JSR PC,SNPG ;PRINT
3760 013552 010304 MOV R3,R4
3761 013554 000304 SWAB R4 ;GET SECOND DIGIT
3762 013556 004737 013610 JSR PC,SNPG ;PRINT
3763 013562 010304 MOV R3,R4
3764 013564 006004 ROR R4
3765 013566 006004 ROR R4
3766 013570 006004 ROR R4
3767 013572 006004 ROR R4
3768 013574 004737 013610 JSR PC,SNPG ;PRINT THIRD DIGIT
3769 013600 010304 MOV R3,R4
3770 013602 004737 013610 JSR PC,SNPG ;PRINT FOURTH DIGIT
3771 013606 000207 RTS PC ;EXIT
3772 013610 012737 000260 000614 SNPG: MOV #260,TOB ;SET BASE = 0
3773 013616 042704 177760 BIC #177760,R4 ;MASK DIGIT
3774 013622 050437 000614 BIS R4,TOB ;SET ASCII
3775 013626 004737 013154 JSR PC,TOG ;TYPE DIGIT
3776 013632 000207 RTS PC ;RETURN
  
```

```

3778
3779      ;THIS ROUTINE GETS THE NEW VALUE FOR THE SOFTWARE SWITCH REG
3780
3781 013634 022737 000176 000570 GTSWR:  CMP    #SWREG,SWR      ;BRANCH IF SOFTWARE SWR NOT
3782 013642 001032                BNE    1$              ;INVOKED
3783 013644 004737 013732        JSR    PC,.SAVE       ;SAVE REGISTERS ON THE STACK
3784 013650 012704 016156        MOV    #SMSWR,R4      ;TYPE 'SWR = '
3785 013654 004737 013054        JSR    PC,TTOUT
3786 013660 017703 164704        MOV    @SWR,R3        ;GET CURRENT SETTING
3787 013664 004737 013172        JSR    PC,OCTPE      ;AND TYPE THEM
3788 013670 012704 016166        MOV    #SMNEW,R4     ;TYPE 'NEW = '
3789 013674 004737 013054        JSR    PC,TTOUT
3790 013700 013705 000570        MOV    SWR,R5        ;TTR ROUTINE RETURN NEW VALUE TO (R5)
3791 013704 012701 000007        MOV    #7,R1         ;LIMIT RESPONSE TO 7 CHARS
3792 013710 012702 177777        MOV    #177777,R2    ;BETWEEN 0 AND 177777
3793 013714 012703 000000        MOV    #0,R3
3794 013720 004737 012514        JSR    PC,TTR        ;GET RESPONSE
3795 013724 004737 013754        JSR    PC,.RESTORE   ;RESTORE REGISTERS
3796 013730 000207                RTS     PC            ;RETURN TO CALLER
3797
3798      ;;ROUTINE TO SAVE REGISTERS ON THE STACK
      .SAVE:  MOV    %5,-(SP)      ;;R5 IS SAVED AT 12(SP)
              MOV    %4,-(SP)      ;;R4 IS SAVED AT 10(SP)
              MOV    %3,-(SP)      ;;R3 IS SAVED AT 6(SP)
              MOV    %2,-(SP)      ;;R2 IS SAVED AT 4(SP)
              MOV    %1,-(SP)      ;;R1 IS SAVED AT 2(SP)
              MOV    %0,-(SP)      ;;R0 IS SAVED AT (SP)
              MOV    14(SP),-(SP)   ;;PUSH RETURN PC ON THE STACK
              RTS     PC            ;;RETURN TO CALLER
3799      ;;ROUTINE TO RESTORE REGISTERS SAVED ON THE STACK
      .RESTORE:MOV    (SP)+,14(SP)  ;;STORE RETURN PC ON STACK
              MOV    (SP)+,%0
              MOV    (SP)+,%1
              MOV    (SP)+,%2
              MOV    (SP)+,%3
              MOV    (SP)+,%4
              MOV    (SP)+,%5
              RTS     PC            ;;RETURN
  
```

```
3801
3802 ;MESSAGE TABLE*****
3803
3804 013776 022445 046524 031460 MSG1: .ASCII'%TM03-TU45 CONTROL LOGIC TEST PART II (CZTUPB0)';**B
      014004 052055 032125 020065
      014012 047503 052116 047522
      014020 020114 047514 044507
      014026 020103 042524 052123
      014034 050040 051101 020124
      014042 044511 020040 041450
      014050 052132 050125 030102
      014056 051
3805 014057 045 025052 040452 .ASCII /%***ASSURE TAPE IS AT BOT***/
      014064 051523 051125 020105
      014072 040524 042520 044440
      014100 020123 052101 041040
      014106 052117 025052 052
3806 014113 045 054524 042520 .ASCII /%TYPE <CR> TO TERMINATE RESPONSE & ^C TO RESTART%/
      014120 036040 051103 020076
      014126 047524 052040 051105
      014134 044515 040516 042524
      014142 051040 051505 047520
      014150 051516 020105 020046
      014156 041536 052040 020117
      014164 042522 052123 051101
      014172 022524 043
3807 014175 105 050130 026524 MSG6: .ASCII /EXPT-NOT RECVD#/
      014202 047516 020124 042522
      014210 053103 021504
3808 014214 041522 042126 047055 MSG7: .ASCII /RCVD-NOT EXPT#/
      014222 052117 042440 050130
      014230 021524
3809 014232 047045 047117 042455 MSG9: .ASCII /%NON-EXIST SLAVE #/
      014240 044530 052123 051440
      014246 040514 042526 021440
3810 014254 051045 040505 020104 MSG10: .ASCII /%READ CONT BUS PAR #/
      014262 047503 052116 041040
      014270 051525 050040 051101
      014276 021440
3811 014300 ^53445 044522 042524 MSG11: .ASCII /%WRITE CONT BUS PAR #/
      014306 041440 047117 020124
      014314 052502 020123 040520
      014322 020122 043
3812 014325 040 054105 052120 MSG12: .ASCII / EXPT #/
      014332 021440
3813 014334 05040 053103 020104 MSG13: .ASCII / RCVD #/
      014342 043
3814 014343 045 051115 041040 MSG14: .ASCII /%MR BITS 4-0#/
      014350 052111 020123 026464
      014356 021460
3815 014360 046445 020122 044502 MSG15: .ASCII /%MR BITS 15-7#/
      014366 051524 030440 026465
      014374 021467
3816 014376 044445 042524 035122 MSG16: .ASCII /%ITER: #/
      014404 021440
3817 014406 052045 020103 044502 MSG18: .ASCII /%TC BITS 12-0 #/
```

	014414	051524	030440	026462		
	014422	020060	043			
3818	014425	045	041506	041040	MSG19:	.ASCII /%FC BITS 15-0 #/
	014432	052111	020123	032461		
	014440	030055	021440			
3819	014444	037440	021440		MSG40:	.ASCII / ? #/
3820	014450	022445	047105	020104	MSG41:	.ASCII /%%END OF PASS #/
	014456	043117	050040	051501		
	014464	020123	043			
3821	014467	045	042522	044507	MSG44:	.ASCII /%REGISTER START: #/
	014474	052123	051105	051440		
	014502	040524	052122	020072		
	014510	043				
3822	014511	045	042526	052103	MSG45:	.ASCII /%VECTOR ADDRESS: #/
	014516	051117	040440	042104		
	014524	042522	051523	020072		
	014532	043				
3823	014533	045	046524	031460	MSG57:	.ASCII /%TM03 DRIVE: #/
	014540	042040	044522	042526		
	014546	020072	043			
3824	014551	045	052524	032464	MSG58:	.ASCII /%TU45 SLAVE: #/
	014556	051440	040514	042526		
	014564	020072	043			
3825	014567	045	043		MSG59:	.ASCII /%#/
3826	014571	134	043		MSG60:	.ASCII /\#/
3827	014573	045	042522	047515	MSG62:	.ASCII /%REMOVE TMDP FROM SLAVE TO BE TESTED%/
	014600	042526	052040	042115		
	014606	020120	051106	046517		
	014614	051440	040514	042526		
	014622	052040	020117	042502		
	014630	052040	051505	042524		
	014636	022504	043			
3828	014641	045	040510	042122	MSG63:	.ASCII /%HARDWARE SWR IN USE%/
	014646	040527	042522	051440		
	014654	051127	044440	020116		
	014662	051525	022505	043		
3829	014667	045	020045	051104	MSG64:	.ASCII /%% DRIVE #/
	014674	053111	020105	043		
3830	014701	054	046123	053101	MSG65:	.ASCII /,SLAVE #/
	014706	020105	043			
3831	014711	116	052117	047440	MSG66:	.ASCII /NOT ON LINE #/
	014716	020116	044514	042516		
	014724	021440				

```
3833 ;TEST HEADER*****
3834
3835 014726 022445 047514 044507 MSLT1: .ASCII /%%LOGIC TEST 1: WRAP 3,NRZ,NORMAL,ODD#/
      014734 020103 042524 052123
      014742 030440 020072 051127
      014750 050101 031440 047054
      014756 055122 047054 051117
      014764 040515 026114 042117
      014772 021504
3836 014774 022445 047514 044507 MSLT2: .ASCII /%%LOGIC TEST 2: WRAP 3,PE,NORMAL,ODD#/
      015002 020103 042524 052123
      015010 031040 020072 051127
      015016 050101 031440 050054
      015024 020505 047516 046522
      015032 046101 047454 042104
      015040 043
3837 015041 045 046045 043517 MSLT3: .ASCII /%%LOGIC TEST 3: WRAP 2,NRZ,NORMAL,ODD#/
      015046 041511 052040 051505
      015054 020124 035063 053440
      015062 040522 020120 026062
      015070 051116 026132 047516
      015076 046522 046101 047454
      015104 042104 043
3838 015107 045 046045 043517 MSLT4: .ASCII /%%LOGIC TEST 4: WRAP 2,PE,NORMAL,ODD#/
      015114 041511 052040 051505
      015122 020124 035064 053440
      015130 040522 020120 026062
      015136 042520 047054 051117
      015144 040515 026114 042117
      015152 021504
3839 015154 022445 047514 044507 MSLT5: .ASCII /%%LOGIC TEST 5: WRAP 1,NRZ,NORMAL,ODD#/
      015162 020103 042524 052123
      015170 032440 020072 051127
      015176 050101 030440 047054
      015204 055122 047054 051117
      015212 040515 026114 042117
      015220 021504
3840 015222 022445 047514 044507 MSLT6: .ASCII /%%LOGIC TEST 6: WRAP 1,PE,NORMAL,ODD#/
      015230 020103 042524 052123
      015236 033040 020072 051127
      015244 050101 030440 050054
      015252 026105 047516 046522
      015260 046101 047454 042104
      015266 043
3841 015267 045 046045 043517 MSLT7: .ASCII /%%LOGIC TEST 7: WRAP 0,NRZ,NORMAL,ODD#/
      015274 041511 052040 051505
      015302 020124 035067 053440
      015310 040522 020120 026060
      015316 051116 026132 047516
      015324 046522 046101 047454
      015332 042104 043
3842 015335 045 046045 043517 MSLT10: .ASCII /%%LOGIC TEST 10: WRAP 0,PE,NORMAL,ODD#/
      015342 041511 052040 051505
      015350 020124 030061 020072
      015356 051127 050101 030040
      015364 050054 026105 047516
```

	015372	046522	046101	047454	
	015400	042104	043		
3843	015403	045	046045	043517	MSLT11: .ASCII /%%LOGIC TEST 11: CORE DUMP WRITE (M8906)#/
	015410	041511	052040	051505	
	015416	020124	030461	020072	
	015424	047503	042522	042040	
	015432	046525	020120	051127	
	015440	052111	020105	046450	
3844	015446	034470	033060	021451	MSLT12: .ASCII /%%LOGIC TEST 12: CORE DUMP READ (M8906)#/
	015454	022445	047514	044507	
	015462	020103	042524	052123	
	015470	030440	035062	041440	
	015476	051117	020105	052504	
	015504	050115	051040	040505	
	015512	020104	046450	034470	
3845	015520	033060	021451		MSLT13: .ASCII /%%LOGIC TEST 13: EVEN PARITY WRITE (M8933 M8934)#/
	015524	022445	047514	044507	
	015532	020103	042524	052123	
	015540	030440	035063	042440	
	015546	042526	020116	040520	
	015554	044522	054524	053440	
	015562	044522	042524	024040	
	015570	034115	031471	020063	
	015576	034115	031471	024464	
3846	015604	043			MSLT14: .ASCII /%%LOGIC TEST 14: EVEN PARITY READ(M8933 M8934)#/
	015605	045	046045	043517	
	015612	041511	052040	051505	
	015620	020124	032061	020072	
	015626	053105	047105	050040	
	015634	051101	052111	020131	
	015642	042522	042101	046450	
	015650	034470	031463	046440	
3847	015656	034470	032063	021451	MSLT15: .ASCII /%%LOGIC TEST 15: READ REVERSE(M8906)#/
	015664	022445	047514	044507	
	015672	020103	042524	052123	
	015700	030440	035065	051040	
	015706	040505	020104	042522	
	015714	042526	051522	024105	
	015722	034115	030071	024466	
3848	015730	043			MSLT16: .ASCII /%%LOGIC TEST 16: CRC CORRECTION SINGLE TRACK,ALL FRAMES#/
	015731	045	046045	043517	
	015736	041511	052040	051505	
	015744	020124	033061	020072	
	015752	051103	020103	047503	
	015760	051122	041505	044524	
	015766	047117	051440	047111	
	015774	046107	020105	051124	
	016002	041501	026113	046101	
	016010	020114	051106	046501	
3849	016016	051505	043		MSLT17: .ASCII /%%LOGIC TEST 17: CRC CORRECTION MULTIPLE BAD TRACKS#/
	016021	045	046045	043517	
	016026	041511	052040	051505	
	016034	020124	033461	020072	
	016042	051103	020103	047503	
	016050	051122	041505	044524	
	016056	047117	046440	046125	

3850 016064 044524 046120 020105  
016072 040502 020104 051124  
016100 041501 051513 043  
016105 045 046045 043517  
016112 041511 052040 051505  
016120 020124 030062 020072  
016126 042522 042101 051040  
016134 053105 051105 042523  
016142 047054 055122 053454  
016150 040522 020120 021463

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

```
3852
3853 ;TAG MESSAGE
3854
3855 016156 051445 051127 036440 SMSWR: .ASCII /%SWR = #/
      016164 021440
3856 016166 047040 053505 036440 SMNEW: .ASCII / NEW = #/
      016174 021440
3857
3858 .EVEN
3859 ;WRITE BUFFER
3860
3861 016176 000100 WDATA:
3863 016176 177777 -1
(1) 016200 177777 -1
(1) 016202 177777 -1
(1) 016204 177777 -1
(1) 016206 177777 -1
(1) 016210 177777 -1
(1) 016212 177777 -1
(1) 016214 177777 -1
(1) 016216 177777 -1
(1) 016220 177777 -1
(1) 016222 177777 -1
(1) 016224 177777 -1
(1) 016226 177777 -1
(1) 016230 177777 -1
(1) 016232 177777 -1
(1) 016234 177777 -1
(1) 016236 177777 -1
(1) 016240 177777 -1
(1) 016242 177777 -1
(1) 016244 177777 -1
(1) 016246 177777 -1
(1) 016250 177777 -1
(1) 016252 177777 -1
(1) 016254 177777 -1
(1) 016256 177777 -1
(1) 016260 177777 -1
(1) 016262 177777 -1
(1) 016264 177777 -1
(1) 016266 177777 -1
(1) 016270 177777 -1
(1) 016272 177777 -1
(1) 016274 177777 -1
(1) 016276 177777 -1
(1) 016300 177777 -1
(1) 016302 177777 -1
(1) 016304 177777 -1
(1) 016306 177777 -1
(1) 016310 177777 -1
(1) 016312 177777 -1
(1) 016314 177777 -1
(1) 016316 177777 -1
(1) 016320 177777 -1
(1) 016322 177777 -1
(1) 016324 177777 -1
```

(1)	016326	177777	-1
(1)	016330	177777	-1
(1)	016332	177777	-1
(1)	016334	177777	-1
(1)	016336	177777	-1
(1)	016340	177777	-1
(1)	016342	177777	-1
(1)	016344	177777	-1
(1)	016346	177777	-1
(1)	016350	177777	-1
(1)	016352	177777	-1
(1)	016354	177777	-1
(1)	016356	177777	-1
(1)	016360	177777	-1
(1)	016362	177777	-1
(1)	016364	177777	-1
(1)	016366	177777	-1
(1)	016370	177777	-1
(1)	016372	177777	-1
(1)	016374	177777	-1

3864  
3865  
3866  
3867

:READ BUFFER

3868	016376	000100	RDATA:	0
3870	016376	000000		0
(1)	016400	000000		0
(1)	016402	000000		0
(1)	016404	000000		0
(1)	016406	000000		0
(1)	016410	000000		0
(1)	016412	000000		0
(1)	016414	000000		0
(1)	016416	000000		0
(1)	016420	000000		0
(1)	016422	000000		0
(1)	016424	000000		0
(1)	016426	000000		0
(1)	016430	000000		0
(1)	016432	000000		0
(1)	016434	000000		0
(1)	016436	000000		0
(1)	016440	000000		0
(1)	016442	000000		0
(1)	016444	000000		0
(1)	016446	000000		0
(1)	016450	000000		0
(1)	016452	000000		0
(1)	016454	000000		0
(1)	016456	000000		0
(1)	016460	000000		0
(1)	016462	000000		0
(1)	016464	000000		0
(1)	016466	000000		0
(1)	016470	000000		0
(1)	016472	000000		0



3886	016741	045	041104	021440	WMSG6I:	.ASCII	/%DB #/
3887	016746	046445	020122	043	WMSG6J:	.ASCII	/%MR #/
3888	016753	045	052104	021440	WMSG6K:	.ASCII	/%DT #/
3889	016760	052045	020103	043	WMSG6L:	.ASCII	/%TC #/
3890	016765	045	047123	021440	WMSG6M:	.ASCII	/%SN #/
3891	016772	041045	042101	042040	WMSG16:	.ASCII	/%BAD DATA#/
	017000	052101	021501				
3892	017004	043445	020072	043	WMSG17:	.ASCII	/%G: #/
3893	017011	045	035102	021440	WMSG20:	.ASCII	/%B: #/
3894	017016	041445	035116	021440	WMSG21:	.ASCII	/%CN: #/
3895	017024	041045	042101	051440	WMSG23:	.ASCII	/%BAD STATUS#/
	017032	040524	052524	021523			
3896	017040	047045	020117	047111	WMSG24:	.ASCII	/%NO INTERRUPT#/
	017046	042524	051122	050125			
	017054	021524					
3897	017056	047045	020117	046103	WMSG25:	.ASCII	/%NO CLOCK UP#/
	017064	041517	020113	050125			
	017072	043					
3898	017073	045	047516	041440	WMSG26:	.ASCII	/%NO CLOCK DOWN#/
	017100	047514	045503	042040			
	017106	053517	021516				
3899	017112	042045	052101	020101	WMSG27:	.ASCII	/%DATA PAT: #/
	017120	040520	035124	043			
3900	017125	045	040502	020104	WMSG28:	.ASCII	/%BAD PREAMBLE#/
	017132	051120	040505	041115			
	017140	042514	043				
3901	017143	045	040502	020104	WMSG29:	.ASCII	/%BAD POSTAMBLE#/
	017150	047520	052123	046501			
	017156	046102	021505				
3902	017162	042445	051117	041440	WMSG31:	.ASCII	/%EOR CLEAR DID NOT CLEAR GOX#/
	017170	042514	051101	042040			
	017176	042111	047040	052117			
	017204	041440	042514	051101			
	017212	043440	022517	043			
3903	017217	040	040520	051124	WMSG32:	.ASCII	/%PATRN #/
	017224	020116	043				
3904							
3905		017230					
3906	017230	000000			PRE:	0	.EVEN
3909	017232	000000				0	
(1)	017234	000000				0	
(1)	017236	000000				0	
(1)	017240	000000				0	
(1)	017242	000000				0	
(1)	017244	000000				0	
(1)	017246	000000				0	
(1)	017250	000000				0	
(1)	017252	000000				0	
(1)	017254	000000				0	
(1)	017256	000000				0	
(1)	017260	000000				0	
(1)	017262	000000				0	
(1)	017264	000000				0	
(1)	017266	000000				0	
(1)	017270	000000				0	
(1)	017272	000000				0	



(1)	017454	000000	0
(1)	017456	000000	0
(1)	017460	000000	0
(1)	017462	000000	0
(1)	017464	000000	0
(1)	017466	000000	0
(1)	017470	000000	0
(1)	017472	000000	0
3914	017474	000000	WBUFF: 0
3915		020106	. = .+410
3916	020106	000000	RBUFF: 0
3917			
3918		000001	.END

















DTBOOT	1243#		
GETANS	767#		
LDPDR	515#		
LPDP11	1268#		
PSPTAG	746#		
REGBOX	132#		
RESLDR	873#		
SAVLDR	855#		
SVTK\$	1141#		
\$CATCH	1124#	1855#	1942
\$CHAIN	89#	1855#	2176
\$CHNMO	105#	1855#	2253
\$CNV16	606#		
\$CNV18	635#		
\$CNV48	704#		
\$CPCHK	897#		
\$CPREG	17#		
\$CPVEC	167#		
\$FPREG	46#		
\$GETAN	771#		
\$KMMRE	347#		
\$KWPDOR	998#		
\$KW11	929#		
\$LCTRL	2#		
\$LPREG	186#		
\$MAMFO	1176#		
\$MMBIT	207#		
\$MMREG	264#		
\$PDRBI	385#		
\$POWER	439#		
\$PSWBI	147#		
\$RECO	795#		
\$RESLD	876#		
\$RESTO	476#	1855#	3799
\$SAVE	464#	1855#	3798
\$SAVLD	858#		
\$SETTB	508#		
\$SHIFT	489#		
\$SMRE	309#		
\$STINS	8#		
\$STKPT	202#		
\$ST200	1136#		
\$SVTK	1146#		
\$SWOPT	56#		
\$TCDRV	1029#		
\$TCREG	192#		
\$TRAPS	402#		
\$TYPE	518#		
\$TYPEF	592#		
\$UMRE	272#		
\$VECTA	1163#		
.\$ACT1	67#	1855#	1944
.\$EOP	78#	1855#	2300

. ABS. 020110 000

TMO3-TU45 CONTROL LOGIC TEST-PART II  
CZTUPB.P11 13-MAY-80 15:06

MACY11 30A(1052) 13-MAY-80<sup>6 7</sup> 15:09 PAGE 53-1  
CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0084

ERRORS DETECTED: 0

CZTUPB,CZTUPB/CRF=CZTUPB.P11  
RUN-TIME: 14 23 2 SECONDS  
RUN-TIME RATIO: 101/41-2.4  
CORE USED: 14K (27 PAGES)