TA11

CASSETTE LOADER
MD-11-DZTAF-C

EP-DZTAF-C-DL-A

FICHE 1 OF 1

NOV 1976
digital
MADE IN USA

```
                        .REM    †
```

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZTAF-C-D

PRODUCT NAME: TA11 CASSETTE ABSOLUTE LOADER (TALDR)

DATE: 21 JANUARY, 1974

MAINTAINER: DIAGNOSTIC GROUP

AUTHOR: R. KOLLER

TABLE OF CONTENTS

1. ABSTRACT

TALDR IS A LOADER PROGRAM DESIGNED TO LOAD MAINDEC-11 DIAGNOSTIC PROGRAMS FROM TA11 CASSETTES.

THE TALDR LOADER IS THE FIRST FILE STORED ON A TA11 MAINDEC-11 CASSETTE. IT IS LABELLED "TALDRB.SYS". IT IS FOLLOWED BY ONE OR MORE MAINDEC-11 DIAGNOSTIC PROGRAMS STORED IN ABS LOADER FORMAT.

EACH TA11 MAINDEC-11 CASSETTE IS PROVIDED WITH A DIRECTORY, THAT LISTS THE TALDR LOADER, AND ALL THE PROGRAMS STORED IN THE CASSETTE.

EACH FILE NAME IN THE CASSETTE IS NUMBERED WITH AN OCTAL SEQUENCE NUMBER. TALDR HAS THE SEQUENCE NUMBER OF 1. THE FILE NUMBER IS USED TO SPECIFY TO THE TALDR LOADER WHICH PROGRAM IS TO BE LOADED.

IN ORDER TO PREVENT ACCIDENTAL ERASURE OF CASSETTES, EACH CASSETTE TO BE LOADED FROM, SHOULD BE "WRITE-LOCKED". WRITE LOCKING OF A CASSETTE IS ACCOMPLISHED BY UNCOVERING THE SMALL HOLES AT THE BACK OF THE CASSETTE BY FLIPPING THE SMALL PLASTIC TABS OUT OF THE WAY.

ABILITY OF THE TALDR LOADER TO SUCCESSFULLY LOAD A PROGRAM DEPENDS ON THE SIZE OF THE PROGRAM TO BE LOADED. TALDR CAN NOT LOAD A PROGRAM THAT INFRINGES ON ITS OWN STORAGE AREA.

2. REQUIREMENTS

2.1 HARDWARE REQUIREMENTS

A. A PDP11 PROCESSOR
B. BM792-YH CASSETTE BOOTSTRAP ROM (MAY BE SIMULATED BY A TA11 BOOTSTRAP LOADER PROGRAM)
C. TA11 TAPE CASSETTE CONTROL UNIT
D. TU60 TAPE CASSETTE TRANSPORT
E. 8K MINIMUM STORAGE

2.2 SOFTWARE REQUIREMENTS

A. MAINDEC-11 DIAGNOSTIC TAPE CASSETTES
B. TA11 BOOTSTRAP LOADER PROGRAM (IF BM792-YH IS NOT INSTALLED) (THIS PROGRAM MUST BE MANUALLY TOGGLED IN)

2.3 STORAGE REQUIREMENTS

ONCE LOADED AND RELOCATED, THE TALDR LOADER OCCUPIES THE UPPERMOST 258(10) WORDS OF AVAILABLE MEMORY, UP TO 28K.

3. USE PROCEDURE

3.1 LOADING FROM UNIT 0. BM792YH ROM INSTALLED

IN THE STEPS THAT FOLLOW, ANY HALT AT OTHER THAN LOC XX7014 IS AN ERROR. REFER TO SECTION 4. HALTS, FOR HALT INFORMATION.

A. MOUNT WRITE-LOCKED CASSETTE IN UNIT 0. (LEFT HAND DRIVE).

B. LOAD ADDRESS 173300 (BM792YH START ADDRESS)

C. SET SR WITH FILE NUMBER OF PROGRAM TO BE LOADED. REFER TO CASSETTE DIRECTORY. FILE NUMBER MUST BE GREATER THAN 1.

D. PRESS START

E. THE CASSETTE IS REWOUND, TALDR IS LOADED, RELOCATED, AND PROGRAM HALTS AT LOC XX7014.

F. THE HALT AT LOC XX7014 INDICATES THAT TALDR WAS SUCCESSFULLY LOADED, AND IS READY TO LOAD A PROGRAM. SINCE YOU HAVE ALREADY SET THE SR TO DESIRED PROGRAM NUMBER, PRESS CONT TO LOAD THE PROGRAM.

G. THE CASSETTE IS REWOUND. TALDR SEARCHES FORWARD FOR THE DESIRED PROGRAM, AND LOADS IT.

H. IF THE PROGRAM JUST LOADED IS SELF-STARTING, THE PROGRAM IS STARTED. IF NOT, THE PROGRAM HALTS AT LOC XX7014, INDICATING A SUCCESSFUL LOAD, AND READINESS TO LOAD ANOTHER PROGRAM. AT THIS POINT, THE PROGRAM JUST LOADED CAN BE STARTED MANUALLY (MOST MAINDEC-11 PROGRAMS START AT LOC 000200), OR ANOTHER PROGRAM CAN BE LOADED.

I. TO LOAD ANOTHER PROGRAM, SET THE SR TO FILE NUMBER OF DESIRED PROGRAM, AND PRESS CONT. GO TO STEP E.

3.2 LOADING FROM UNIT 1. BM792YH ROM INSTALLED

IN THE STEPS THAT FOLLOW, ANY HALT OTHER THAN AT LOC XX7014 IS AN ERROR. REFER TO SECTION 4. HALTS.

A. MOUNT WRITE-LOCKED CASSETTE ON UNIT 1. (RIGHT HAND DRIVE).

B. DEPOSIT 177500 INTO R0 (SEE NOTE BELOW)

C. DEPOSIT 000400 INTO LOCATION 177500 (SELECTS UNIT 1) (SEE NOTE BELOW)

D. DEPOSIT 173306 IN PC (R7) (BM792YH START ADDRESS+6)

E. SET SR WITH FILE NUMBER OF PROGRAM TO BE LOADED. REFER TO CASSETTE DIRECTORY. FILE NUMBER MUST BE GREATER THAN 1.

F. PRESS CONT

G. THE CASSETTE IS REWOUND, TALDR IS LOADED, RELOCATED, AND PROGRAM HALTS AT LOC XX7014

H. THE HALT AT LOC XX7014 INDICATES THAT TALDR HAS SUCCESSFULLY LOADED, AND IS READY TO LOAD A PROGRAM. SINCE YOU HAVE ALREADY SET THE SR WITH THE DESIRED FILE NUMBER (STEP E), PRESS CONT TO LOAD PROGRAM.

I. THE CASSETTE IS REWOUND. TALDR SEARCHES FORWARD FOR THE DESIRED PROGRAM, AND LOADS IT.

J. IF THE PROGRAM JUST LOADED IS SELF-STARTING, THE PROGRAM IS STARTED. IF NOT, THE PROGRAM HALTS AT LOC XX7014, INDICATING A SUCCESSFUL LOAD, AND READINESS TO LOAD ANOTHER PROGRAM. AT THIS POINT, THE PROGRAM JUST LOADED CAN BE STARTED MANUALLY (MOST MAINDEC-11 PROGRAMS START AT LOC 000200), OR ANOTHER PROGRAM CAN BE LOADED.

K. TO LOAD ANOTHER PROGRAM, SET THE SR TO FILE NUMBER OF DESIRED PROGRAM, AND PRESS CONT. GO TO STEP G.

NOTE
STEP B. FOR TA11 WITH NON-STANDARD ADDRESS, DEPOSIT NON-STANDARD ADDRESS IN R0.
STEP C. FOR TA11 WITH NON-STANDARD ADDRESS, DEPOSIT 000400 IN NON-STANDARD ADDRESS.

3.3 LOADING FROM UNIT 0. NO BM792YH ROM

IN THE STEPS THAT FOLLOW, ANY HALT AT OTHER THAN LOC XX7014 IS AN ERROR. REFER TO SECTION 4. HALTS.

A. MOUNT WRITE-LOCKED CASSETTE ON UNIT 0. (LEFT HAND DRIVE).

B. TOGGLE-IN (DEPOSIT) THE 28 WORD SOFTWARE BOOT SHOWN IN APPENDIX B, STARTING AT LOCATION XX7706.

C. LOAD ADDRESS XX7706

D. SET SR WITH THE FILE NUMBER OF PROGRAM TO BE LOADED. REFER TO CASSETTE DIRECTORY. FILE NUMBER MUST BE GREATER THAN 1.

E. PRESS START

F. THE CASSETTE IS REWOUND, TALDR IS LOADED, RELOCATED, AND PROGRAM HALTS AT LOC XX7014.

G. THE HALT AT LOC XX7014 INDICATES THAT TALDR WAS SUCCESSFULLY LOADED, AND IS READY TO LOAD A PROGRAM. SINCE YOU HAVE ALREADY SET THE SR TO THE DESIRED PROGRAM NUMBER, PRESS CONT TO LOAD THE PROGRAM.

H. THE CASSETTE IS REWOUND, TALDR SEARCHES FORWARD FOR THE DESIRED PROGRAM, AND LOADS IT.

I. IF THE PROGRAM JUST LOADED IS SELF STARTING, THE PROGRAM IS STARTED. IF NOT, THE PROGRAM HALTS AT LOC XX7014, INDICATING A SUCCESSFUL LOAD, AND READINESS TO LOAD ANOTHER PROGRAM. AT THIS POINT, THE PROGRAM JUST LOADED CAN BE STARTED MANUALLY (MOST MAINDEC-11 PROGRAMS START AT LOC 000200), OR ANOTHER PROGRAM CAN BE LOADED.

J. TO LOAD ANOTHER PROGRAM, SET THE SR TO FILE NUMBER OF DESIRED PROGRAM, AND PRESS CONT. GO TO STEP H.

3.4 LOADING FROM UNIT 1. NO BM792YH ROM

IN THE STEPS THAT FOLLOW, ANY HALT AT OTHER THAN AT LOC XX7014 IS AN ERROR. REFER TO SECTION 4. HALTS.

A. MOUNT WRITE-LOCKED CASSETTE ON UNIT 1. (RIGHT HAND DRIVE).

B. TOGGLE-IN (DEPOSIT IN CORE) THE 28 WORD SOFTWARE BOOT SHOWN IN APPENDIX B, STARTING AT LOC XX7706

C. DEPOSIT 177500 INTO R0 (SEE NOTE BELOW)

D. DEPOSIT 000400 INTO LOCATION 177500 (SELECTS UNIT 1) (SEE NOTE BELOW)

E. DEPOSIT XX7714 IN PC (R7)

F. SET SR WITH FILE NUMBER OF PROGRAM TO BE LOADED. REFER TO CASSETTE DIRECTORY. FILE NUMBER MUST BE GREATER THAN 1.

G. PRESS CONT

H. THE CASSETTE IS REWOUND, TALDR IS LOADED AND RELOCATED, AND THE PROGRAM HALTS AT LOC XX7014.

I. THE HALT AT LOC XX7014 INDICATES THAT TALDR WAS LOADED SUCCESSFULLY, AND IS READY TO LOAD A PROGRAM. SINCE YOU HAVE ALREADY SET THE SR WITH THE DESIRED PROGRAM NUMBER, PRESS CONT TO LOAD THE PROGRAM.

J. THE CASSETTE IS REWOUND, TALDR SEARCHES FORWARD FOR THE DESIRED PROGRAM, AND LOADS IT.

K. IF THE PROGRAM JUST LOADED IS SELF-STARTING, THE PROGRAM IS STARTED. IF NOT, THE PROGRAM HALTS AT LOC XX7014, INDICATING A SUCCESSFUL LOAD, AND READINESS TO LOAD ANOTHER PROGRAM. AT THIS POINT THE PROGRAM JUST LOADED CAN BE STARTED MANUALLY (MOST MAINDEC-11 PROGRAMS START AT LOC 000200), OR ANOTHER PROGRAM CAN BE LOADED.

L. TO LOAD ANOTHER PROGRAM, SET THE SR TO FILE NUMBER OF DESIRED PROGRAM, AND PRESS CONT. GO TO STEP J.

NOTE
STEP C. FOR TA11 WITH NON-STANDARD ADDRESS, DEPOSIT NON-STANDARD ADDRESS IN R0.
STEP D. FOR TA11 WITH NON-STANDARD ADDRESS, DEPOSIT 000400 IN NON-STANDARD ADDRESS.

3.5 LOADING A PROGRAM AFTER EXECUTING PREVIOUSLY LOADED PROGRAM

IT IS POSSIBLE TO LOAD A PROGRAM FROM CASSETTE AFTER A PREVIOUSLY LOADED PROGRAM HAS EXECUTED, PROVIDED THAT THE PREVIOUS PROGRAM HAS NOT DESTROYED THE TALDR LOADER. TO LOAD A PROGRAM:

A. MOUNT WRITE-LOCKED CASSETTE ON SAME UNIT AS PREVIOUSLY USED. (TALDR SAVED THE TA11 ADDRESS, AND UNIT NUMBER USED IN PREVIOUS LOAD).

B. LOAD ADDRESS XX7776

C. PRESS START

D. PROGRAM SHOULD HALT AT LOC XX7014. IF ANYTHING ELSE HAPPENS, TALDR HAS BEEN DESTROYED. USE ONE OF THE LOADING PROCEDURES DESCRIBED IN SECTIONS 3.1 THROUGH 3.4

E. SET SR WITH FILE NUMBER OF PROGRAM TO BE LOADED. REFER TO CASSETTE DIRECTORY. FILE NUMBER MUST BE GREATER THAN 1.

F. PRESS CONT

G. THE CASSETTE IS REWOUND, TALDR SEARCHES FORWARD FOR DESIRED PROGRAM AND LOADS IT.

H. IF THE PROGRAM JUST LOADED IS SELF-STARTING THE PROGRAM IS STARTED. IF NOT, THE PROGRAM HALTS AT LOC XX7014, INDICATING A SUCCESSFUL LOAD, AND READINESS TO LOAD ANOTHER PROGRAM. AT THIS POINT, THE PROGRAM JUST LOADED CAN BE STARTED MANUALLY, OR ANOTHER PROGRAM CAN BE LOADED.

3.6 DIAGNOSTIC MODE OPERATION

TALDR HAS A DIAGNOSTIC MODE, DESIGNED TO AID THE MAINTENANCE ENGINEER IN DIAGNOSING TA11 PROBLEMS. DIAGNOSTIC MODE OPERATES DURING LOADING AND RELOCATING OF THE TALDR LOADER, AND IS ACTIVATED BY SETTING THE SR TO 000000, INSTEAD OF THE FILE NUMBER OF A PROGRAM TO BE LOADED. THE SEQUENCE THAT FOLLOWS ASSUMES USE OF THE BM792YH, TA11 CONTROL AT STANDARD ADDRESS, AND USE OF UNIT 0 .

A. MOUNT WRITE-LOCKED CASSETTE ON UNIT 0. (LEFT HAND DRIVE).

B. LOAD ADDRESS 173300

C. SET SR TO 000000 (INDICATES DIAGNOSTIC MODE).

D. PRESS START

E. CASSETTE REWINDS. TAPE MOVES FORWARD AND STOPS. PROGRAM HALTS AT LOC 000016. THIS IS THE "POST BOOT" HALT. AT THIS POINT THE FIRST 64 WORDS OF THE TALDR LOADER SHOULD BE STORED IN MEMORY, STARTING AT LOC 000000. THESE 1ST 64 WORDS ARE REFERRED TO AS THE "PRE-LOADER". THE PRELOADER CONTENTS CAN BE VERIFIED BY EXAMINING CORE STARTING AT LOC 000000, AND REFERENCING THE TALDR LISTING. AFTER VERIFICATION, OR IF NOT VERIFYING, PRESS CONT.

F. THE CASSETTE REWINDS, MOVES FORWARD, AND THEN HALTS AT LOC 000132. THIS OPERATION IS PERFORMED BY THE "PRE-LOADER"; THE 64 WORDS OF CODE READ IN DURING STEP E. IF THE CASSETTE GETS HUNG (DOES NOT STOP, DOES NOT EVEN GET STARTED), THE PRE-LOADER CODE SHOULD BE EXAMINED FOR CORRECT OPERATION. WHEN THE PROGRAM HALTS AT LOC 000132, THE ENTIRE TALDR LOADER HAS BEEN READ INTO LOWER CORE. THE HALT OCCURS JUST PRIOR TO RELOCATING THE TALDR LOADER TO UPPER MEMORY. R4 SHOULD CONTAIN THE VALUE XX6714, WHICH IS THE INITIAL ADDRESS WHERE TALDR IS TO BE RELOCATED. PRESS CONT.

G. TALDR IS RELOCATED TO UPPER MEMORY AND THE PROGRAM HALTS AT LOC XX7014, WHICH IS THE "READY TO LOAD" HALT. AT THIS POINT A PROGRAM NUMBER MAY BE SET IN THE SR TO LOAD A PROGRAM. R0 SHOULD CONTAIN THE ADDRESS OF TA11 CONTROL (IN THIS CASE 177500). IF LOADING A PROGRAM, PRESS CONT.

4. HALTS

IN THE HALT DESCRIPTIONS THAT FOLLOW, HALTS ARE IDENTIFIED BY THE LOCATION OF THE HALT IN MEMORY, AND NOT BY ANY CONSOLE DISPLAY LIGHTS. IN ORDER TO CORRECTLY IDENTIFY A HALT ON THE VARIOUS PDP-11 PROCESSORS, USE THE TABLE BELOW:

CPU USED	ADDRESS LIGHTS DISPLAY
11/20	HALT LOCATION
11/05	HALT LOCATION+2
11/45	HALT LOCATION+2
11/40	HALT LOCATION+2

LOC 000016	POST BOOT HALT. OCCCURS AFTER 1ST BLOCK OF TALDR HAS BEEN READ BY HARDWARE OR SOFTWARE BOOT, AND SR=0 (DIAGNOSTIC MODE OPERATION). PRESS CONT TO PROCEED WITH TALDR LOAD OPERATION.
LOC 000132	TALDR PRE-RELOCATION HALT. OCCURS IN DIAGNOSTIC MODE ONLY. INDICATES THAT THE ENTIRE TALDR LOADER HAS BEEN READ INTO CORE, AND THE PROGRAM IS READY TO RELOCATE THE TALDR LOADER TO UPPER CORE. PRESS CONT TO ACCOMPLISH RELOCATION.
LOC 000172	TA11 HARDWARE ERROR DETECTED IN PRE-LOADER SECTION OF TALDR LOADER (1ST 64 WORDS OF THE LOADER). PRESS CONT TO TRY AGAIN.
LOC XX7014	GOOD LOAD HALT; READINESS TO LOAD A PROGRAM. THE TALDR LOADER RESIDES IN UPPER MEMORY, AND IS READY TO LOAD A PROGRAM. THIS HALT ALSO OCCURS AFTER TALDR HAS SUCCESSFULLY LOADED A PROGRAM, AND IS READY TO LOAD ANOTHER PROGRAM.
LOC XX7026	ERROR. THE USER HAS SPECIFIED A PROGRAM FILE NUMBER THAT IS LESS THAN 2. THE 1ST FILE (TALDR) IS NOT A VALID PROGRAM FOR LOADING, AS IT IS NOT IN ABSOLUTE FORMAT. ALSO IF THE SR IS SET TO 0, TALDR DOES NOT KNOW WHAT TO DO ABOUT THAT. SET CORRECT FILE NUMBER IN SR AND PRESS CONT TWICE.
LOC XX7060	ERROR. SENTINEL FILE FOUND. THE END OF WRITTEN TAPE HAS BEEN FOUND BEFORE THE FILE SPECIFIED BY USER. INCORRECT FILE NUMBER HAS BEEN SPECIFIED. SET THE CORRECT FILE NUMBER IN SR AND PRESS CONT TWICE.

(4. CONT'D)

LOC XX7164	CHECKSUM ERROR. TALDR HAS DETECTED A CHECKSUM ERROR WHILE READING THE PROGRAM. TO TRY AGAIN PRESS CONT. TALDR REVERSES TO START OF PROGRAM AND TRIES AGAIN. IF AFTER SEVERAL TRIES THE HALT STILL OCCURS, THE PROGRAM IS NOT LOADABLE. GO TO SECTION 3.5 TO LOAD ANOTHER PROGRAM IF DESIRED.
LOC XX7210	TALDR LOADER OVERRUN. THE PROGRAM IN PROCESS OF BEING LOADED IS ENTERING INTO THE TALDR STORAGE SPACE, DUE TO ITS SIZE. THE PROGRAM CAN NOT BE LOADED COMPLETELY WITH THE SYSTEM'S CURRENT STORAGE CAPACITY.
LOC XX7462	TA11 ERROR. CASSETTE ERROR HAS BEEN DETECTED BY THE TALDR LOADER WHILE PERFORMING CASSETTE OPERATIONS. EXAMINE CONTENTS OF TA11 CONTROL REGISTER TO DETERMINE CAUSE OF ERROR. TO TRY AGAIN, PRESS CONT TWICE.
LOC XX7756	SOFTWARE BOOT TA11 ERROR. HARDWARE ERROR DETECTED BY SOFTWARE BOOT, OR 1ST BYTE OF THE BLOCK READ BY THE BOOT IS NOT A 240, INDICATING THAT THE FIRST PROGRAM IN THE CASSETTE IS NOT THE TALDR LOADER. IF HARDWARE ERROR (EXAMINE TA11 CSR), PRESS CONT TO TRY AGAIN. IF NOT HARDWARE ERROR, LOAD THE PROPER CASSETTE, AND PRESS CONT.
LOC 173350	TA11 ERROR, OR 1ST BYTE OF DATA READ FROM CASSETTE'S 1ST BLOCK NOT A 240. DETECTED BY BM792YH ROM. IF HARDWARE ERROR (EXAMINE TA11 CSR), PRESS CONT TO TRY AGAIN. IF NOT HARDWARE ERROR, LOAD THE CORRECT CASSETTE, AND PRESS CONT.

5. DESCRIPTION

5.1 BOOTSTRAP LOADER (CBOOT)

THIS PROGRAM IS SPECIFICALLY DESIGNED TO READ IN THE FIRST BLOCK OF THE TA11 ABSOLUTE LOADER PROGRAM, VERIFY THAT THE BLOCK READ IN IS THE 1ST BLOCK OF TALDR, AND TRANSFER PROGRAM CONTROL TO THE TALDR PROGRAM. IT IS AVAILABLE IN TWO VERSIONS, 1) HARD WIRED INTO THE BM792-YH ROM OR 2) AS A SOFTWARE COPY THAT MAY BE DEPOSITED BY THE OPERATOR. REFER TO APPENDIX A FOR A COPY OF THE BOOT LOADER PROGRAM.

REGISTER USAGE:

R0 - CONTAINS THE ADDRESS OF THE TA11 CONTROL AND STATUS REGISTER

R1 - USED AS A POINTER TO THE COMMAND TABLE THAT HOLDS THE REQUIRED TA11 CASSETTE COMMANDS

R2 - USED AS A DATA BUFFER POINTER AND DATA FLAG

R3 - HOLDS BIT TEST MASK REQUIRED TO QUERY THE TA11 CONTROL AND STATUS REGISTER

PROGRAM SEQUENCE:

CBOOT - THE FIRST INSTRUCTION LOADS R0 WITH THE TACSR ADDRESS, FOLLOWED BY A CLEAR OF THE TACSR TO INSURE SELECTING UNIT 0.

RESTRT - THE MOVE FOLLOWED BY THE ADD SETS UP R1 TO POINT TO THE FIRST BYTE IN THE COMMAND TABLE. THESE TWO INSTRUCTIONS RELATE THE LOCATION OF THE COMMAND TABLE TO THE PC WHICH INSURES THAT THE BOOT PROGRAM IS RELOCATABLE. FOLLOWING THIS, R2 IS LOADED WITH A CONSTANT OF 375(8) WHICH WILL FIRST SERVE TO DETECT THE BEGINNING OF THE DATA XFR AND THEN BE USED AS A DATA BUFFER POINTER TO LOAD 128 BYTES INTO CORE STARTING AT LOC. 0. R3 IS LOADED WITH THE FIRST BYTE IN THE COMMAND TABLE (A 240(8) WHICH ALLOWS TESTING THE "READY" AND "TRANSFER REQ" FLAGS IN THE TACSR) AND R1 AUTOMATICALLY UPDATED TO POINT TO THE SECOND BYTE AFTER AN ERROR HALT WILL RESTART THE LOADER AT "RESRTR".

(5.1 CONT'D)

LOOP1 - THE NEXT COMMAND IN THE COMMAND TABLE IS LOADED INTO THE TACSR AND R1 UPDATED TO POINT TO THE NEXT COMMAND. THIS LOOP IS EXECUTED FIVE TIMES TO FETCH THE FOLLOWING COMMANDS:

1) REWIND TO BEGINNING OF TAPE
2) SPACE FORWARD OVER HEADER BLOCK
3) READ FWD ONE BLOCK (128 BYTES TRANSFERRED INTO CORE LOC 0 TO 177)
4) CHECK THE CRC BYTES
5) TERMINATE (HIGH ORDER BIT OF COMMAND BYTE IS SET)

AFTER LOADING EACH COMMAND, A BMI TESTS FOR THE TERMINATOR AND TRANSFERS CONTROL TO "DONE" IF THE COMMAND BYTE IS NEGATIVE. IF NOT NEGATIVE THE PROGRAM FALLS THROUGH TO "LOOP2".

LOOP2 - A BITB FOLLOWED BY A BEQ TESTS FOR SUCCESSFUL COMPLETION OF THE CURRENT COMMAND. THE PROGRAM BRANCHES ON "LOOP2" UNTIL EITHER "READY" OR "TRANSFER REQUEST" IS ASSERTED. IF ONE OR BOTH OF THESE FLAGS ARE ASSERTED THE PROGRAM FALLS THROUGH AND INCREMENTS R2. (NOTE THAT R2 IS INCREMENTED WITH A BYTE INSTRUCTION AND STARTED AT 375) UNTIL THE "READ" COMMAND IS FETCHED THE PROGRAM LOOPS ON "LOOP1" VIA A BMI. AFTER THE READ COMMAND IS LOADED R2 INCREMENTS TO 000000 (HIGH ORDER CARRIES NOT PROPAGATED BEYOND BIT 7) AND THE PROGRAM FALLS THROUGH TO THE MOVB THAT LOADS DATA INTO CORE. IF THE FIRST CHARACTER READ IS NOT A 240(8) (INDICATING THE TALDR PROGRAM) THE PROGRAM BRANCHES TO A HALT AT "STOP". IF THE FIRST BYTE IS A 240(8) THE PROGRAM LOOPS ON "LOOP2" UNTIL 128 BYTES ARE TRANSFERRED INTO CORE AT WHICH TIME R2 GOES TO 200(8) (BECOMES NEGATIVE AGAIN). AT THIS TIME THE PROGRAM RETURNS TO "LOOP1" TO FETCH THE ILBS COMMAND FOLLOWED BY A TERMINATOR COMMAND.

(5.1 CONT'D)

STOP - PROGRAM BRANCHES TO A HALT HERE IF THE 1ST BYTE TRANSFERRED INTO CORE IS NOT A 240(8). DEPRESSING CONTINUE CAUSES PROGRAM TO BRANCH TO "RESTRT".

DONE - PROGRAM BRANCHES HERE AFTER FETCHING COMMAND TABLE TERMINATOR. IT THEN TESTS FOR CASSETTE CRC ERRORS AND CLEARS THE PC TO TRANSFER CONTROL TO LOCATION 0 IF NO ERRORS. IF AN ERROR IS DETECTED THE PROGRAM BRANCHES TO "STOP" AND HALTS.

TABLE - COMMAND TABLE:

```
TABLE+0  /  240        TEST FOR READY OR XFR REQ
     +1  /  037        REWIND
     +2  /  015        SPACE FWD BLOCK
     +3  /  005        READ
     +4  /  024        ILBS - TEST CRC
     +5  /  224        TERMINATOR
     +6  /  000        TWO WORDS OF FILLER
     +7  /  000
     +10 /  000
     +11 /  000
```

VECTOR - IN SOME OF THE PDP11 PROCESSORS THE POWER FAIL VECTOR IS DESIGNED TO POINT TO A HARDWARE ROM. THE LAST TWO WORDS OF THE BOOT LOADER POINT THE PC TO THE START OF THE TA BOOT LOADER AND SET THE PROCESSOR PRIORITY TO 7 IF THE POWER FAIL VECTOR IS DESIGNED TO POINT TO 773376. THESE TWO WORDS AND THE TWO FILLER WORDS IN THE COMMAND TABLE ARE NOT REQUIRED IN THE SOFTWARE VERSION OF THE BOOT LOADER.

5.2 TALDR TAPE CASSETTE ABSOLUTE LOADER

TALDR IS DESIGNED TO LOAD "ABS" FORMAT" PROGRAMS FROM MAINDEC-11 TA11 CASSETTES. THE PROGRAM CONSISTS OF TWO SECTIONS. FIRST, A 128 BYTE (64 WORDS) "PRE-LOADER" WHOSE FUNCTION IS TO READ IN THE ENTIRE TALDR LOADER INTO LOWER CORE, AND THEN RELOCATE IT TO THE TOP OF MEMORY. SECOND, THE CASSETTE LOADER ITSELF, WHOSE FUNCTION IS TO LOAD AN ABS FORMAT PROGRAM FROM CASSETTE WHOSE FILE NUMBER HAS BEEN SET IN THE SR.

5.2.1 PRE-LOADER PROGRAM SEQUENCE

GO: THE PROGRAM TESTS THE SWITCH REGISTER. IF SR = 0, THE PROGRAM HALTS AND THE OPERATOR MUST DEPRESS CONTINUE TO PROCEED TO "GO1". IF THE SR IS NOT ZERO THE PROGRAM FALLS THROUGH TO "GO1".

GO1: THE STACK POINTER IS SET UP AND THE CORE MEMORY IS SIZED TO DETERMINE THE HIGHEST POSSIBLE MEMORY ADDRESS. LOCATION 4 IS SET TO POINT TO "GOTCOR" WHEN A BUS TIMEOUT OCCURS (NON-EX MEM). WHEN THE PROGRAM EXITS THE SIZER ROUTINE R4 CONTAINS AN ADDRESS TWO GREATER THAN THE HIGHEST POSSIBLE MEMORY ADDRESS.

GOTCOR: THE PROGRAM NOW BACKSPACES ONE FILE, SPACES FORWARD ONE BLOCK TO SKIP THE HEADER (32 BYTES), AND THEN LOADS THE ENTIRE TALDR PROGRAM (6 BLOCKS - 768 BYTES) INTO CORE STARTING AT LOCATION 0. THE FIRST BLOCK WHICH HAD BEEN READ IN BY THE BOOTSTRAP LOADER GETS OVERLAYED. IF ANY ERRORS OCCUR WHILE READING, THE PROGRAM HALTS AT LOC 000172. IF NO ERRORS OCCUR THE PROGRAM TESTS THE SR AND IF IT IS EQUAL TO 000000, HALTS AFTER LOADING; OTHERWISE THE PROGRAM FALLS THROUGH TO "RELOC".

RELOC: THIS ROUTINE RELOCATES THE TALDR CODE TO THE TOP OF MEMORY STARTING AT "PSTART". AFTER RELOCATION CONTROL IS TRANSFERRED TO "PSTART" AT LOC XX6714. THE PROGRAM THEN HALTS AT LOC XX7014 TO AWAIT OPERATOR INPUT.

WAITA: SUBROUTINE CALLED TO ALLOW WAITING FOR THE TAPE CASSETTE TO TERMINATE ITS COMMANDS AND ALSO TO TEST FOR CASSETTE ERRORS.

ERRORA: PROGRAM BRANCHES TO A HALT HERE IF IT DETECTS ANY CASSETTE ERRORS. INCLUDES BRANCH TO "GO" TO RESTART AFTER ERRORS WHEN CONTINUE IS DE-PRESSED.

5.2.2 TALDR PROPER - PROGRAM SEQUENCE

PSTART: START OF CODE THAT SETS UP POINTERS IN ORDER TO MAKE TALDR COMPLETELY POSITION INDEPENDENT.

START: THIS CODE USES POINTERS SET UP BY PSTART, IN PREPARATION TO LOADING A PROGRAM. IT THEN HALTS AT LOC XX7014 INDICATING READINESS TO LOAD A PROGRAM.

GETNUM: START OF CODE TO GET FILE NUMBER OF PROGRAM TO BE LOADED. IT READS THE FILE NUMBER FROM THE SR. IF THE NUMBER IS LESS THAN TWO (2), AN ERROR HALT OCCURS, AS THE 1ST FILE ON THE CASSETTE (TALDR ITSELF) IS NOT LOADABLE VIA TALDR.

GETFIL: CODE TO GET TO DESIRED PROGRAM FILE. THE SEQUENCE IS:

A. REWIND THE CASSETTE
B. SKIP THE NUMBER OF FILES REQUIRED TO GET TO THE FILE TO BE LOADED. AFTER A FILE IS SKIPPED, THE HEADER FOR THE NEXT FILE IS READ TO VERIFY THAT THE END OF WRITTEN TAPE HAS NOT BEEN REACHED (SENTINEL FILE). IF END OF TAPE IS REACHED, AN ERROR HALT OCCURS, INDICATING THAT AN INVALID FILE NUMBER HAS BEEN SPECIFIED.

LOADFL: THIS CODE IS THE ACTUAL PROGRAM LOADER. SEQUENCE FOLLOWS:

A. LOAD READ BUFFER ("BUF") WITH ONE BLOCK OF DATA (JSR PC,GTDATA).

B. CLEAR THE CHECKSUM WORD.

C. GET A BYTE OF DATA UNTIL ONE WITH VALUE OF 1 IS FOUND.
D. GET ANOTHER BYTE. IT MUST BE 0. IF NOT GO TO B. IF YES, BLOCK START HAS BEEN DETECTED.

E. GET THE BYTE COUNT. (NUMBER OF BYTES IN BLOCK). 1 WORD.

F. PUT BYTE COUNT IN R4. DEDUCT 4 FROM IT TO ACCOUNT FOR 2 BYTES OF BLOCK START, AND 2 BYTES OF BYTE COUNT.

G. SEE IF REMAINDER IS 2. IF 2, THE BLOCK IS ADDRESS TRANSFER BLOCK. GO TO STEP N. IF NOT, IT IS A DATA BLOCK. FALL THROUGH.

H. GET THE LOAD ADDRESS. 1WORD. LOAD ADDRESS IS THE ADDRESS WHERE DATA IN BLOCK IS TO BE STORED. LOAD ADDRESS IS IN R5.

I. NOW GET A DATA BYTE. BRANCH IF BYTE COUNT COUNT IS NOT YET 0. GO TO STEP L.

J. BYTE COUNT IS 0. TEST THE CHECKSUM. MUST BE 0. IF 0, GO TO STEP B. HALT IF NOT. CHECKSUM ERROR.

(5.2.2 CONT'D)

K. UPON CONTINUE, REVERSE TO START OF FILE, SKIP THE HEADER, AND GO TO STEP A TO TRY AGAIN.

L. CHECK THAT ADDRESS WHERE DATA IS TO BE STORED HAS NOT INFRINGED ON TALDR CORE SPACE. IF YES, HALT. UPON CONTINUE, GO TO "START"

M. STORE DATA BYTE. GO TO STEP I.

N. GET THE TRANSFER ADDRESS. (SAVED IN R5).

O. GET ANOTHER BYTE. TEST THE CHECKSUM. IF NOT 0, GO TO J.

P. SET A 5 IN LOC 41, INDICATING THAT LOAD MEDIUM IS CASSETTE.

Q. SEE IF TRANSFER ADDRESS (IN R5) IS ODD. IF ODD, THE PROGRAM IS NOT SELF-STARTING. GO TO "START". IF EVEN, PUT R5 IN PC TO START PROGRAM JUST LOADED.

RDFRM: GET NEXT SEQUENTIAL BYTE OF DATA SUBROUTINE. THE BYTE IS IS RETURNED IN R2. TAKEN FROM READ BUFFER.

RD2FRM: THIS SUBROUTINE ASSEMBLES 2 BYTES OF DATA INTO ONE WORD. THE THE WORD IS RETURNED IN R5.

RDHDR: READ FILE HEADER SUBROUTINE. A FILE HEADER CONSISTS OF A DATA BLOCK OF 32 BYTES. THE HEADER CONTAINS INFORMATION IDENTIFYING THE FILE, SUCH AS NAME, DATE, BLOCK SIZE, ETC. FOR PURPOSES OF TALDR, THE HEADER IS CHECKED ONLY TO INSURE THAT THE END OF TAPE HAS NOT BEEN REACHED (SENTINEL FILE; ALL ZEROES).

GTDATA: SUBROUTINE TO LOAD BUFFER WITH ONE BLOCK'S WORTH OF DATA (128 BYTES). FILE DATA IS STORED IN BLOCKS OF 128 BYTES. GTDATA READS 128 BYTES OF DATA INTO A READ BUFFER CALLED "BUF".

WAIT: WAIT SUBROUTINE. AWAITS COMPLETION OF PREVIOUSLY ISSUED CASSETTE FUNCTION AND CHECKS FOR ERRORS. IF AN ERROR OCCURS, A TA11 HARDWARE ERROR OCCURS.

APPENDIX A

TA11 BOOTSTRAP LOADER

```
                                        .ABS
                                        .=173300
            173300
            000000                      R0=%0
            000001                      R1=%1
            000002                      R2=%2
            000003                      R3=%3
            000007                      PC=%7

773300  012700  177500          CBOOT:  MOV #177500,R0  ;R0 HOLDS ADDRESS OF TA11
773304  005010                          CLR (R0)        ;SELECT UNIT ZERO
773306  010701                  RESTRT: MOV PC,R1       ;USE FOR PIC
773310  062701  000052                  ADD #TABLE-.,R1 ;R1 HOLDS ADDRESS OF COMMAND TABLE
773314  012702  000375                  MOV #375,R2     ;MEMORY POINTER AND DATA FLAG
773320  112103                          MOVB (R1)+,R3   ;MOVE TEST BITS TO R3

773322  112110                  LOOP1:  MOVB (R1)+,(R0) ;MOVE COMMAND FROM TABLE TO TA11
773324  100413                          BMI DONE        ;IF COMMAND IS NEGATIVE, THEN QUIT
773326  130310                  LOOP2:  BITB R3,(R0)    ;TEST READY AND TRANSFER REQUEST BITS IN TACS
773330  001776                          BEQ LOOP2       ;BRANCH IF BITS ARE NOT SET
773332  105202                          INCB R2         ;ADVANCE MEMORY POINTER
773334  100772                          BMI LOOP1       ;IF MINUS, TRY ANOTHER TABLE COMMAND
773336  116012  000002                  MOVB 2(R0),(R2) ;READ DATA INTO MEMORY
773342  120337  000000                  CMPB R3,@#0     ;FIRST BYTE READ SHOULD BE '240'
773346  001767                          BEQ LOOP2       ;IF EQUAL, GO READ ANOTHER BYTE

773350  000000                  STOP:   HALT            ;HALT ON ERROR
773352  000755                          BR RESTRT       ;RESTART ON CONTINUE

773354  005710                  DONE:   TST (R0)        ;CHECK FOR ERROR
773356  100774                          BMI STOP        ;BRANCH TO HALT ON ERROR
773360  005007                          CLR PC          ;JUMP TO 0

773362                          TABLE:                  ;HIGH BYTE          LOW BYTE
773362  017640                  .WORD   037*400 + 240   ;ILBS+REWIND+GO     READY+TRANSFER REQUEST
773364  002415                  .WORD   005*400 + 015   ;READ +GO           SPACE FORWARD BLOCK+GO
773366  112024                  .WORD   224*400 + 024   ;READ+ILBS+END TABLE  READ+ILBS
773370  000000  000000  **      .WORD   0,0             ;TWO WORDS OF FILLER

773374  173300          **      VECTOR: CBOOT           ;POWER-UP VECTOR (PC)
773376  000340          **              000340          ;POWER-UP STATUS (PS)
```

** NOT INCLUDED IN THE SOFTWARE VERSION

APPENDIX B

TA11 BOOTSTRAP LOADER - TOGGLE-IN LISTING

ALTHOUGH THE TOGGLE-IN BOOT IS POSITION INDEPENDENT, IT IS RECOMMENDED THAT IT BE STORED STARTING AL LOCATION XX7706

LOCATION	CONTENTS
XX7706	012700
XX7710	177500
XX7712	005010
XX7714	010701
XX7716	062701
XX7720	000052
XX7722	012702
XX7724	000375
XX7726	112103
XX7730	112110
XX7732	100413
XX7734	130310
XX7736	001776
XX7740	105202
XX7742	100772
XX7744	116012
XX7746	000002
XX7750	120337
XX7752	000000
XX7754	001767
XX7756	000000
XX7760	000755
XX7762	005710
XX7764	100774
XX7766	005007
XX7770	017640
XX7772	002415
XX7774	112024

```
                        .TITLE  DZTAF-C TALDR TA11 DIAGNOSTIC ABS LOADER
                ;
                        .ABS
                                        .LIST   ME,BIN,SEQ
                                        .NLIST  MD,TOC,MC

            177570              SR      =177570
            000000              R0      =%0
            000001              R1      =%1
            000002              R2      =%2
            000003              R3      =%3
            000004              R4      =%4
            000005              R5      =%5
            000006              R6      =%6
            000006              SP      =%6
            000007              PC      =%7
                                ;
            000005              READ    =5              ;TACS READ FUNCTION
            000020              ILBS    =20             ;TACS INIT LAST BYTE SEQUENCE CODE.
            000007              SRF     =7              ;TACS SPACE REV FILE AND GO
            000035              SFB     =35             ;TACS SPACE FWD BLOCK AND GO
            000033              SFF     =33             ;TACS SPACE FORWARD FILE AND GO.
            000031              SRB     =31             ;TACS SPACE REV BLOCK.
            000037              RWD     =37             ;TACS REWIND AND GO.
            001064              MOVCNT  =PRGEND-PSTART
            001400              BYTCNT  =PRGEND+200&177600
            000270              SVR0    =STRT2+2
            000274              SVUNIT  =STRT3+2
            000364              CHKSUM  =RX20+2
            000544              CHRCNT  =RDFRAM+2
            000636              BUFADR  =GTDATB+2
            000260              SPADR   =START+2
            000264              WAITAD  =STRT1+2
```

```
                                        .SBTTL  PRE-LOADER SECTION OF TALDR (128 BYTES)
000000  000240                          240                     ;NOP FOR CBOOT.
000002  000402                          BR      GO
000004  000036  000000                  .WORD   GOTCOR,0
000010  005767  177554          GO:     TST     SR              ;SR = 0?
000014  001001                          BNE     GO1             ;BR IF NOT.
                                        ;***************************************************************
000016  000000                          HALT                    ;YES. POST BOOT HALT
                                        ;***************************************************************
000020  012706  001500          GO1:    MOV     #BYTCNT+100,SP
000024  005004                          CLR     R4              ;SIZE CORE.
000026  005714                  1$:     TST     (4)             ;REFERENCE.
000030  062704  020000                  ADD     #20000,R4
000034  000774                          BR      1$
000036  162704  001064          GOTCOR: SUB     #MOVCNT,R4      ;DETERMINE RELOC ADDR.
000042  012703  000156                  MOV     #WAITA,R3       ;WILL GO TO WAITA VIA R3
000046  112710  000007                  MOVB    #SRF,(0)        ;REVERSE TO START OF FILE.
000052  004713                          JSR     PC,(3)          ;WAIT.
000054  112710  000035                  MOVB    #SFB,(0)        ;SKIP 1ST BLOCK.
000060  004713                          JSR     PC,(3)          ;WAIT.
000062  005005                          CLR     R5              ;LOAD ADDRESS IS 0.
000064  012702  001400                  MOV     #BYTCNT,R2      ;R2 = TOTAL BYTES TO READ.
000070  005001          1$:     CLR     R1              ;START READING A BLOCK.
000072  112710  000005                  MOVB    #READ,(0)       ;ISSUE READ FUNCTION.
000076  004713                          JSR     PC,(3)          ;WAIT.
000100  116025  000002          2$:     MOVB    2(0),(5)+       ;STORE BYTE.
000104  004713                          JSR     PC,(3)          ;WAIT FOR XFR REQUEST.
000106  105201                          INCB    R1              ;HAVE READ 128 BYTES?
000110  100373                          BPL     2$              ;BR IF NO.
000112  152710  000020                  BISB    #ILBS,(0)       ;START END BLOCK SEQUENCE.
000116  004713                          JSR     PC,(3)          ;WAIT.
000120  160102                          SUB     R1,R2           ;ALL BYTES READ?
000122  001362                          BNE     1$              ;BR IF NOT TO READ NEXT BLOCK.
000124  005767  177440                  TST     SR              ;SR 0?
000130  001001                          BNE     RELOC           ;BR IF NOT. SKIP DIAGNOSTIC HALT.
                                        ;***************************************************************
000132  000000                          HALT                    ;PRE-RELOCATION HALT
                                        ;***************************************************************
000134  012701  000176          RELOC:  MOV     #PSTART,R1      ;RELOCATE CODE STARTING AT "PSTART"
000140  010402                          MOV     R4,R2           ;
000142  012703  001064                  MOV     #MOVCNT,R3      ;
000146  112122          1$:     MOVB    (1)+,(2)+       ;RELOCATE A BYTE.
000150  005303                          DEC     R3              ;DONE?
000152  001375                          BNE     1$              ;BR IF NOT.
000154  010407                          MOV     R4,PC           ;GET STARTED.
000156  032710  000240          WAITA:  BIT     #240,(0)        ;CHECK FOR ERROR/XFR REQUEST
000162  001775                          BEQ     WAITA           ;
000164  005710                          TST     (0)             ;CHECK FOR ERROR.
000166  100401                          BMI     .+4             ;BR IF ERROR.
000170  000207                          RTS     PC              ;OK. EXIT.
                                        ;***************************************************************
000172  000000                          HALT                    ;HARDWARE ERROR
                                        ;***************************************************************
000174  000711                          BR      GO1             ;GO TRY AGAIN.
```

```
                                .SBTTL  TALDR LOADER PROPER
                        ;THIS CODE IS RELOCATED TO TOP OF CORE.
000176  010703          PSTART: MOV     PC,R3           ;SAVE ADDR OF "START"
000200  062703  000056          ADD     #START-.,R3
000204  010367  000540          MOV     R3,LIMIT
000210  010703                  MOV     PC,R3           ;SAVE ADDR OF "WAIT" ROUTINE.
000212  062703  000512          ADD     #WAIT-.,R3
000216  010367  000042          MOV     R3,WAITAD
000222  010703                  MOV     PC,R3           ;COMPUTE AND SAVE STACK ADDR.
000224  062703  000744          ADD     #STACK-.,R3
000230  010367  000024          MOV     R3,SPADR
000234  010703                  MOV     PC,R3           ;COMPUTE AND SAVE "BUFADR"
000236  062703  000516          ADD     #BUF-.,R3
000242  010367  000370          MOV     R3,BUFADR
000246  010067  000016          MOV     R0,SVR0         ;SAVE TA11 ADDR.
000252  011067  000016          MOV     (0),SVUNIT      ;SAVE UNIT NUMBER.


                        ;START OF CODE THAT MUST REMAIN INTACT WHILE LOADING A PROGRAM
000256  012706  000000  START:  MOV     #0,SP           ;SET UP STACK. (#0 IS SPADR)
000262  012703  000000  STRT1:  MOV     #0,R3           ;SET UP WAIT ROUTINE ADDR. (#0 IS WAITAD)
000266  012700  000000  STRT2:  MOV     #0,R0           ;RESTORE TA11 ADDR.(#0 IS SVR0)
000272  012710  000000  STRT3:  MOV     #0,(0)          ;RESTORE UNIT NUMBER. (#0 IS SVUNIT).
                                ;*****************************************************************
000276  000000                  HALT                    ;GOOD LOAD HALT. READY TO LOAD AGAIN
                                ;*****************************************************************


                        ;GET FILE # OF PROGRAM TO BE LOADED.
000300  013704  177570  GETNUM: MOV     @#SR,R4         ;SR TO R4.
000304  005304                  DEC     R4              ;SR-1
000306  003002                  BGT     GETFIL          ;BR IF GREATER THAN 0.
                                ;*****************************************************************
000310  000000                  HALT                    ;ERROR! FIRST FILE IS NOT LOADABLE!!!!
                                ;*****************************************************************
000312  000761                  BR      START           ;TRY AGAIN.


                        ;GET TO REQUIRED FILE
000314  112710  000037  GETFIL: MOVB    #RWD,(0)        ;REWIND CASSETTE.
000320  004713                  JSR     PC,(3)          ;WAIT.
000322  112710  000033  1$:     MOVB    #SFF,(0)        ;ISSUE SKIP FILE FWD FUNCTION.
000326  004713                  JSR     PC,(3)          ;WAIT.
000330  004767  000250          JSR     PC,RDHDR        ;READ HEADER.
000334  005777  000276          TST     @BUFADR         ;1ST WORD MUST NOT BE 0.
000340  001002                  BNE     2$              ;BR IF IT IS NOT.
                                ;*****************************************************************
000342  000000                  HALT                    ;SENTINEL FILE FOUND! INVALID FILE #!!
                                ;*****************************************************************
000344  000744                  BR      START
000346  005304          2$:     DEC     R4              ;DONE SKIPPING?
000350  001364                  BNE     1$              ;BR IF NOT.
```

```
                                        ;LOAD THE FILE.
000352  005067  000374          LOADFL: CLR     BLKCNT          ;CLEAR BLOCK COUNTER.
000356  004767  000232                  JSR     PC,GTDATA       ;READ FIRST DATA BLOCK.
000362  005027  000000          RX20:   CLR     #0              ;CLEAR THE CHECKSUM. (#0 IS CHKSUM)
000366  004767  000150                  JSR     PC,RDFRAM       ;GET A BYTE.
000372  005302                          DEC     R2              ;BYTE IN R2. IS IS A SYNC? (1).
000374  001372                          BNE     RX20            ;BR IF NOT.
000376  004767  000140                  JSR     PC,RDFRAM       ;NEXT BYTE MUST BE 0.
000402  105702                          TSTB    R2              ;IS IT 0?
000404  001366                          BNE     RX20            ;BR IF NOT. NOT A BLOCK START.
000406  004767  000150                  JSR     PC,RD2FRM       ;OK.GET THE BYTE COUNT (WORD).
000412  010504                          MOV     R5,R4           ;PUT IT IN R4.
000414  162704  000004                  SUB     #4,R4           ;DEDUCT HEADER (SYNC AND BYTE COUNT).
000420  022704  000002                  CMP     #2,R4           ;IF 2, IT'S XFR BLOCK.
000424  001426                          BEQ     6$              ;BR IF YES.
000426  004767  000130                  JSR     PC,RD2FRM       ;GET LOAD ADDR.
000432  004767  000104          2$:     JSR     PC,RDFRAM       ;GET A BYTE.
000436  100012                          BPL     4$              ;BR IF BYTE COUNT STILL NOT 0.
000440  105767  177720                  TSTB    CHKSUM          ;CHECKSUM OK?
000444  001746                          BEQ     RX20            ;BR IF YES.
000446                          3$:
                                        ;*****************************************************************
000446  000000                          HALT                    ;NO. CHECKSUM ERROR.
                                        ;*****************************************************************
000450  112710  000007                  MOVB    #SRF,(0)        ;ISSUE REVERSE FILE FUNCTION
000454  004713                          JSR     PC,(3)          ;WAIT
000456  004767  000122                  JSR     PC,RDHDR        ;SKIP HEADER.
000462  000733                          BR      LOADFL          ;TRY AGAIN.
000464  020567  000260          4$:     CMP     R5,LIMIT        ;REACHED LOADER START?
000470  103402                          BLO     5$              ;BR IF NOT.
                                        ;*****************************************************************
000472  000000                          HALT                    ;TALDR LOADER OVERRUN
                                        ;*****************************************************************
000474  000670                          BR      START
000476  110225                  5$:     MOVB    R2,(5)+         ;STORE BYTE.
000500  000754                          BR      2$
000502  004767  000054          6$:     JSR     PC,RD2FRM       ;GET TRANSFER ADDR.
000506  004767  000030                  JSR     PC,RDFRAM       ;CHECK FOR CORRECT CHECKSUM.
000512  105767  177646                  TSTB    CHKSUM          ;IS IT?
000516  001353                          BNE     3$              ;BR IF NOT.
000520  112737  000005  000041          MOVB    #5,@#41         ;SET CASSETTE LOAD INDICATION.
000526  006005                          ROR     R5              ;CHECK FOR ODD XFR ADDR.
000530  103652                          BCS     START           ;BR IF ODD ADDR.
000532  006305                          ASL     R5              ;RESTORE XFR ADDR.
000534  010507                          MOV     R5,PC           ;GO SELF START PROGRAM.
```

```
                                        ;READ FRAME SUBROUTINE.
000536  004767  000052          RDFRMA: JSR     PC,GTDATA       ;FILL UP BUFFER.
000542  005327  000000          RDFRAM: DEC     #0              ;BUF EMPTY? (#0 IS CHRCNT)
000546  100773                  BMI     RDFRMA          ;BR IF YES.
000550  112102                  MOVB    (1)+,R2         ;GET A BYTE FROM BUFFER.
000552  060267  177606          ADD     R2,CHKSUM       ;ADDIT TO CHECKSUM.
000556  005304                  DEC     R4              ;DECREMENT BYTE COUNT.
000560  000207                  RTS     PC              ;EXIT.
                                        ;READ 2 FRAMES SUB. (WORD).
000562  004767  177754          RD2FRM: JSR     PC,RDFRAM       ;GET A BYTE.
000566  010246                  MOV     R2,-(6)         ;SAVE FRAME IN STACK.
000570  004767  177746          JSR     PC,RDFRAM       ;GET ANOTHER.
000574  110266  000001          MOVB    R2,1(6)         ;MAKE A WORD.
000600  012605                  MOV     (6)+,R5         ;PUT IT IN R5
000602  000207                  RTS     PC              ;EXIT.
                                        ;GET DATA ROUTINE. FILLS BUFFER WITH ONE BLOCKS WORTH.
000604  012767  000040  177732  RDHDR:  MOV     #32.,CHRCNT     ;WILL READ 32 BYTES.
000612  000410                  BR      GTDATB
000614  005267  000132          GTDATA: INC     BLKCNT          ;INCREMENT BLOCK COUNT.
000620  042767  177774  000124          BIC     #177774,BLKCNT  ;WANT 2 LEAST SIGNIFICANT DIGITS.
000626  012767  000200  177710          MOV     #128.,CHRCNT    ;WILL READ 128 BYTES.
000634  012701  000000          GTDATB: MOV     #0,R1           ;BUF START ADDR TO R1. (#0 IS BUFADR)
000640  112710  000005                  MOVB    #READ,(0)       ;ISSUE READ FUNCTION.
000644  004713                  1$:     JSR     PC,(3)          ;WAIT.
000646  116021  000002                  MOVB    2(0),(1)+       ;PUT BYTE IN BUFFER.
000652  004713                          JSR     PC,(3)          ;WAIT FOR XFR REQUEST.
000654  005367  177664                  DEC     CHRCNT          ;DONE 128?
000660  001372                          BNE     1$              ;BR IF NOT.
000662  152710  000020                  BISB    #ILBS,(0)       ;DO END BLOCK STUFF.
000666  004713                          JSR     PC,(3)          ;WAIT.
000670  016701  177742                  MOV     BUFADR,R1       ;BUFFER ADDR TO R1.
000674  012767  000200  177642          MOV     #128.,CHRCNT
000702  022767  000001  000042          CMP     #1,BLKCNT       ;COUNT =1?
000710  001004                          BNE     2$              ;BR IF NOT.
000712  005721                          TST     (1)+            ;FIRST WORD OF BUFFER IS NOT DATA.
000714  162767  000002  177622          SUB     #2,CHRCNT       ;DEDUCT 2 FROM DATA COUNT.
000722  000207                  2$:     RTS     PC              ;EXIT.
                                        ;WAIT ROUTINE
000724  105710                  WAIT:   TSTB    (0)             ;XFR REQUEST?
000726  100405                          BMI     1$              ;BR IF YES.
000730  032710  000040                  BIT     #40,(0)         ;READY?
000734  001773                          BEQ     WAIT            ;BR IF NOT
000736  005710                          TST     (0)             ;YES. ERROR SET?
000740  100401                          BMI     .+4             ;BR IF ERROR.
000742  000207                  1$:     RTS     PC              ;EXIT.
                                        ;*****************************************************************
000744  000000                          HALT                    ;HARDWARE ERROR HALT.
                                        ;*****************************************************************
000746  011707                  REPEAT: MOV     (7),PC          ;GO TO "START".
000750  000000                  LIMIT:  0                       ;CONTAINS ADDR OF "START"
000752  000000                  BLKCNT: 0                       ;BLOCK COUNTER.
000754  000100                  BUF:    .BLKW   64.             ;READ BUFFER.
001154  000006                          .BLKW   6               ;STACK.
001170                          STACK:
```

```
                                        .SBTTL  CBOOT COPY
                                ;
                        000000  R0=      %0
                        000001  R1=      %1
                        000002  R2=      %2
                        000003  R3=      %3
                        000007  PC=      %7
                        177500  TACS=    177500          ;TA-11 CONTROL AND STATUS REG.
                                ;

001170  012700  177500  CBOOT:  MOV     #TACS,R0
001174  005010                  CLR     (R0)            ;SELECT UNIT #0
001176  010701          RESTRT: MOV     PC,R1           ;USE FOR PIC
001200  062701  000052          ADD     #TABLE-.,R1     ;R1 HOLDS ADDR. OF COMMAND TABLE
001204  012702  000375          MOV     #375,R2         ;MEMORY PTR. AND DATA FLAG
001210  112103                  MOVB    (R1)+,R3        ;TEST BITS
                                ;
001212  112110          LOOP1:  MOVB    (R1)+,(R0)      ;COMMAND FROM TABLE TO TACS
001214  100413                  BMI     DONE            ;WHEN COMMAND CODE NEG., QUIT
001216  130310          LOOP2:  BITB    R3,(R0)         ;TEST READY AND T-REQ. BITS IN TACS
001220  001776                  BEQ     LOOP2           ;LOOP 'TILL SOMETHING COMES UP
001222  105202                  INCB    R2              ;ADVANCE MEMORY PTR.
001224  100772                  BMI     LOOP1           ;IF MINUS, TRY NEXT COMMAND
001226  116012  000002          MOVB    2(R0),(R2)      ;READ DATA INTO MEMORY
001232  120337  000000          CMPB    R3,@#0          ;FIRST BYTE READ SHOULD BE '240'
001236  001767                  BEQ     LOOP2           ;IF O.K., GO READ ANOTHER BYTE.
001240  000000          STOP:   HALT                    ;HALT ON ERROR
001242  000755                  BR      RESTRT          ;RESTART ON CONTINUE
                                ;
001244  005710          DONE:   TST     (R0)            ;CHECK FOR ERROR
001246  100774                  BMI     STOP            ;HALT ON ERROR
001250  005007                  CLR     PC              ;= 'JMP @#0'
                                ;
001252  017640          TABLE:  .WORD   37*400+240      ;READY+T-REQ./ILBS+READY+GO
001254  002415                  .WORD   5*400+15        ;SFB+GO/READ+GO
001256  112024                  .WORD   224*400+24      ;READ+ILBS/READ+ILBS+E.O.TABLE


                        ;RESTART CODE. NOT PART OF CBOOT CODE ABOVE.
001260  000632                  BR      REPEAT          ;GO TO ABS LOADER.
001262          PRGEND:
                                .END
```

```
BLKCNT  000752            939*   1002*  1003*  1016   1034#
BUF     000754            900    1035#
BUFADR= 000636            830#    901*   930   1014
BYTCNT= 001400            825#    843    855
CBOOT   001170           1049#
CHKSUM= 000364            828#    956    976    989*
CHRCNT= 000544            829*   1000*  1004*  1010*  1015*  1019*
DONE    001244           1057    1068#
GETFIL  000314            918     925#
GETNUM  000300            916#
GO      000010            836     838#
GOTCOR  000036            837     848#
GO1     000020            839     843#   887
GTDATA  000614            940     985   1002#
GTDATB  000634            830    1001   1005#
ILBS  = 000020            818#    863   1012
LIMIT   000750            892*    966   1033#
LOADFL  000352            939#    965
LOOP1   001212           1056#   1061
LOOP2   001216           1058#   1059   1064
MOVCNT= 001064            824#    848    874
PC    =%000007            815#    851*   853*   858*   860*   864*   878*   883*   890    893    896    899    926*
                          928*    929*   940*   942*   945*   948*   953*   954*   963*   964*   974*   975*   982*
                          985*    991*   993*   995*   998*  1007*  1009*  1013*  1020*  1028*  1032*  1044#  1051
                         1070*
PRGEND  001262            824     825   1078#
PSTART  000176            824     872    890#
RDFRAM  000542            829     942    945    954    975    986#   993    995
RDFRMA  000536            985#    987
RDHDR   000604            929     964   1000#
RD2FRM  000562            948     953    974    993#
READ  = 000005            817#    857   1006
RELOC   000134            868     872#
REPEAT  000746           1032#   1077
RESTRT  001176           1051#   1066
RWD   = 000037            823#    925
RX20    000362            828     941#   944    947    957
R0    =%000000            807#    902    909*  1040#  1049*  1050*  1056*  1058   1062   1068
R1    =%000001            808#    856*   861*   865    872*  1005*  1014*  1041#  1051*  1052*  1054   1056
R2    =%000002            809#    855*   865*   873*   943*   946    972    988*   989    994    996   1042#  1053*
                         1060*   1062*
R3    =%000003            810#    849*   874*   876*   890*   891*   892    893*   894*   895    896*   897*   898
                          899*    900*   901    908*  1043#  1054*  1058   1063
R4    =%000004            811#    844*   846*   848*   873    878    916*   917*   936*   949*   950*   951    990*
R5    =%000005            812#    854*   949    966    979*   981*   982    997*
R6    =%000006            813#
SFB   = 000035            820#    852
SFF   = 000033            821#    927
SP    =%000006            814#    843*   907*
SPADR = 000260            831#    898*
SR    = 177570            806#    838    867    916
SRB   = 000031            822#
SRF   = 000007            819#    850    962
STACK   001170            897    1037#
START   000256            831     891    907#   922    935    971    980
STOP    001240           1065#   1069
```

```
STRT1    000262            832     908*
STRT2    000266            826     909*
STRT3    000272            827     910*
SVRO   = 000270            826*    902*
SVUNIT= 000274             827*    903*
TABLE    001252           1052    1072*
TACS   = 177500           1045*   1049
WAIT     000724            894    1022*   1025
WAITA    000156            849     879*    880
WAITAD= 000264             832*    895*
.      = 001262            882     891     894     897     900    1027    1035*   1036*   1052
```