KDJ11-B

EEPROM USENG LANG LDR
COEEHA0

AH-FF31A-MC
1 OF 1 JUL 1985

digital
MADE IN USA

```
  W
A ::                                                                    SEQ 000
1
COEEHA EEPROM USENG LANG LDR    MACRO Y05.02  Wednesday 20-Feb-85 16:23  Page 1


                                                                .REM &

                                        IDENTIFICATION
                                        --------------

                                PRODUCT CODE:  AC-FF30A-MC

                                PRODUCT NAME:  COEEHAO EEPROM USENG LANG LDR

                                PRODUCT DATE:  FEBRUARY, 1985

                                MAINTAINER:    DIAGNOSTIC ENGINEERING




                                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 DOCUMENT.

                                NO RESPONSIBILITY IS ASSUMED  FOR  THE  USE  OR  RELIABILITY  OF
                                SOFTWARE ON EQUIPMENT THAT IS NOT  SUPPLIED  BY  DIGITAL  OR ITS
                                AFFILIATED COMPANIES.

                                COPYRIGHT (C) 1985 BY DIGITAL EQUIPMENT CORPORATION


                        THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

                                DIGITAL         PDP             UNIBUS          MASSBUS
                                DEC             DECUS           DECTAPE
```

TABLE OF CONTENTS

1. PROGRAM ABSTRACT

The KDJ11-B is a PDP-11 CPU that incorporates the J11 chip set as the heart of the processor. It is a quad height Q22 bus module. The KDJ11-B has two on-board ROM's. One of them, the 16-bit addressable ROM, contains the self-test and the boot codes. The other ROM, the 8-bit addressable one, contains the base area with hardware selection parameters, optional bootstraps, optional UFD (User Friendly Diagnostic) system desctiption area, and optional foreign language text.

On units to be shipped to non-English speaking countries, a dummy or "null" language is loaded into the EEPROM. The purpose of this is to disable English language error messages when the system is first installed. If and when the system passes its internal self tests, the user will be instructed to run a UFD (User Friendly Diagnostics) package which will be part of a "country kit" for each seperate language. The UFD package will use the native language for the particular country and, in addition, will load diagnostic and error messages in the native language into the EEPROM, so each subsequent power-up or reboot will have diagnostic and error messages in the user's own language.

The purpose of this program is to select the diagnostic/boot ROM messages rather than any EEPROM language area-based messages. It will delete any language in the EEPROM to free up space but will leave the UFD (user friendly diagnostics) area intact. If the program detects an error while modifying the EEPROM setup area, the program will attempt to restore the setup area to its previous state. If the program detects an error while updating the UFD area in the EEPROM, a warning message will be issued, stating that the UFD area has been corrupted.

2. SYSTEM REQUIREMENTS

Hardware Requirements

To run successfully this utility needs:

1. KDJ11-B CPU module
2. console terminal
3. at least 28K of memory

3. LOADING AND STARTING PROCEDURES

To start-up this program:

1. Boot XXDP+

2. Type "R NAME", where NAME is the name of the BIN or BIC file for this program.

The starting address of the program is 1000.
Note: if trying to restart the program in an arbitrary place after

```
                        HALT on Break the following registers should be set up:
                        17777572=0      to disable memmory management
                        17777520=1000   to clear diagnostic mode (bit 8), but still save
                                        HALT on Break
                        17777746=400    to flush the cache
```

```
132                                     4.  SPECIAL ENVIRONMENTS
133
134                                     The program is not APT compatible.
135
136                                     5.  PROGRAM OPTIONS
137
138                                     None.
139
140                                     6. EXECUTION TIMES
141
142                                     The program runs in under 10 seconds.
143
144                                     7. ERROR INFORMATION
145
146                                     7.1 DEFECTIVE BYTE IN EEPROM
147
148                                     After each write, the Byte which should have been written is
149                                     compared to the Byte in the proper location, and if it is not correct,
150                                     the following error message is displayed:
151
152                                             EEPROM write error, PCR page n, address mmmmmm.
153                                             Data written qqq, data read rrr.
154
155                                     where n is the EEPROM page selected by the Page Control Register (PCR),
156                                     mmmmmm is the physical address of the bad byte in question, qqq is the
157                                     byte value that was written out to the address and rrr what was read
158                                     back in after the write. (should be identical to qqq)
159
160                                     7.2 PROCESSOR NOT KDJ11-B
161
162                                     The program checks the type of CPU it is running on, which must be a
163                                     KDJ11-B processor (MFPT returns 5 in r0).  If not, the following message
164                                     is printed:
165
166                                             Language area not supported by this processor.
167
168                                     7.3 CHECKSUM ERROR IN SETUP AREA
169
170                                     The checksum in the setup area is checked to see if it contains a valid
171                                     checksum.  Also, bytes 6 and 103 (addresses 17765022 and 17765314,
172                                     respectively) are checked to see if they contain 0 and 252 octal,
173                                     respectively.  If any of these conditions is not met, the following
174                                     message is printed:
175
176                                             EEPROM checksum error in setup area.
177
178                                     No attempt is made to correct a checksum error.
179
180                                     7.4 DIFFERENCES BETWEEN UFD "QUIET" MODE AND "STANDALONE" MODE
181
182                                     When this program is run in UFD "Quiet" mode (which will usually be
183                                     the case) none of the error messages will appear.  If no error is
184                                     detected, no messages whatsoever are printed.  If an error is detected
185                                     when writing the setup area, the program will attempt to restore the
186                                     setup area to the state it was in when the program was started. If an
187                                     error was detected when the UFD area is being moved, the following
188                                     message is printed:
```

```
189
190                                 WARNING: UFD area has been corrupted.
191
192                                 8. EXAMPLES
193
194                                 After booting XXDP+ and running the program, no message should
195                                 appear, just the XXDP dot prompt ( . )
196
197                                 If a problem occurred, one of the messages in section 7 should appear.
198
199                                 9. PROGRAM DESCRIPTION
200
201                                 This program consists of a body of code which checks certain
202                                 conditions such as correct processor type, clears the EEPROM language
203                                 bit in the EEPROM, and frees up any language area in the EEPROM.  If
204                                 there was an unsuccessful attempt to write any of the bytes in the
205                                 EEPROM, one of the message in section 7 appears.  If run under UFD
206                                 "Quiet" mode, no message is printed if the program was successful,
207                                 otherwise the messages in 7.4 appear.  In both cases, the XXDP prompt
208                                 appears.
209                                                                                    &
```

```
                                .TITLE  COEEHA EEPROM USENG LANG LDR

                                .SBTTL  PROGRAM CONSTANTS
000000                          .ENABL  ABS
                                .NLIST  MD,CND
                                .LIST   ME

        177520                  BCSR    =       177520
        177522                  PCR     =       177522
        177522                  PCRLB   =       177522
        165000                  E2PROM  =       165000
        165316                  E2PAR   =       E2PROM+316              ;E2PROM PARITY BYTE
        165006                  E2LLB   =       E2PROM+6                ;LOCAL LANGUAGE BIT IN E2PROM
        166000                  ENDE2R  =       E2PROM+1000             ;LAST ADDRESS OF E2PROM+2
        173002                  RMVTST  =       173002                  ;WORD TO TEST ROM VERSION NUMBER
        025370                  DELAY   =       11000.
        000140                  LNGHDR  =       140                     ;I.D. OF A LANGUAGE AREA
        000040                  UFDHDR  =       040                     ;I.D. OF A UFD BLOCK
        000003                  RETRY   =       3                       ;NUMBER OF ATTEMPTS TO WRITE A
                                                                        ;BYTE IN E2PROM BEFORE GIVING UP
        000004                  MAXERR  =       4                       ;NO. OF ERRORS ALLOWED IN LOCAL
                                                                        ;LANGUAGE TEXT BEFORE QUITTING
        177524                  BDR     =       177524
        000015                  CR      =       15
        000012                  LF      =       12
        000200                  BIT7    =       200
        000100                  BIT6    =       100
        000011                  tab     =       11
        000010                  backsp  =       10
        000040                  space   =       40
        000033                  esc     =       33
```

```
                                        .SBTTL  CHECK FOR CERTAIN EXCEPTIONS FIRST

                001000                  .=1000

001000  005037  177522        START:  CLR     @#PCR           ;SELECT PAGE 0 OF EEPROM
001004  013746  177520                MOV     @#BCSR,-(SP)    ;SAVE OLD BCSR VALUE
001010  112737  000060  177520        MOVB    #60,@#BCSR      ;WRITE ENABLE THE E2PROM & ENABLE ROM

001016  000007                        MFPT                    ;GET PROCESSOR TYPE
001020  020027  000005                CMP     R0,#5           ;CHECK TO SEE IF ORION
001024  001404                        BEQ     1$              ;YES - CONTINUE
001026                                .TYPMSG #FMSG2          ;FIELD-SERVICE MESSAGE
                000001                .NARG   NARGS
                000027                .NTYPE  NTYPE,#FMSG2
  001026  012700  002215                MOV   #FMSG2,R0
  001032  104003                        EMT   3
001034  000426                        BR      203$

001036  012700  165000        1$:     MOV     #E2PROM,R0      ;STARTING ADDRESS TO CHECKSUM
001042  005001                        CLR     R1              ;INITIALIZE CHECKSUM
001044  012703  000151                MOV     #105.,R3        ;NO. OF BYTES TO CKSUM
001050  012005                201$:   MOV     (R0)+,R5        ;GET A BYTE
001052  042705  177400                BIC     #177400,R5      ;NO BUS NOISE, THANK YOU.
001056  060501                        ADD     R5,R1           ;ACCUMULATE CHECKSUM
001060  077305                        SOB     R3,201$         ;CONTINUE TILL DONE
001062  105701                        TSTB    R1              ;IS CKSUM 0?
001064  001007                        BNE     202$            ;NO, ERROR
001066  105737  165022                TSTB    @#E2PROM+22     ;BYTE TO TEST FOR VALID ROM, SHOULD BE 0
001072  001004                        BNE     202$            ;NO, ERROR
001074  123727  165314  000252        CMPB    @#E2PROM+314,#252       ;BYTE TO TEST FOR VALID ROM
001102  001404                        BEQ     USETUP          ;GO TO NEXT CHECK IF OK
001104                          202$:   .TYPMSG #FMSG4          ;FIELD SERVICE MESSAGE
                000001                .NARG   NARGS
                000027                .NTYPE  NTYPE,#FMSG4
  001104  012700  002350                MOV   #FMSG4,R0
  001110  104003                        EMT   3
001112  000573                203$:   BR      EXIT            ;QUIT

                                        .SBTTL  CLEAR LANGUAGE BIT AND UPDATE CKSUM

001114  012700  165006        USETUP: MOV     #E2LLB,R0       ;ADDRESS OF LANG. BIT
001120  111005                        MOVB    (R0),R5         ;GET BYTE WHERE LANGUAGE BIT EXISTS
001122  002032                        BGE     LANG            ;IF ALREADY CLEAR, NO UPDATING IS NECESSARY
001124  042705  000200                BIC     #BIT7,R5        ;CLEAR LANGUAGE BIT
001130  004767  000460                CALL    WRBYTE          ;WRITE IT BACK
001134  001405                        BEQ     1$              ;WRITE WENT OK
001136  052705  000200                BIS     #BIT7,R5        ;PUT THINGS BACK THE WAY WE FOUND THEM
001142  004767  000446                CALL    WRBYTE          ;
001146  000555                        BR      EXIT            ;GIVE UP
001150  012700  165316        1$:     MOV     #E2PAR,R0       ;ADDRESS OF PARITY BYTE
001154  111005                        MOVB    (R0),R5         ;GET PARITY BYTE
001156  162705  000200                SUB     #BIT7,R5        ;UPDATE PARITY
001162  004767  000426                CALL    WRBYTE          ;WRITE IT BACK
001166  001410                        BEQ     LANG            ;IF ALL WENT WELL
001170  062705  000200                ADD     #BIT7,R5        ;TRY TO PUT EVERYTHING BACK
001174  004767  000414                CALL    WRBYTE          ;THE WAY WE FOUND IT
001200  152737  000200  165006        BISB    #BIT7,@#E2LLB   ;RESTORE LANGUAGE BIT
```

CLEAR LANGUAGE BIT AND UPDATE CKSUM

```
001206  000535                          BR      EXIT

                                        .SBTTL  CHECK FOR LANGUAGE

001210  012700  165776          LANG:   MOV     #ENDE2R-2,R0    ;LAST ADDRESS (CKSUM) OF E2PROM
001214  112737  000016  177522          MOVB    #16,@#PCRLB     ;SELECT LAST PAGE
001222  012701  000005                  MOV     #5,R1           ;NO. OF BYTES IN HEADER TO CHECKSUM
001226  010005                          MOV     R0,R5           ;SAVE ADDRESS
001230  005003                          CLR     R3              ;
001232  111004                  4$:     MOVB    (R0),R4         ;GET A BYTE
001234  060403                          ADD     R4,R3           ;ACCUMULATE CHECKSUM
001236  005740                          TST     -(R0)           ;CORRECT ADDRESS
001240  077104                          SOB     R1,4$           ;LOOP FOR 5 BYTES
001242  105703                          TSTB    R3              ;IF NOT ZERO, NO LANGUAGE LOADED
001244  001116                          BNE     EXIT            ;NON-EXISTANT OR CORRUPTED LANGUAGE - SKIP

001246  014504                          MOV     -(R5),R4        ;HIGH BYTE OF BYTE COUNT
001250  014546                          MOV     -(R5),-(SP)     ;LOW BYTE OF BYTE COUNT
001252  110466  000001                  MOVB    R4,1(SP)        ;SET UPPER BYTES OF SIZE
001256  042704  177437                  BIC     #177437,R4      ;EXTRACT ID CODE
001262  012601                          MOV     (SP)+,R1        ;GET SIZE BACK
001264  042701  160000                  BIC     #160000,R1      ;R1 NOW CONTAINS SIZE OF BLOCK IN BYTES
001270  062701  000005                  ADD     #5,R1           ;ADD BYTE COUNT FOR HEADER BLOCK
001274  120427  000140                  CMPB    R4,#LNGHDR      ;IS THIS A LANGUAGE HEADER?
001300  001100                          BNE     EXIT            ;NO - QUIT

                                ;IF LAST BLOCK IS NOT LANGUAGE IT IS EITHER A UFD BLOCK OR UNUSED.  IF IT IS
                                ;UFD, IT IS IN CORRECT POSITION.  IF UNUSED, IT CAN BE IGNORED. IN EITHER CASE,
                                ;NO ACTION IS NECESSARY.

001302  010146                          MOV     R1,-(SP)        ;SAVE SIZE FOR NOW
001304  062701  000005                  ADD     #5,R1           ;ADD SIZE OF (POSSIBLE) UFD HEADER
001310  004767  000460                  CALL    ROMADR          ;SET UP PCR AND R0
001314  005003                          CLR     R3              ;INITIALIZE CKSUM
001316  004767  000424                  CALL    REAROM          ;GET A BYTE
001322  004767  000420                  CALL    REAROM          ;GET A BYTE
001326  004767  000414                  CALL    REAROM          ;GET A BYTE
001332  010546                          MOV     R5,-(SP)        ;SAVE LOW BYTE OF SIZE FOR LATER
001334  004767  000406                  CALL    REAROM          ;GET A BYTE
001340  110566  000001                  MOVB    R5,1(SP)        ;SAVE HIGH BYTE OF SIZE AND ID
001344  004767  000376                  CALL    REAROM          ;GET A BYTE
001350  116600  000001                  MOVB    1(SP),R0        ;GET I.D.
001354  012601                          MOV     (SP)+,R1        ;GET SIZE
001356  105703                          TSTB    R3              ;SEE IF VALID CKSUM
001360  001005                          BNE     1$              ;NO - WE HAVE LANGUAGE ONLY.

001362  042700  177437                  BIC     #177437,R0      ;GET ID ONLY
001366  120027  000040                  CMPB    R0,#UFDHDR      ;IS THIS A UFD BLOCK?
001372  001413                          BEQ     2$              ;YES, CONTINUE.

                                ;IF WE GET HERE, WE HAVE A LANGUAGE ONLY AND NO UFD.  WE CLOBBER
                                ;THE LANGUAGE TO MAKE ROOM FOR EEPROM BOOTS, ETC.

001374  005726                  1$:     TST     (SP)+           ;CORRECT STACK
001376  112737  000016  177522          MOVB    #16,@#PCRLB     ;SELECT LAST PAGE
001404  012700  165774                  MOV     #165774,R0      ;ADDRESS OF LANGUAGE I.D.
001410  112705  000377                  MOVB    #377,R5         ;BAD I.D.
```

CHECK FOR LANGUAGE

```
                                        CALL    WRBYTE          ;GO CLOBBER LANGUAGE AREA TO FREE SPACE
                                        BR      EXIT

                                ;WE HAVE BOTH A LANGUAGE AREA AND A UFD BLOCK.  SAVE THE UFD BLOCK.

                                2$:     BIC     #160000,R1      ;GET RID OF ID
                                        ADD     #5,R1           ;SIZE OF HEADER
                                        MOV     R1,R4           ;BYTE COUNT TO MOVE
                                        MOV     R1,UFDSIZ       ;SAVE UFD SIZE
                                        ADD     (SP)+,R1        ;ADD SIZE OF LANGUAGE AREA
                                        MOV     #BUFF,R2        ;MEMORY ADDRESS TO SAVE TO
                                        CALL    MOVROM          ;SAVE UFD AREA
                                        BNE     EXIT            ;IT IS BAD, QUIT.

                                ;WRITE UFD BLOCK

                                        MOV     UFDSIZ,R1       ;GET THE LENGTH OF THE UFD
                                        BEQ     EXIT            ;NO UFD AREA - SKIP
                                        JSR     PC,ROMADR       ;COMPUTE E2PROM PAGE AND ADDR
                                        MOV     #BUFF,R2        ;ADDRESS OF BEGINNING OF UFD AREA
                                35$:    MOVB    (R2)+,R5        ;GET SOME DATA
                                        CALL    E2WRIT          ;GO WRITE IT
                                        SOB     R1,35$          ;FINISHED UFD?
                                                                ;YES-EXIT

                                EXIT:   BICB    #60,(SP)        ;BE SURE ROM IS DISABLED
                                        MOV     (SP)+,@#BCSR    ;RESTORE BCSR
                                        CLR     @#PCR           ;
                                        RTS     PC

                                E2WRIT: CALL    WRBYTE          ;WRITE THE BYTE TO E2PROM
                                        BEQ     3$              ;OK THIS TIME
                                        .FRCTYP #FMSG3          ;WARN USER OF CORRUPTED UFD AREA
                                        .NARG   NARGS
                                        .NTYPE  NTYPE,#FMSG3
                                          MOV   #FMSG3,R0
                                          EMT   44
                                        TST     (SP)+           ;CORRECT STACK
                                        BR      EXIT            ;QUIT

                                3$:     ADD     #2,R0           ;INCREMENT LOCATION
                                        CMP     R0,#ENDE2R      ;FINISHED THIS PAGE ?
                                        BNE     10$             ;NO-RETURN
                                        MOV     #E2PROM,R0      ;YES-RESET ADDRESS
                                        ADD     #2,@#PCR        ;INCREMENT PCR TO NEXT PAGE
                                10$:    RETURN

                                        .SBTTL  PROGRAM SUBROUTINES

                                ;MOVROM - MOVE BYTES FROM EEPROM TO MEMORY
                                ;ENTRY- R1 = STARTING ADDRESS IN EEPROM (# OF BYTES FROM END)
                                ;       R2 = ADDRESS OF MEMORY BUFFER
                                ;       R4 = # OF BYTES TO MOVE
                                ;EXIT   R1 - UNCHANGED
                                ;       R2 - UPDATED MEMORY ADDRESS
                                ;       R3 = (BYTE) 0 IF VALID CKSUM
                                ;       "Z" FLAG SET IF CKSUM VALID
```

PROGRAM SUBROUTINES

```
                                        ;
001566  010403          MOVROM: MOV     R4,R3           ;SAVE R4
001570  004767  000200          CALL    ROMADR          ;LOAD PCR AND R0 WITH LANGUAGE START AREA
001574  010304                  MOV     R3,R4           ;RESTORE BYTE COUNT
001576  005003                  CLR     R3              ;INIT CHECKSUM
001600  004767  000142  5$:     CALL    REAROM          ;GET A BYTE
001604  110522                  MOVB    R5,(R2)+        ;SAVE IT
001606  077404                  SOB     R4,5$           ;LOOP TILL DONE
001610  105703                  TSTB    R3              ;IS CHECKSUM GOOD?
001612  000207                  RETURN
                                        ;
001614  120510          WRBYTE: CMPB    R5,(R0)         ;IS THE NEW DATA DIFFERENT ?
001616  001452                  BEQ     10$             ;NO-DO NOT WRITE OVER
                                        ;
001620  012703  000003          MOV     #RETRY,R3
001624  010510          1$:     MOV     R5,(R0)         ;WRITE A LOCATION
001626  012704  025370          MOV     #DELAY,R4       ;11 MS WAIT
001632  077401                  SOB     R4,.            ;WASTE TIME
001634  120510                  CMPB    R5,(R0)         ;SEE IF IT TOOK
001636  001442                  BEQ     10$             ;YES, ALL OKAY
001640  077307                  SOB     R3,1$           ;IF AT FIRST YOU DON'T SUCCEED...
001642  113704  177522          MOVB    @#PCRLB,R4      ;PCR PAGE OF BAD BYTE
001646  106204                  ASRB    R4              ;CONVERT TO PAGE #
001650  062704  000060          ADD     #'0,R4          ;CONVERT TO OCTAL
001654  110467  000237          MOVB    R4,FMSG1A       ;STORE IT FOR PRINTING
001660  010046                  MOV     R0,-(SP)        ;SAVE ROM ADDRESS
001662                          .ITOA   ,#FMSG1B        ;CONVERT ROM ADDRESS TO OCTAL
        000002                  .NARG   NARGS
        000027                     .NTYPE       NTYPE,#FMSG1B
001662  012701  002132             MOV  #FMSG1B,R1
001666  104030                     EMT  30
001670                          .TYPMSG #FMSG1          ;PRINT OUT FIRST PART OF MESSAGE
        000001                  .NARG   NARGS
        000027                  .NTYPE  NTYPE,#FMSG1
001670  012700  002062            MOV   #FMSG1,R0
001674  104003                    EMT   3
001676  042705  177400          BIC     #177400,R5      ;MAKE SURE R5 IS POSITIVE AND A BYTE
001702                          .ITOA   R5,#DUMMY1      ;CONVERT TO OCTAL
        000002                  .NARG   NARGS
        000005                    .NTYPE       NTYPE,R5
001702  010500                    MOV  R5,R0
        000027                    .NTYPE       NTYPE,#DUMMY1
001704  012701  002160            MOV  #DUMMY1,R1
001710  104030                    EMT  30
001712                          .TYPMSG #FMSG1C         ;PRINT OUT LAST 3 DIGITS OF NUMBER & MESSAGE
        000001                  .NARG   NARGS
        000027                  .NTYPE  NTYPE,#FMSG1C
001712  012700  002163            MOV   #FMSG1C,R0
001716  104003                    EMT   3
001720  013600                  MOV     @(SP)+,R0       ;GET BYTE AT ROM ADDRESS
001722  042700  177400          BIC     #177400,R0      ;GET RID OF BUS NOISE
001726                          .ITOA   ,#DUMMY2        ;CONVERT TO OCTAL
        000002                  .NARG   NARGS
        000027                     .NTYPE       NTYPE,#DUMMY2
001726  012701  002203            MOV  #DUMMY2,R1
001732  104030                    EMT  30
001734                          .TYPMSG #FMSG1D         ;PRINT LOWER 3 BYTES & REST OF MESSAGE
```

PROGRAM SUBROUTINES

```
                000001                          .NARG   NARGS
                000027                          .NTYPE  NTYPE,#FMSG1D
        001734  012700  002206                    MOV   #FMSG1D,R0
        001740  104003                            EMT   3
    481 001742  000244                          CLZ                     ;COULDN'T DO IT, SET ERROR FLAG
    482 001744  000207                  10$:    RETURN
    483
    484                                 ;REAROM - READS A BYTE FROM E2PROM ADDRESS (R0)+ INTO R5. AUTOMATICLY ADJUSTS
    485                                 ;PCRLB.  UPDATES CKSUM IN R3
    486                                 ;       ENTRY - R0      ADDRESS IN ROM TO READ FROM
    487                                 ;               R3      PARTIAL CKSUM
    488                                 ;               PCRLB   CORRECT VALUE FOR BYTE TO READ
    489                                 ;       EXIT    R0      ADDRESS OF NEXT BYTE
    490                                 ;               R3      UPDATED CKSUM
    491                                 ;               R5      BYTE READ
    492                                 ;               PCRLB   CORRECT VALUE FOR NEXT BYTE
    493
    494 001746  012005                  REAROM: MOV     (R0)+,R5        ;GET A BYTE & UPDATE ADDR. BY 2
    495 001750  060503                          ADD     R5,R3           ;UPDATE CKSUM
    496 001752  020027  166000                  CMP     R0,#ENDE2R      ;SEE IF WE SHOULD SWITCH PAGES
    497 001756  001005                          BNE     10$             ;NO
    498 001760  012700  165000                  MOV     #E2PROM,R0      ;YES - GO TO START OF PAGE
    499 001764  062737  000002  177522          ADD     #2,@#PCR                ;ADVANCE A PAGE
    500 001772  000207                  10$:    RETURN
    501
    502
    503                                 ;ROMADR - CALCULATE PAGE OFFSET FROM END OF ROM GIVEN SIZE IN BYTES
    504                                 ;       ENTRY - R1      SIZE IN BYTES
    505                                 ;       EXIT -  R0      INITIAL ADDRESS FOR FIRST BYTE IN ROM
    506                                 ;               R1      SIZE IN BYTES
    507                                 ;               PCRLB   CORRECT VALUE FOR FIRST BYTE IN ROM
    508
    509 001774  010100                  ROMADR: MOV     R1,R0           ;COPY BYTE COUNT
    510 001776  010105                          MOV     R1,R5           ;SECOND COPY
    511 002000  072527  177770                  ASH     #-8.,R5         ;DIVIDE BYTE COUNT BY 256. BYTE PAGES
    512 002004  012704  000010                  MOV     #7+1,R4         ;LAST PAGE IN 2 K PART + 1
    513 002010  160504                          SUB     R5,R4           ;STARTING PAGE NUMBER
    514
    515 002012  042700  177400                  BIC     #177400,R0      ;LEAVE ONLY BITS 7:0
    516 002016  006300                          ASL     R0              ;DOUBLE VALUE
    517 002020  001003                          BNE     20$
    518 002022  012700  165000                  MOV     #E2PROM,r0      ;
    519 002026  000406                          BR      30$             ;IF 0
    520
    521 002030  005400                  20$:    NEG     R0              ;MAKE STARTING ADDRESS BITS 8:0
    522 002032  042700  177000                  BIC     #177000,R0      ;
    523 002036  052700  165000                  BIS     #E2PROM,R0      ;MAKE A E2PROM ADDRESS
    524 002042  005304                          DEC     R4              ;DECREMENT PAGE NUMBER BY 1
    525
    526 002044  006304                  30$:    ASL     R4              ;MAKE PAGE NUMBER CORRECT FOR PCR
    527 002046  110437  177522                  MOVB    R4,@#PCRLB      ;CORRECT PAGE IN PCRLB
    528 002052  000207                          RTS     PC              ;RETURN
    529
    530 002054  000000                  WERR:   0                       ;FLAG FOR BAD BYTE
    531 002056  177777                  OLDSIZ: -1                      ;>0 - SIZE IN BYTES OF OLD LANGUAGE, 0 IF NO
    532                                                                 ;LANGUAGE, -1 IF E2PROM MAY BE BAD/NONEXISTANT
    533 002060  000000                  UFDSIZ: 0                       ;SIZE IN BYTES OF OLD UFD AREA
```

PROGRAM SUBROUTINES

```
                                        .SBTTL  "FIELD SERVICE MODE" ERROR MESSAGES

                                        .ENABL LC
002062  105     105     120     FMSG1:  .ASCII  /EEPROM write error, PCR page /
002065  122     117     115
002070  040     167     162
002073  151     164     145
002076  040     145     162
002101  162     157     162
002104  054     040     120
002107  103     122     040
002112  160     141     147
002115  145     040
002117  130     054     040     FMSG1A: .ASCII  /X, address /
002122  141     144     144
002125  162     145     163
002130  163     040
002132                          FMSG1B: .BLKB   6               ;FOR ADDRESS
002140  015     012     104             .ASCIZ  <CR><LF>/Data written /
002143  141     164     141
002146  040     167     162
002151  151     164     164
002154  145     156     040
002157  000
002160                          DUMMY1: .BLKB   3               ;3 UPPER BYTES NOT TO BE PRINTED
002163                          FMSG1C: .BLKB   3
002166  054     040     104             .ASCIZ  /, Data read /
002171  141     164     141
002174  040     162     145
002177  141     144     040
002202  000
002203                          DUMMY2: .BLKB   3               ;3 UPPER BYTES NOT TO BE PRINTED
002206                          FMSG1D: .BLKB   3
002211  056                             .ASCII  /./
002212  015     012     000     CRLF:   .ASCIZ  <CR><LF>
002215  114     141     156     FMSG2:  .ASCIZ  /Language Area not supported on this processor./<CR><LF>
002220  147     165     141
002223  147     145     040
002226  101     162     145
002231  141     040     156
002234  157     164     040
002237  163     165     160
002242  160     157     162
002245  164     145     144
002250  040     157     156
002253  040     164     150
002256  151     163     040
002261  160     162     157
002264  143     145     163
002267  163     157     162
002272  056     015     012
002275  000
002276  015     012     127     FMSG3:  .ASCIZ  <CR><LF>/WARNING: UFD area has been corrupted./<CR><LF>
002301  101     122     116
002304  111     116     107
002307  072     040     125
```

"FIELD SERVICE MODE" ERROR MESSAGES

```
        002312      106      104      040
        002315      141      162      145
        002320      141      040      150
        002323      141      163      040
        002326      142      145      145
        002331      156      040      143
        002334      157      162      162
        002337      165      160      164
        002342      145      144      056
        002345      015      012      000
    551 002350      103      150      145  FMSG4:   .ASCIZ  /Checksum error in EEPROM setup area./<CR><LF>
        002353      143      153      163
        002356      165      155      040
        002361      145      162      162
        002364      157      162      040
        002367      151      156      040
        002372      105      105      120
        002375      122      117      115
        002400      040      163      145
        002403      164      165      160
        002406      040      141      162
        002411      145      141      056
        002414      015      012      000
    552 002417                            BUFF:            ;TEMPORARY SAVE AREA FOR OLD AREA
    553          001000                                .END    START
```

C2

Symbol table

```
BACKSP= 000010          DUMMY2  002203          FMSG1B  002132          MOVROM  001566          ROMADR  001774
BCSR  = 177520          ENDE2R= 166000          FMSG1C  002163          NARGS = 000001          SPACE = 000040
BDR   = 177524          ESC   = 000033          FMSG1D  002206          NTYPE = 000027          START   001000
BIT6  = 000100          EXIT    001502          FMSG2   002215          OLDSIZ  002056          TAB   = 000011
BIT7  = 000200          E2LLB = 165006          FMSG3   002276          PCR   = 177522          UFDHDR= 000040
BUFF    002417          E2PAR = 165316          FMSG4   002350          PCRLB = 177522          UFDSIZ  002060
CR    = 000015          E2PROM= 165000          LANG    001210          REAROM  001746          USETUP  001114
CRLF    002212          E2WRIT  001520          LF    = 000012          RETRY = 000003          WERR    002054
DELAY = 025370          FMSG1   002062          LNGHDR= 000140          RMVTST= 173002          WRBYTE  001614
DUMMY1  002160          FMSG1A  002117          MAXERR= 000004

. ABS.  002417    000   (RW,I,GBL,ABS,OVR)
        000000    001   (RW,I,LCL,REL,CON)
Errors detected:  0

*** Assembler statistics


Work  file  reads: 0
Work  file writes: 0
Size of work file: 8503 Words  ( 34 Pages)
Size of core pool: 19402 Words  ( 74 Pages)
Operating  system: RSX-11M/PLUS (Under VAX/VMS)

Elapsed time: 00:00:35.75
OEEHA0.BIC,COEEHA0/CR=COEEHA0
```

COEEHAO CREATED BY MACRO ON 20-FEB-85 AT 16:23 PAGE 1

SEQ 0016

SYMBOL CROSS REFERENCE CREF V02

SYMBOL	VALUE	REFERENCES								
BACKSP	= 000010	#5-239								
BCSR	= 177520	#5-218	6-292	*6-293	*6-419					
BDR	= 177524	#5-233								
BIT6	= 000100	#5-237								
BIT7	= 000200	#5-236	6-322	6-325	6-330	6-333	6-335			
BUFF	002417	6-403	6-412	#6-552						
CR	= 000015	#5-234	6-541	6-548	6-549	6-550	6-550	6-551		
CRLF	002212	#6-548								
DELAY	= 025370	#5-226	6-462							
DUMMY1	002160	6-475	6-475	#6-542						
DUMMY2	002203	6-479	6-479	#6-545						
ENDE2R	= 166000	#5-224	6-340	6-430	6-496					
ESC	= 000033	#5-241								
EXIT	001502	6-315	6-327	6-336	6-350	6-360	6-394	6-405	6-410	#6-418
		6-427								
E2LLB	= 165006	#5-223	6-319	*6-335						
E2PAR	= 165316	#5-222	6-328							
E2PROM	= 165000	#5-221	5-222	5-223	5-224	6-301	6-310	6-312	6-432	6-498
		6-518	6-523							
E2WRIT	001520	6-414	#6-423							
FMSG1	002062	6-473	6-473	#6-538						
FMSG1A	002117	*6-470	#6-539							
FMSG1B	002132	6-472	6-472	#6-540						
FMSG1C	002163	6-476	6-476	#6-543						
FMSG1D	002206	6-480	6-480	#6-546						
FMSG2	002215	6-298	6-298	#6-549						
FMSG3	002276	6-425	6-425	#6-550						
FMSG4	002350	6-314	6-314	#6-551						
LANG	001210	6-321	6-332	#6-340						
LF	= 000012	#5-235	6-541	6-548	6-549	6-550	6-550	6-551		
LNGHDR	= 000140	#5-227	6-359							
MAXERR	= 000004	#5-231								
MOVROM	001566	6-404	#6-447							
NARGS	= 000001	#6-298	6-298	#6-314	6-314	#6-425	6-425	#6-472	6-472	6-472
		#6-473	6-473	#6-475	6-475	6-475	#6-476	6-476	#6-479	6-479
		6-479	#6-480	6-480						
NTYPE	= 000027	#6-298	6-298	#6-314	6-314	#6-425	6-425	#6-472	6-472	#6-473
		6-473	#6-475	6-475	#6-475	6-475	#6-476	6-476	#6-479	6-479
		#6-480	6-480							
OLDSIZ	002056	#6-531								
PCR	= 177522	#5-219	*6-291	*6-420	*6-433	*6-499				
PCRLB	= 177522	#5-220	*6-341	*6-390	6-467	*6-527				
REAROM	001746	6-370	6-371	6-372	6-374	6-376	6-451	#6-494		
RETRY	= 000003	#5-229	6-460							
RMVTST	= 173002	#5-225								
ROMADR	001774	6-368	6-411	6-448	#6-509					
SPACE	= 000040	#5-240								
START	001000	#6-291	6-553							
TAB	= 000011	#5-238								
UFDHDR	= 000040	#5-228	6-383							
UFDSIZ	002060	*6-401	6-409	#6-533						
USETUP	001114	6-313	#6-319							

E2

SEQ 0017

SYMBOL CROSS REFERENCE CREF V02

SYMBOL	VALUE	REFERENCES						
WERR	002054	#6-530						
WRBYTE	001614	6-323	6-326	6-331	6-334	6-393	6-423	#6-457

F2

SEQ 0018

MACRO CROSS REFERENCE CREF V02

MACRO NAME	REFERENCES					
.FRCTY	#5-276	6-425				
.ITOA	#5-255	6-472	6-475	6-479		
.TYPMS	#5-244	6-298	6-314	6-473	6-476	6-480