

M9301-YH

BOOTSTRAP TERMINATOR
MD-11-DZM9A-C

EP-DZM9A-C-DL-C

APR 1977

COPYRIGHT © 1977

digital

FICHE 1 OF 1

MADE IN USA

This microfiche card contains a grid of frames, likely representing a data table or a series of small diagrams. The frames are arranged in approximately 10 rows and 3 columns. Each frame contains text and possibly small graphical elements, but the resolution is too low to read the specific content. The text appears to be organized in a structured format, possibly a table with multiple columns and rows of data points.

000000

.REPT 0

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49

IDENTIFICATION

PRODUCT CODE:	MAINDEC-11-DZM9A-C-D
PRODUCT NAME:	BOOTSTRAP/TERMINATOR (M9301, M9400)
PROGRAM DATE:	FEBRUARY 15, 1977
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

THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED UNDER A LICENSE AND MAY ONLY BE USED OR COPIED IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE.

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

COPYRIGHT (C) 1976, 1977 BY DIGITAL EQUIPMENT CORPORATION

51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91

*
* SUMMARY OF OPERATING INSTRUCTIONS *
*

THE FOLLOWING PROCEDURE CAN BE USED TO RUN THIS DIAGNOSTIC
IN A DEVICE VERIFICATION MODE. IF THE PROGRAM DOES NOT
RUN SUCCESSFULLY CONSULT THE FOLLOWING DOCUMENT FOR ASSISTANCE.

OPERATING PROCEDURE:

1. LOAD THE PROGRAM USING NORMAL PROCEDURES.
2. LOAD ADDRESS 200
3. SET SWITCH REGISTER TO SELECT THE PROPER
VERSION OF THE ROM UNDER TEST.
(SEE INSTRUCTIONS IF A SOFTWARE SWITCH REGISTER
IS TO BE USED.)
4. PRESS START
5. THE PROGRAM SHOULD TAKE ABOUT 1 SEC TO
COMPLETE THE TEST AND PRINT: "END OF TEST".
6. IF THE PROGRAM DOES NOT RUN AS DESCRIBED
ABOVE, CONSULT THE FULL OPERATING INSTRUCTIONS
WHICH FOLLOW.

* * CAUTION * *

BECAUSE THE CONTENTS READ FROM LOCATION 773024
OF THE M9301 OPTION IS CONFIGURATION DEPENDANT(SWITCH
REGISTER DEPENDANT), THIS LOCATION IS NOT INCLUDED IN
THE DATA CHECK.
THIS LOCATION CAN BE VERIFIED BY EXAMINING IT OR BY USING
THE ALTERNATE STARTING ADDRESS(SEE SECTION 2.1.3) TO
PRINT OUT THIS LOCATION.

93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146

1.0 GENERAL PROGRAM INFORMATION

1.1 PROGRAM PURPOSE

THIS DIAGNOSTIC PROGRAM IS INTENDED TO VERIFY THE ROM CONTENTS OF THE ROM BOOTSTRAP MODULES. THE PROGRAM COMPUTES AND CHECKS A CYCLIC REDUNDANCY CHARACTER AND A LONGITUDINAL PARITY CHARACTER FOR THE CONTENTS OF THE ROM STORAGE AVAILABLE IN AN M9301 OR M9400 MODULE.

A SEPARATE ROUTINE INCLUDED ALLOWS THE USER TO TYPE THE CONTENTS OF THE ROM STORAGE ON THE TELETYPE AS AN AID TO DEBUGGING.

1.2 SYSTEM REQUIREMENTS

1.2.1 HARDWARE

PDP/11 PROCESSOR
TELETYPE OR EQUIVALENT
4K OF MEMORY
M9301 OR M9400 MODULE

1.2.2 SOFTWARE

THIS PROGRAM IS WRITTEN TO BE RUN AS A STAND-ALONE PROGRAM. HOWEVER, THE PROGRAM IS DESIGNED TO RUN UNDER AUTOMATED PRODUCT TEST SYSTEM (APT) IN ALL THREE MODES.

THE PROGRAM CAN ALSO BE RUN UNDER THE ACT 11 MONITOR.

1.3 RELATED DOCUMENTS AND STANDARDS

DIAGNOSTIC ENGINEERING STANDARDS AND CONVENTIONS PROGRAMMING PRACTICES
DOCUMENT NO. 175-003-009-00

APT INTERFACE SPECIFICATION, REV. 13

1.4 DIAGNOSTIC HIERARCHY PREREQUISITES

NONE, HOWEVER THE CPU IS ASSUMED TO BE FUNCTIONING

1.5 FAILURE ASSUMPTIONS

THE PROCESSOR IS ASSUMED TO BE FUNCTIONING PROPERLY.

148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200

2.0 OPERATING INSTRUCTIONS

2.1 LOADING AND STARTING PROCEDURES

2.1.1 LOADING

USE NORMAL PROCEDURES FOR LOADING DIAGNOSTIC PROGRAMS.

2.1.2 NORMAL START

1. LOAD SOFTWARE SWITCH REGISTER (IF USED) TO SELECT THE ROM VERSION UNDER TEST. (SEE 2.3)
2. LOAD ADDRESS 200
3. SET HARDWARE SWITCH REGISTER (IF AVAILABLE) TO SELECT THE ROM VERSION UNDER TEST (SEE 2.3).
4. START

2.1.3 OPTIONAL START

THE OPTIONAL STARTING ADDRESS IS USED TO TYPE OUT THE CONTENTS OF THE ROM FOR USE IN VISUAL VERIFICATION OR AS A DEBUGGING TOOL.

USE THE SAME PROCEDURE AS A NORMAL START EXCEPT USE ADDRESS 210 IN STEP 2.

2.2 SPECIAL ENVIRONMENTS

THIS PROGRAM IS WRITTEN TO COMPLY WITH ALL THE REQUIREMENTS OF THE APT INTERFACE SPECIFICATION. IT WILL RUN UNDER APT IN EITHER QUICK VERIFY, PROGRAM OR RUN-TIME MODES.

THIS PROGRAM IS WRITTEN TO COMPLY WITH THE ACT11/XXDP INTERFACE REQUIREMENTS.

WHEN RUNNING IN ACT11 QUICK VERIFY OR XXDP CHAIN MODE (LOC. 42 NOT 0), OR APT QUICK VERIFY AND PROGRAM MODE THE PROGRAM ATTEMPTS TO RUN WITHOUT OPERATOR INTERVENTION OR SWITCH REGISTER SELECTION. THE COMPUTED CRC IS COMPARED AGAINST THE CRC FOR ALL KNOWN VERSIONS OF THE ROM BOOTSTRAP. WHEN A MATCH IS FOUND THE VERSION OF THE MODULE IS TYPED FOR VISUAL VERIFICATION.

202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
2572.3 PROGRAM OPTIONS

THE PROGRAM AUTOMATICALLY CHECKS FOR THE PRESENCE OF A HARDWARE SWITCH REGISTER. IF NO RESPONSE IS FOUND WHEN ADDRESSING THE HARDWARE SWR (177570), THE ADDRESS OF THE SOFTWARE SWR (176) IS SUBSTITUTED.

FOR PROCESSORS WITH NO HARDWARE SWITCH REGISTER, THE OPERATOR SHOULD SET THE DESIRED SWITCH VALUE IN LOCATION 176

WARNING... IN ORDER TO ALLOW TESTING OF M7942-YB BOARDS ON THE VT71, IF LOCATION 176 IS SET TO 6, THE SOFTWARE SWITCH REGISTER WILL BE USED REGARDLESS OF HARDWARE SWITCH REGISTER AVAILABILITY.

2.3.1 SWITCH SELECTION

THE SWITCH REGISTER (HARDWARE OR SOFTWARE) IS USED TO SELECT THE VERSION OF THE ROM BOOTSTRAP TO BE TESTED ACCORDING TO THE FOLLOWING TABLE.

SWR	MODULE VERSION
---	-----
1	M9301-YA
2	M9301-YB
3	M9301-YC
4	M9400-YA (OR YC)
5	M9301-YF
6	M7942-YB
7	M9301-YD
10	M9400-YH (OR YK)
11	M9311
12	M9301-YH

IF THE CRC AND LPC FOR NEW VERSIONS ARE KNOWN BUT NOT IN THE ABOVE TABLE, SET THE SWITCH REGISTER TO ZERO AND ANSWER THE TELETYPE DIALOG.

TO DETERMINE THE CRC AND LPC FOR A NEW VERSION, START THE DIAGNOSTIC AT 200 WITH SWR=0. ANSWER 0 TO THE REQUESTS FOR THE LPC AND CRC. THE RESULTING MESSAGES WILL INDICATE THE CORRECT FUTURE RESPONSES FOR CRC AND LPC PROVIDED THE TEST IS RUN ON A KNOWN-GOOD MODULE.

2.3.2 TELETYPE DIALOG

SEVERAL QUESTIONS ARE ASKED OF THE OPERATOR IN ORDER TO OBTAIN SUFFICIENT INFORMATION FOR TESTING A ROM MODULE NOT PREVIOUSLY SUPPORTED IN THE DIAGNOSTIC. THE DIALOG IS INITIATED IF THE PROGRAM IS STARTED WITH THE SWR = 0. ALL RESPONSES ARE IN OCTAL AND TERMINATED BY A CARRIAGE RETURN.

258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307

ALL RESPONSES ARE CHECKED FOR VALID OCTAL NUMBERS.
IF AN ILLEGAL CHARACTER IS TYPED, THE PROGRAM WILL TYPE
A "?", CARRIAGE RETURN-LINE FEED AND
AWAIT THE PROPER INPUT.

IF A MISTAKE IS NOTICED BEFORE THE CARRIAGE RETURN IS USED
TO TERMINATE THE INPUT, A RUBOUT CAN BE
USED TO DELETE MISTYPED INPUT.

1. TYPE CRC VALUE:
THIS REQUESTS THE VALUE OF THE CYCLIC REDUNDANCY CHECK
PREVIOUSLY CALCULATED FOR THIS VERSION OF THE ROM MODULE.
IT IS THE VALUE AGAINST WHICH THE UNIT UNDER TEST'S CRC
WILL BE COMPARED.

2. TYPE LPC VALUE:
THIS REQUESTS THE VAULE OF THE LONGITUDINAL PARITY CHECK
PREVIOUSLY CALCULATED FOR THIS VERSION OF THE ROM MODULE.
IT IS THE VALUE AGAINST WHICH THE UNIT UNDER TEST'S LPC
WILL BE COMPARED.

3. TYPE STARTING ADDR. OF 1ST ROM ADDR. SPACE:
THIS QUESTION REFERS TO THE FACT THAT THE ROM SPACE IN
A ROM BOOTSTRAP MODULE IS DIVIDED INTO 2 DISTINCT ADDRESS SPACES. TYPE
THE STARTING ADDRESS OF THE 1ST RANGE OF ADDRESSES. THE
STANDARD M9301 & M9400 BEGIN AT 173000.

4. TYPE LENGTH (BYTES) OF 1ST ROM ADDR. SPACE:
THIS REQUESTS THE LENGTH OF THE 1ST GROUP OF ROM ADDRESSES
IN BYTES. THE STANDARD M9301 & M9400 HAVE AN INITIAL ADDRESS SPACE
OF 1000 BYTES. IF THIS SECTION OF ADDRESSES IS NOT
USED BY THIS VERSION, ANSWER 0 TO THIS QUESTION.

5. TYPE STARTING ADDR. OF 2ND ROM ADDR. SPACE:
THIS REFERS TO THE FIRST ADDRESS IN THE SECOND DISTINCT
GROUP OF ROM ADDRESSES. THE RESPONSE FOR A STANDARD
M9301 & M9400 WOULD BE 165000.

6. TYPE LENGTH (BYTES) OF 2ND ROM ADDR. SPACE:
THIS REQUESTS THE LENGTH OF THE 2ND GROUP OF ROM ADDRESSES
IN BYTES. THE STANDARD M9301 & M9400 HAVE A SECOND ADDRESS SPACE
OF 1000 BYTES. IF THIS SECTION OF ADDRESSES IS NOT USED
BY THIS VERSION, ANSWER 0 TO THIS QUESTION.

2.4 EXECUTION TIMES

THE DIAGNOSTIC COMPLETES 1 PASS IN LESS THAN 1 SEC.
ONCE THE INPUT DIALOG HAS BEEN COMPLETED.
THE PROGRAM WILL HALT UPON COMPLETION; HOWEVER, IF RUNNING
UNDER APT THE PROGRAM WILL CYCLE CONTINUOUSLY.

309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
3643.0 ERROR INFORMATION

WHEN THE DIAGNOSTIC DETECTS A DISCREPANCY BETWEEN THE EXPECTED AND COMPUTED CRC OR LPC BOTH ARE PRINTED ON THE TELETYPE.

ANY DISCREPANCY IN THE LPC CAN ASSIST IN ISOLATING THE PROBLEM TO THE ROM IC.

UNDER APT THE ERROR IS INDICATED BY DEPOSITING ERROR INFORMATION IN THE APT MAILBOX BEFORE HALTING.

4.0 PROGRESS REPORTS

AT THE END OF EACH PASS THE PROGRAM INCREMENTS TO THE LOCATION \$PASS WHICH IS IN THE APT MAILBOX. THIS LOCATION WILL ALWAYS CONTAIN THE NUMBER OF PASSES COMPLETED. \$PASS IS RESET WITH EVERY RESTART.

ADDITIONALLY, THE MESSAGE "END OF TEST" IS PRINTED ON THE CONSOLE TELETYPE AFTER EACH PASS. NORMALLY ONLY ONE PASS NEEDS TO RUN TO VERIFY THE MODULE.

5.0 TROUBLE SHOOTING

THE ALGORITHM FOR COMPUTING THE CRC IS THE SAME AS THAT USED ON 9-TRACK MAGNETIC TAPE WITH ODD PARITY. THE CRC IS CALCULATED ON A BYTE-BY-BYTE BASIS. WHILE THE ALGORITHM IS SUCCESSFUL IN DETECTING MULTIPLE ERRORS, ITS USE AS A DEBUGGING AID IS LIMITED.

THE LPC IS CALCULATED BY ASSEMBLING THE XOR OF EVERY WORD IN THE ROM. WHILE ONLY USEFUL IN CATCHING AN ODD NUMBER OF ERRORS IN EACH BIT POSITION, IT IS VERY USEFUL IN ISOLATING THE PROBLEM TO A CHIP. BY LOCATING WHICH BIT POSITIONS ARE IN DISCREPANCY, THE CORRESPONDING ROM CHIPS CAN BE ISOLATED.

IF NO OTHER CLUES CAN BE OBTAINED, START THE PROGRAM AT 210 AND COMPARE THE PRINTOUT OF THE CODE WITH THE LISTING FOR THE VERSION BEING TESTED.

* * CAUTION * *

BECAUSE THE CONTENTS READ FROM LOCATION 773024 OF THE M9301 OPTION IS CONFIGURATION DEPENDANT (SWITCH REGISTER DEPENDANT), THIS LOCATION IS NOT INCLUDED IN THE DATA CHECK. THIS LOCATION CAN BE VERIFIED BY EXAMINING IT OR BY USING THE ALTERNATE STARTING ADDRESS (SEE SECTION 2.1.3) TO PRINT OUT THIS LOCATION.

I01

.MAIN. MACY11 27(1006) 09-FEB-77 10:15 PAGE 6-1
DZM9AC.P11 09-FEB-77 10:02

SEQ 0008

365
366
367
368
369
370
371
372
373

6.0 LISTING

.ENDR

```

375
376      .ENABLE ABS
377      .LIST ME
378      .NLIST MC,MD,CND
382      000000      $$SWR=0
383      .SBTTL BASIC DEFINITIONS
(1)
(1)      .;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
(1)      001100      STACK= 1100
(1)      .EQUIV EMT,ERROR      ;;BASIC DEFINITION OF ERROR CALL
(1)      .EQUIV IOT,SCOPE      ;;BASIC DEFINITION OF SCOPE CALL
(1)
(1)      ;*MISCELLANEOUS DEFINITIONS
(1)      000011      HT= 11      ;;CODE FOR HORIZONTAL TAB
(1)      000012      LF= 12      ;;CODE FOR LINE FEED
(1)      000015      CR= 15      ;;CODE FOR CARRIAGE RETURN
(1)      000200      CRLF= 200    ;;CODE FOR CARRIAGE RETURN-LINE FEED
(1)      177776      PS= 177776   ;;PROCESSOR STATUS WORD
(1)      .EQUIV PS,PSW
(1)      177774      STKLMT= 177774 ;;STACK LIMIT REGISTER
(1)      177772      PIRQ= 177772  ;;PROGRAM INTERRUPT REQUEST REGISTER
(1)      177570      DSWR= 177570  ;;HARDWARE SWITCH REGISTER
(1)      177570      DDISP= 177570 ;;HARDWARE DISPLAY REGISTER
(1)
(1)      .;*GENERAL PURPOSE REGISTER DEFINITIONS
(1)      000000      R0= %0      ;;GENERAL REGISTER
(1)      000001      R1= %1      ;;GENERAL REGISTER
(1)      000002      R2= %2      ;;GENERAL REGISTER
(1)      000003      R3= %3      ;;GENERAL REGISTER
(1)      000004      R4= %4      ;;GENERAL REGISTER
(1)      000005      R5= %5      ;;GENERAL REGISTER
(1)      000006      R6= %6      ;;GENERAL REGISTER
(1)      000007      R7= %7      ;;GENERAL REGISTER
(1)      000006      SP= %6      ;;STACK POINTER
(1)      000007      PC= %7      ;;PROGRAM COUNTER
(1)
(1)      .;*PRIORITY LEVEL DEFINITIONS
(1)      000000      PR0= 0      ;;PRIORITY LEVEL 0
(1)      000040      PR1= 40     ;;PRIORITY LEVEL 1
(1)      000100      PR2= 100    ;;PRIORITY LEVEL 2
(1)      000140      PR3= 140    ;;PRIORITY LEVEL 3
(1)      000200      PR4= 200    ;;PRIORITY LEVEL 4
(1)      000240      PR5= 240    ;;PRIORITY LEVEL 5
(1)      000300      PR6= 300    ;;PRIORITY LEVEL 6
(1)      000340      PR7= 340    ;;PRIORITY LEVEL 7
(1)
(1)      .;*SWITCH REGISTER SWITCH DEFINITIONS
(1)      100000      SW15= 100000
(1)      040000      SW14= 40000
(1)      020000      SW13= 20000
(1)      010000      SW12= 10000
(1)      004000      SW11= 4000
(1)      002000      SW10= 2000
(1)      001000      SW09= 1000
(1)      000400      SW08= 400
(1)      000200      SW07= 200
  
```



```

(1)          000060      TKVEC= 60          ;; TTY KEYBOARD VECTOR
(1)          000064      TPVEC= 64          ;; TTY PRINTER VECTOR
(1)          000240      PIRQVEC=240       ;; PROGRAM INTERRUPT REQUEST VECTOR
384          000000      .=0
385          .SBTTL TRAP CATCHER
(1)          000000      .=0
(1)          ;; *ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
(1)          ;; *SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
(1)          ;; *LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
(1)          000174      .=174
(1) 000174 000000      DISPREG: .WORD 0          ;; SOFTWARE DISPLAY REGISTER
(1) 000176 000000      SWREG:   .WORD 0          ;; SOFTWARE SWITCH REGISTER
387          000200      .=200
388 000200 005067 000624      CLR      TYP0UT
389 000204 000167 000666      JMP      START
390 000210 012767 000001 000612      MOV     #1,TYP0UT
391 000216 000167 000654      JMP      START
392
393          177776      PS=177776
394          000034      TRAPVEC=34
395
396          001000      .=1000
397 001000 177570      SWR:      177570
398 001002 177570      DTSPY:      177570
399 001004 173000      ROMSA1: 173000
400 001006 001000      DATLN1: 512.
401 001010 165000      ROMSA2: 165000
402 001012 001000      DATLN2: 512.
403 001014 000000      XORS:   0
404 001016 000000      EXCRC:  0
405 001020 000000      EXLPC:  0
406 001022 000000      ACTCRC: 0
407 001024 000000      ACTLPC: 0
408 001026 000000      PARCNT: 0
409 001030 000000      TYP0UT: 0
410
411          .SBTTL ACT11 HOOKS
(1)
(2)          ;; *****
(1)          ;; HOOKS REQUIRED BY ACT11
(1)          001032      $SVPC=.          ;; SAVE PC
(1)          000046      .=46
(1) 000046 002060      $ENDAD          ;; 1)SET LOC.46 TO ADDRESS OF $ENDAD IN .SEOP
(1)          000052      .=52
(1) 000052 000000      .WORD 0          ;; 2)SET LOC.52 TO ZERO
(1)          001032      .= $SVPC          ;; RESTORE PC
412          .SBTTL APT PARAMETER BLOCK
(1)
(2)          ;; *****
(1)          ;; SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
(2)          ;; *****
(1)          001032      .SX=.          ;; SAVE CURRENT LOCATION
(1)          000024      .=24          ;; SET POWER FAIL TO POINT TO START OF PROGRAM
(1) 000024 000200      200          ;; FOR APT START UP
(1)          000044      .=44          ;; POINT TO APT INDIRECT ADDRESS PNTR.

```

MO1

```

(1) 000044 001032          SAPHDR ;; POINT TO APT HEADER BLOCK
(1)          001032          =.SX ;; RESET LOCATION COUNTER
(2)          ;; *****
(1)          ;; SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
(1)          ;; INTERFACE SPEC.
(1) 001032          SAPHDR:
(1) 001032 000000 $HIBTS: .WORD 0 ;; TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
(1) 001034 001046 $MADR: .WORD $MAIL ;; ADDRESS OF APT MAILBOX (BITS 0-15)
(1) 001036 000002 $TSTM: .WORD 2 ;; RUN TIME OF LONGEST TEST
(1) 001040 000002 $PASTM: .WORD 2 ;; RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
(1) 001042 000000 $UNITM: .WORD 0 ;; ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
(1) 001044 000014          .WORD $ETEND-$MAIL/2 ;; LENGTH MAILBOX-ETABLE(WORDS)
413          .SBTTL APT MAILBOX-ETABLE
(1)          ;; *****
(2)          .EVEN
(1) 001046          $MAIL:          ;; APT MAILBOX
(1) 001046 000000 $MSGTY: .WORD AMSGTY ;; MESSAGE TYPE CODE
(1) 001050 000000 $FATAL: .WORD AFATAL ;; FATAL ERROR NUMBER
(1) 001052 000000 $TESTN: .WORD ATESTN ;; TEST NUMBER
(1) 001054 000000 $PASS: .WORD APASS ;; PASS COUNT
(1) 001056 000000 $DEVCT: .WORD ADEVCT ;; DEVICE COUNT
(1) 001060 000000 $UNIT: .WORD AUNIT ;; I/O UNIT NUMBER
(1) 001062 000000 $MSGAD: .WORD AMSGAD ;; MESSAGE ADDRESS
(1) 001064 000000 $MSGLG: .WORD AMSLG ;; MESSAGE LENGTH
(1) 001066          $ETABLE:          ;; APT ENVIRONMENT TABLE
(1) 001066          000          $ENV: .BYTE AENV ;; ENVIRONMENT BYTE
(1) 001067          000          $ENVM: .BYTE AENVM ;; ENVIRONMENT MODE BITS
(1) 001070 000000 $SWREG: .WORD ASWREG ;; APT SWITCH REGISTER
(1) 001072 000000 $USWR: .WORD AUSWR ;; USER SWITCHES
(1) 001074 000000 $CPUOP: .WORD ACPUOP ;; CPU TYPE, OPTIONS
(1)          *          BITS 15-11=CPU TYPE
(1)          *          11/04=01, 11/05=02, 11/20=03, 11/40=04, 11/45=05
(1)          *          11/70=06, PDQ=07, Q=10
(1)          *          BIT 10=REAL TIME CLOCK
(1)          *          BIT 9=FLOATING POINT PROCESSOR
(1)          *          BIT 8=MEMORY MANAGEMENT
(1) 001076          $ETEND:
(1)          .MEXIT
414
415 001076 005067 177746 START: CLR $FATAL ;; CLEAR ERROR NO.
416 001102 005067 177740 CLR $MSGTYP ;; CLEAR MESSAGE TYPE (APT)
417 001106 012767 000001 177736 MOV #1, $TESTN ;; SET TEST NO.
418          .SBTTL INITIALIZE THE COMMON TAGS
(1) 001114 012706 000500 MOV #500, SP ;; SETUP THE STACK POINTER
(1)          ;; INITIALIZE A FEW VECTORS
(1) 001120 012737 004570 000034 MOV #STRAP, @TRAPVEC ;; TRAP VECTOR FOR TRAP CALLS
(1) 001126 012737 000340 000036 MOV #340, @TRAPVEC+2; LEVEL 7
(1) 001134 012737 004412 000024 MOV #SPWRN, @PWAVEC ;; POWER FAILURE VECTOR
(1) 001142 012737 000340 000026 MOV #340, @PWAVEC+2 ;; LEVEL 7
(2)          ;; SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
(2)          ;; EQUAL TO A "-1" SETUP FOR A SOFTWARE SWITCH REGISTER.
(2) 001150 013746 000004 MOV @ERRVEC, -(SP) ;; SAVE ERROR VECTOR
(2) 001154 012737 001210 000004 MOV #64, @ERRVEC ;; SET UP ERROR VECTOR
(2) 001162 012767 177570 177610 MOV #DSWR, SWR ;; SETUP FOR A HARDWARE SWICH REGISTER

```

NO1

.MAIN. MRCY11 27(1006) 09-FEB-77 10:15 PAGE 7-4
 DZM9AC.P11 09-FEB-77 10:02

INITIALIZE THE COMMON TAGS

SEG 0013

```

(2) 001170 012767 177570 177604      MOV    #DDISP,DISPLAY ;: AND A HARDWARE DISPLAY REGISTER
(2) 001176 022777 177777 177574      CMP    #-1,#SWR      ;: TRY TO REFERENCE HARDWARE SWR
(2) 001204 001012                    BNE    66$           ;: BRANCH IF NO TIMEOUT TRAP OCCURRED
(2) 001206 000403                    BR     65$           ;: AND THE HARDWARE SWR IS NOT = -1
(2) 001210 012716 001216 64$:        MOV    #65$, (SP)    ;: BRANCH IF NO TIMEOUT
(2) 001214 000002                    RTI                               ;: SET UP FOR TRAP RETURN
(2) 001216 012767 000176 177554 65$:  MOV    #SWREG,SWR    ;: POINT TO SOFTWARE SWR
(2) 001224 012767 000174 177550      MOV    #DISPREG,DISPLAY
(2) 001232 012637 000004 66$:        MOV    (SP)+,#ERRVEC ;: RESTORE ERROR VECTOR
(1)
(2) 001236 005067 177612                    CLR    $PASS        ;: CLEAR PASS COUNT
(2) 001242 132767 000200 177617      BITB  #APTSIZE,$ENVM ;: TEST USER SIZE UNDER APT
(2) 001250 001403                    BEQ    67$           ;: YES, USE NON-APT SWITCH
(2) 001252 012767 001070 177520      MOV    #SSWREG,SWR  ;: NO, USE APT SWITCH REGISTER
(2) 001260
419 001260 026727 176556 002060      CMP    42,#SENDAD   ;: ACT AUTO MODE?
420 001266 001402                    BEQ    RESTRT       ;: YIF SO: BR
421 001270 104401                    TYPE
422 001272 005224                    TITL
423
424 001274 123727 000176 000006 RESTRT: CMPB  #176,#6        ;: IS SOFTWARE SWITCH REGISTER =6?
425 001302 001003                    BNE    1$           ;: IF NOT, TEST NORMALLY
426 001304 012767 000176 177466      MOV    #SWREG,SWR   ;: IF SO USE THE SOFTWARE SWITCH REG
427 001312 017700 177462 1$:        MOV    #SWR,R0      ;: GET SWR
428 001316 001424                    BEQ    GETIN        ;: IF ZERO: GET INPUT
429 001320 006300                    ST2:  ASL    R0
430 001322 016067 004706 177470      MOV    TXLPC(R0),EXLPC ;: FETCH EXPECT. LPC
431 001330 016067 004644 177460      MOV    TXCRC(R0),EXCRC ;: FETCH EXPECTED CRC
432 001336 016067 004752 177442      MOV    TDLN1(R0),DATLN1 ;: FETCH 1ST LENGTH
433 001344 016067 005014 177432      MOV    TRMSA1(R0),ROMSA1 ;: FETCH 1ST STARTING ADDR.
434 001352 016067 005056 177432      MOV    TDLN2(R0),DATLN2 ;: FETCH 2ND LENGTH
435 001360 016067 005120 177422      MOV    TRMSA2(R0),ROMSA2 ;: FETCH 2ND STARTING ADDR
436 001366 000450                    BR     CHECK        ;: GO COMPUTE
437 001370 005767 176446  GETIN:  TST    42           ;: UNDER ACT AUT ACCEPT?
438 001374 001045                    BNE    CHECK        ;: IF SO; BR USE DEFAULT PARAMETERS
439 001376 122767 000001 177462      CMPB  #1,$ENV       ;: UNDER APT?
440 001404 001441                    BEQ    CHECK        ;: IF SO: BR
441 001406 005767 177416      TST    TYP0UT       ;: ROM TYPE OPTION
442 001412 001012                    BNE    GET2         ;: IF SO: BR
443 001414 104401                    TYPE
444 001416 005237                    GETCRC
445 001420 104407                    RDOCT
446 001422 012667 177370      MOV    (SP)+,EXCRC  ;: STORE EXPECT. CRC
447 001426 104401                    TYPE                ;: TYPE LPC INPUT REQUEST
448 001430 005263                    GETLPC
449 001432 104407                    RDOCT
450 001434 012667 177360      MOV    (SP)+,EXLPC  ;: STORE EXPECTED LPC
451 001440 104401  GET2:  TYPE
452 001442 005443                    SA1                  ;: REQUEST 1ST ADDRESS SPACE
453 001444 104407                    RDOCT                ;: INPUT SA
454 001446 012667 177332      MOV    (SP)+,ROMSA1 ;: REQUEST LENGTH OF 1ST ADDR. SPACE
455 001452 104401                    TYPE
456 001454 005603                    SIZE1
457 001456 104407                    RDOCT                ;: INPUT LENGTH
458 001460 012667 177322      MOV    (SP)+,DATLN1

```


515	001750	005332				EXLPMG				
516	001752	016746	177042			MOV	EXLPC,-(SP)			;PUT EXPECTED LPC ON STACK
517	001756	104402				TYPOC				;TYPE EXPECTED LPC
518	001760	104401				TYPE				;TYPE ACTUAL LPC MESSG.
519	001762	005401				ACLPMG				
520	001764	016746	177034			MOV	ACTLPC,-(SP)			;PUT ACTUAL LPC ON STACK
521	001770	104402				TYPOC				;TYPE ACTUAL LPC
522	001772	026727	176044	002060		CMP	42,#SENDAD			;UNDER ACT AUTO MODE?
523	002000	001404				BEQ	15			;IF SO: BR
524	002002	122767	000001	177056		CMPB	#1,SENV			;UNDER APT?
525	002010	001007				BNE	CK2			;IF NOT: BR
526	002012	012767	000003	177030	1S:	MOV	#3,\$FATAL			;MOVE TO MAILBOX ERROR NO. **** 3 ****
527	002020	012767	000001	177020		MOV	#1,\$MSGTYP			;SET MAILBOX FOR FATAL ERROR
528	002026	000000				HALT				;LPC ERROR
529										
530	002030	026727	176006	002060	CK2:	CMP	42,#SENDAD			;ACT AUTO ACCEPT?
531	002036	001402				BEQ	15			;IF SO: BR
532	002040	104401				TYPE				;TYPE END OF TEST
533	002042	005424				EOTST				
534	002044	005267	177004		1S:	INC	\$PASS			;BUMP PASS COUNT
535	002050	013700	000042			MOV	#42,R0			;CHECK APT
536	002054	001405				BEQ	GOAGIN			;KEEP GOING
537	002056	000005				RESET				
538	002060	004710			SENDAD:	JSR	PC,(R0)			;ACT HOOKS
539	002062	000240				NOP				
540	002064	000240				NOP				
541	002066	000240				NOP				
542	002070	000167	177200		GOAGIN:	JMP	RESTRT			;DO AGAIN
543										
544	002074	016767	176722	176712	CRC:	MOV	ACTCRC,XORS			
545	002102	111104			CLO:	MOVB	(R1),R4			;GET CHAR.
546	002104	022701	173024			CMP	#173024,R1			;LOCATION EFFECTED BY SWITCHES
547	002110	001004				BNE	CL3			;IF NOT: BR
548	002112	005300				DEC	R0			;FIX COUNTERS
549	002114	005300				DEC	R0			
550	002116	005721				TST	(R1)+			;FIX POINTER
551	002120	000770				BR	CLO			;CONTINUE
552	002122	004767	000114		CL3:	JSR	PC,PARITY			;GO GET PARITY
553	002126	004767	000166			JSR	PC,XOR			;XOR CHAR
554	002132	000241				CLC				
555	002134	006004				ROR	R4			;ROTATE 1 POS. RIGHT
556	002136	103014				BCC	CL2			;IF NO CARRY: BR
557	002140	052704	000400			BIS	#400,R4			;SET BIT NINE
558	002144	000241				CLC				
559	002146	010405			CL1:	MOV	R4,R5			;SAVE CHAR
560	002150	042705	177703			BIC	#177703,R5			
561	002154	005105				COM	R5			
562	002156	042705	177703			BIC	#177703,R5			
563	002162	042704	000074			BIC	#74,R4			
564	002166	050504				BIS	R5,R4			
565	002170	010467	176620		CL2:	MOV	R4,XORS			
566	002174	005300				DEC	R0			
567	002176	001402				BEQ	CLLAST			;IF LAST CAR.: BR
568	002200	000167	177676			JMP	CLO			;GET NEXT CHAR.
569	002204	016704	176604		CLLAST:	MOV	XORS,R4			
570	002210	005167	176600			COM	XORS			

.MAIN. MACY11 27(1006) 09-FEB-77 10:15 PAGE 7-7
 DZM9AC.P11 09-FEB-77 10:02 INITIALIZE THE COMMON TAGS

SEG 0016

571	002214	042767	177050	176572		BIC	#177050,XORS	
572	002222	042704	177727			BIC	#177727,R4	;COMPLEMENT ALL BUT BITS 3 & 5
573	002226	050467	176562			BIS	R4,XORS	
574	002232	016767	176556	176562		MOV	XORS,ACTCRC	
575	002240	000207				RTS	PC	
576	002242	005067	176560		PARITY:	CLR	PARCNT	;CLEAR BIT COUNTER
577	002246	012703	000010			MOV	#10,R3	;SET NO. OF BITS
578	002252	032704	000001		CLPO:	BIT	#1,R4	;SEE IF ONE BIT
579	002256	001402				BEQ	CLP1	;IF NOT: BR
580	002260	005267	176542			INC	PARCNT	;BUMP COUNTER
581	002264	000241			CLP1:	CLC		
582	002266	006004				ROR	R4	;ROTATE TO NEXT BIT
583	002270	005303				DEC	R3	
584	002272	001367				BNE	CLPO	;CONTINUE FOR ALL BITS
585	002274	112104				MOVB	(R1)+,R4	
586	002276	042704	177400			BIC	#177400,R4	
587	002302	032767	000001	176516		BIT	#1,PARCNT	;SEE IF ODD # OF ONE BITS
588	002310	001002				BNE	CLP2	;IF SO: BR
589	002312	052704	000400			BIS	#400,R4	;SET PARITY BIT
590	002316	000207			CLP2:	RTS	PC	;EXIT
591								
592	002320	010446			XOR:	MOV	R4 -(SP)	;XOR SUBROUTINE: R4 WITH XORS
593	002322	046716	176466			BIC	XORS,(SP)	
594	002326	040467	176462			BIC	R4,XORS	
595	002332	052667	176456			BIS	(SP)+,XORS	
596	002336	016704	176452			MOV	XORS,R4	
597	002342	000207				RTS	PC	
598								
599	002344	016767	176454	176442	LPC:	MOV	ACTLPC,XORS	
600	002352	012104			LPC1:	MOV	(R1)+,R4	
601	002354	022701	173026			CMP	#173026,R1	;LOCATION EFFECTED BY SWITCHES
602	002350	001402				BEQ	LPC2	;IF SO: SKIP LOC. BY BRANCHING
603	002362	004767	177732			JSR	PC,XOR	
604	002366	005300			LPC2:	DEC	R0	
605	002370	001370				BNE	LPC1	
606	002372	016767	176416	176424		MOV	XORS,ACTLPC	
607	002400	000207				RTS	PC	
608								
609	002402	104401			TYPROM:	TYPE		;TYPE HEADER
610	002404	005751				TYPHOR		
611	002406	016700	176372			MOV	ROMSA1,R0	;POINT OT 1ST ROM SPACE
612	002412	016701	176370			MOV	DATLN1,R1	;PUT LENGTH IN R1
613	002416	006201				ASR	R1	;CONVERT TO WORDS
614	002420	001402				BEQ	TYPRI	;BRANCH IF 1ST ROM SPACE NOT USED
615	002422	004767	000026			JSR	PC,TYP	;GO TYPE 1ST ADDR. SPACE
616	002428	016700	176356		TYPRI:	MOV	ROMSA2,R0	;POINT TO 2ND ADDR. SPACE
617	002432	016701	176354			MOV	DATLN2,R1	;PUT LENGTH IN R1
618	002436	006201				ASR	R1	;CONVERT TO WORDS
619	002440	001402				BEQ	ENDOT	;BR IF 2ND ADDR. SPACE NOT USED
620	002442	004767	000006			JSR	PC,TYP	;GO TYPE 2ND ADDR. SPACE
621	002446	104401			ENDOT:	TYPE		
622	002450	005424				EOTST		
623	002452	000000				HALT		
624								
625	002454	104401			TYP:	TYPE		
626	002456	005743				CARLF		

```

627 002460 000403          BR      TYP3
628 002462 032700 000003  TYP0:  BIT      #3,RO      ;ADDRESS MULTIPLE OF 4?
629 002466 001006          BNE     TYP2      ;IF NOT: BR
630 002470 104401          TYP3:  TYPE
631 002472 005743          CARLF
632 002474 010046          MOV     RO,-(SP)      ;PUT ADDRESS ON STACK
633 002476 104402          TYPOC  ;TYPE ADDR.
634 002500 104401          TYPE
635 002502 006006          COLON
636 002504 012046          TYP2:  MOV     (RO)+,-(SP) ;PUT DATA ON STACK
637 002506 104402          TYPOC  ;TYP DATA
638 002510 104401          TYPE  ;TYPE 2 SPACES
639 002512 005746          SP2
640 002514 005301          DEC     R1          ;FINISHED?
641 002516 001361          BNE     TYP0      ;IF NOT: BR
642 002520 000207          RTS    PC          ;RETURN
643 002522 005000          AUTACT: CLR    RO
644 002524 062700 000002  AUT1:  ADD     #2,RO   ;BUMP TALBE INDEX
645 002530 026027 005162 177777  CMP     TMSG(RO),#-1 ;CHECKED ALL KNOWN VERSIONS?
646 002536 001425          BEQ    AUTERR     ;IF SO: BR
647 002540 026067 004644 176254  CMP     TXCRC(RO),ACTCRC ;DOES THIS CRC AGREE?
648 002546 001366          BNE     AUT1     ;IF NOT: KEEP LOOKING
649 002550 023727 000042 002060  CMP     #42,#SENDAD ;UNDER ACT AUTO ACCEPT?
650 002556 001405          BEQ    AUT3     ;IF SO: BR
651 002560 016067 005162 000002  MOV     TMSG(RO),AUT2 ;SET UP VERSION MESSAGE
652 002566 104401          TYPE
653 002570 000000          AUT2:  0
654 002572 016067 004644 176216  AUT3:  MOV     TXCRC(RO),EXCRC ;SET EXPECTED CRC
655 002600 016067 004706 176212  MOV     TXLPC(RO),EXLPC ;SET EXPECTED LPC
656 002606 000167 177124          JMP    CK1        ;CHECK LPC
657
658 002612 104401          AUTERR: TYPE
659 002614 006012          AUTERM
660 002616 012767 000001 176224  MOV     #1,$FATAL   ;MOVE TO MAILBOX ERROR NO. **** 1 ****
661 002624 012767 000001 176214  MOV     #1,$MSGTYP  ;SET MAILBOX FOR FATAL ERROR
662 002632 000000          HALT ;AUTO ACCEPT FAILED
663
664 .SBTTL TTY INPUT ROUTINE
(1)
(2)
(1) 002634 177560 $TKS: .WORD 177560 ;;TTY KBD STATUS
(1) 002636 177562 $TKB: .WORD 177562 ;;TTY KBD BUFFER
(1) .ENABL LSB
(1)
(1) .DSABL LSB
(1)
(2)
(1) ;*****
(1) ;THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
(1) ;CALL:
(1) * RDCHR ;;INPUT A SINGLE CHARACTER FROM THE TTY
(1) * RETURN HERE ;;CHARACTER IS ON THE STACK
(1) * ;;WITH PARITY BIT STRIPPED OFF
(1)
(1)
(1) $RDCHR: MOV (SP),-(SP) ;;PUSH DOWN THE PC
    
```



```

(1) 003464 010067 000004      MOV    RO,61$      ;; SETUP MESSAGE ADDRESS FOR APT
(1) 003470 004767 000230      JSR    PC,$ATY3   ;; SPOOL MESSAGE TO APT
(1) 003474 000000              .WORD 0           ;; MESSAGE ADDRESS
(1) 003476 132767 000040 175363 61$: BITB    #APTC SUP,$ENVM  ;; APT CONSOLE SUPPRESSED
(1) 003504 001003              BNE    60$        ;; YES, SKIP TYPE OUT
(1) 003506 112046              2$:  MOVB   (RO)+,-(SP)  ;; PUSH CHARACTER TO BE TYPED ONTO STACK
(1) 003510 001005              BNE    4$        ;; BR IF IT ISN'T THE TERMINATOR
(1) 003512 005726              TST   (SP)+      ;; IF TERMINATOR POP IT OFF THE STACK
(1) 003514 012600              60$: MOV   (SP)+,RO  ;; RESTORE RO
(1) 003516 062716 000002      3$:  ADD   #2,(SP)  ;; ADJUST RETURN PC
(1) 003522 000002              RTI                   ;; RETURN
(1) 003524 122716 000011      4$:  CMPB   #HT,(SP)  ;; BRANCH IF <HT>
(1) 003530 001430              BEQ   8$        ;;
(1) 003532 122716 000200      CMPB   #CRLF,(SP)  ;; BRANCH IF NOT <CRLF>
(1) 003536 001006              BNE   5$        ;;
(1) 003540 005726              TST   (SP)+      ;; POP <CR><LF> EQUIV
(1) 003542 104401              TYPE                   ;; TYPE A CR AND LF
(1) 003544 003225              $CRLF
(1) 003546 105067 000130      CLRB   $CHARCNT   ;; CLEAR CHARACTER COUNT
(1) 003552 000755              BR    2$        ;; GET NEXT CHARACTER
(1) 003554 004767 00005$      5$:  JSR    PC,$TYPEC  ;; GO TYPE THIS CHARACTER
(1) 003560 126726 00013'      6$:  CMPB   $FILLC,(SP)+  ;; IS IT TIME FOR FILLER CHARS.?
(1) 003564 001350              BNE   2$        ;; IF NO GO GET NEXT CHAR.
(1) 003566 016746 0001$      MOV   $NULL,-(SP)  ;; GET # OF FILLER CHARS. NEEDED
(1)                                AND THE NULL CHAR.
(1) 003572 105366 0000 .      7$:  DECB   1(SP)     ;; DOES A NULL NEED TO BE TYPED?
(1) 003576 002770              BLT   6$        ;; BR IF NO--GO POP THE NULL OFF OF STACK
(1) 003600 004767 000' 2      JSR    PC,$TYPEC  ;; GO TYPE A NULL
(1) 003604 105367 000 '2      DECB   $CHARCNT   ;; DO NOT COUNT AS A COUNT
(1) 003610 000770              BR    7$        ;; LOOP
(1)
(1)                                ;HORIZONTAL TAB PROCESSOR
(1)
(1) 003612 112716 000040      8$:  MOVB   #' (SP)   ;; REPLACE TAB WITH SPACE
(1) 003616 004767 000014      9$:  JSR    PC,$TYPEC  ;; TYPE A SPACE
(1) 003622 132767 000007 000052 BITB   #7,$CHARCNT  ;; BRANCH IF NOT AT
(1) 003630 001372              BNE   9$        ;; TAB STOP
(1) 003632 005726              TST   (SP)+      ;; POP SPACE OFF STACK
(1) 003634 000724              BR    2$        ;; GET NEXT CHARACTER
(1) 003636 105777 000044      $TYPEC: TSTB   2$STPS  ;; WAIT UNTIL PRINTER IS READY
(1) 003642 100375              BPL   $TYPEC
(1) 003644 116677 000002 000036 MOVB   2(SP),2$STPB  ;; LOAD CHAR TO BE TYPED INTO DATA REG.
(1) 003652 122766 000015 000002 CMPB   #CR,2(SP)   ;; IS CHARACTER A CARRIAGE RETURN?
(1) 003660 001003              BNE   1$        ;; BRANCH IF NO
(1) 003662 105067 000014      CLRB   $CHARCNT   ;; YES--CLEAR CHARACTER COUNT
(1) 003666 000406              BR    $TYPEX
(1) 003670 122766 000012 000002 1$:  CMPB   #LF,2(SP)  ;; IS CHARACTER A LINE FEED?
(1) 003676 001402              BEQ   $TYPEX     ;; BRANCH IF YES
(1) 003700 105227              INCB   (PC)+     ;; COUNT THE CHARACTER
(1) 003702 000000      $CHARCNT: .WORD 0  ;; CHARACTER COUNT STORAGE
(1) 003704 000207      $TYPEX: RTS    PC
(1)
(1) 003706 177564      $STPS: .WORD 177564  ;; TTY PRINTER STATUS REG. ADDRESS
(1) 003710 177566      $STPB: .WORD 177566  ;; TTY PRINTER BUFFER REG. ADDRESS
(1) 003712 000          $NULL: .BYTE 0      ;; CONTAINS NULL CHARACTER FOR FILLS
(1) 003713 002          $FILLS: .BYTE 2    ;; CONTAINS # OF FILLER CHARACTERS REQUIRED
    
```

```

(1) 003714 012 $FILLC: .BYTE 12 ;; INSERT FILL CHARS. AFTER A "LINE FEED"
(1) 003715 000 $STPFLG: .BYTE 0 ;; "TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
667 .SBTTL APT COMMUNICATIONS ROUTINE
(1)
(2)
(1) 003716 112767 000001 000236 $SATY1: MOVB #1,$FFLG ;; TO REPORT FATAL ERROR
(1) 003724 112767 000001 000226 $SATY3: MOVB #1,$I'FLG ;; TO TYPE A MESSAGE
(1) 003732 000403 BR $ATYC
(1) 003734 112767 000001 000220 $SATY4: MOVB #1,$FFLG ;; TO ONLY REPORT FATAL ERROR
(1) 003742 $SATYC:
(3) 003742 010046 MOV R0,-(SP) ;; PUSH R0 ON STACK
(3) 003744 010146 MOV R1,-(SP) ;; PUSH R1 ON STACK
(1) 003746 105767 000206 TSTB $MFLG ;; SHOULD TYPE A MESSAGE?
(1) 003752 001450 BEQ 55 ;; IF NOT: BR
(1) 003754 122767 000001 175104 CMPB #APTENV,$ENV ;; OPERATING UNDER APT?
(1) 003762 001031 BNE 35 ;; IF NOT: BR
(1) 003764 132767 000100 175075 BITB #APTPOOL,$ENVM ;; SHOULD SPOOL MESSAGES?
(1) 003772 001425 BEQ 35 ;; IF NOT: BR
(1) 003774 017600 000004 MOV #24(SP),R0 ;; GET MESSAGE ADDR.
(1) 004000 062766 000002 000004 ADD #2,4(SP) ;; BUMP RETURN ADDR.
(1) 004006 005767 175034 15: TST $MSGTYPE ;; SEE IF DONE W/ LAST XMISSION?
(1) 004012 001375 BNE 15 ;; IF NOT: WAIT
(1) 004014 010067 175042 MOV R0,$MSGAD ;; PUT ADDR IN MAILBOX
(1) 004020 105720 25: TSTB (R0)+ ;; FIND END OF MESSAGE
(1) 004022 001376 BNE 25
(1) 004024 166700 175032 SUB $MSGAD,R0 ;; SUB START OF MESSAGE
(1) 004030 006200 ASR R0 ;; GET MESSAGE LNTH IN WORDS
(1) 004032 010067 175026 MOV R0,$MSGLGT ;; PUT LENGTH IN MAILBOX
(1) 004036 012767 000004 175002 MOV #4,$MSGTYPE ;; TELL APT TO TAKE MSG.
(1) 004044 000413 BR 55
(1) 004046 017667 000004 000016 35: MOV #24(SP),R4 ;; PUT MSG ADDR IN JSR LINKAGE
(1) 004054 062766 000002 000004 ADD #2,4(SP) ;; BUMP RETURN ADDRESS
(3) 004062 016746 173710 MOV 177776,-(SP) ;; PUSH 177776 ON STACK
(1) 004066 004767 177332 JSR PC,$TYPE ;; CALL TYPE MACRO
(1) 004072 000000 45: .WORD 0
(1) 004074 55:
(1) 004074 105767 000062 105: TSTB $FFLG ;; SHOULD REPORT FATAL ERROR?
(1) 004100 001416 BEQ 125 ;; IF NOT: BR
(1) 004102 005767 174760 TST $ENV ;; RUNNING UNDER APT?
(1) 004106 001413 BEQ 125 ;; IF NOT: BR
(1) 004110 005767 174732 115: TST $MSGTYPE ;; FINISHED LAST MESSAGE?
(1) 004114 001375 BNE 115 ;; IF NOT: WAIT
(1) 004116 017667 000004 174724 MOV #24(SP),$FATAL ;; GET ERROR #
(1) 004124 062766 000002 000004 ADD #2,4(SP) ;; BUMP RETURN ADDR.
(1) 004132 005267 174710 INC $MSGTYPE ;; TELL APT TO TAKE ERROR
(1) 004136 105067 000020 125: CLRB $FFLG ;; CLEAR FATAL FLAG
(1) 004142 105067 000013 CLRB $LFLG ;; CLEAR LOG FLAG
(1) 004146 105067 000006 CLRB $MFLG ;; CLEAR MESSAGE FLAG
(3) 004152 012601 MOV (SP)+,R1 ;; POP STACK INTO R1
(3) 004154 012600 MOV (SP)+,R0 ;; POP STACK INTO R0
(1) 004156 000207 RTS PC ;; RETURN
(1) 004160 000 $MFLG: .BYTE 0 ;; MESSG. FLAG
(1) 004161 000 $LFLG: .BYTE 0 ;; LOG FLAG
(1) 004162 000 $FFLG: .BYTE 0 ;; FATAL FLAG
(1) 004164 .EVEN
(1) 000200 APTSIZE=200

```

```

(1)          000001      APTENV=001
(1)          000100      APTSPool=100
(1)          000040      APTCSUP=040
668          .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
(1)
(2)          ;*****
(1)          ;THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
(1)          ;OCTAL (ASCII) NUMBER AND TYPE IT.
(1)          ;$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
(1)          ;CALL:
(1)          ;      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
(1)          ;      TYPOS      ;;CALL FOR TYPEOUT
(1)          ;      .BYTE  N      ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
(1)          ;      .BYTE  M      ;;M=1 OR 0
(1)          ;      ;;1=TYPE LEADING ZEROS
(1)          ;      ;;0=SUPPRESS LEADING ZEROS
(1)          ;
(1)          ;$STYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
(1)          ;$TYPOS OR $TYPOC
(1)          ;CALL:
(1)          ;      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
(1)          ;      TYPON      ;;CALL FOR TYPEOUT
(1)          ;
(1)          ;$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
(1)          ;CALL:
(1)          ;      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
(1)          ;      TYPOC      ;;CALL FOR TYPEOUT
(1)          ;
(1) 004164 017646 000000      $TYPOS: MOV      2(SP),-(SP)      ;;PICKUP THE MODE
(1) 004170 116667 000001 000211  MOVB     1(SP),%ZFILL      ;;LOAD ZERO FILL SWITCH
(1) 004176 112667 000207      MOVB     (SP)+,%MODE+1      ;;NUMBER OF DIGITS TO TYPE
(1) 004202 062716 000C02      ADD      #2,(SP)          ;;ADJUST RETURN ADDRESS
(1) 004206 000406      BR      $TYPON
(1) 004210 112767 000001 000171  $TYPOC: MOVB     #1,%ZFILL      ;;SET THE ZERO FILL SWITCH
(1) 004216 112767 000006 000165      MOVB     #6,%MODE+1      ;;SET FOR SIX(6) DIGITS
(1) 004224 112767 000005 000154  $TYPON: MOVB     #5,%CNT      ;;SET THE ITERATION COUNT
(1) 004232 010346      MOV      R3,-(SP)        ;;SAVE R3
(1) 004234 010446      MOV      R4,-(SP)        ;;SAVE R4
(1) 004236 010546      MOV      R5,-(SP)        ;;SAVE R5
(1) 004240 116704 000145      MOVB     %MODE+1,R4      ;;GET THE NUMBER OF DIGITS TO TYPE
(1) 004244 005404      NEG      R4
(1) 004246 062704 000006      ADD      #6,R4          ;;SUBTRACT IT FOR MAX. ALLOWED
(1) 004252 110467 000132      MOVB     R4,%MODE      ;;SAVE IT FOR USE
(1) 004256 116704 000125      MOVB     %ZFILL,R4      ;;GET THE ZERO FILL SWITCH
(1) 004262 016605 000012      MOV      12(SP),R5      ;;PICKUP THE INPUT NUMBER
(1) 004266 005003      CLR      R3            ;;CLEAR THE OUTPUT WORD
(1) 004270 006105      1$: ROL      R5          ;;ROTATE MSB INTO "C"
(1) 004272 000404      BR      3$            ;;GO DO MSB
(1) 004274 006105      2$: ROL      R5          ;;FORM THIS DIGIT
(1) 004276 006105      ROL      R5
(1) 004300 006105      ROL      R5
(1) 004302 010503      MOV      R5,R3
(1) 004304 006103      3$: ROL      R3          ;;GET LSB OF THIS DIGIT
(1) 004306 105367 000076      DECB     %MODE          ;;TYPE THIS DIGIT?
(1) 004312 100016      BPL      7$            ;;BR IF NO
(1) 004314 042703 177770      BIC      #177770,R3      ;;GET RID OF JUNK

```

```

(1) 004320 001002      BNE      4$      ;: TEST FOR 0
(1) 004322 005704      TST      R4      ;: SUPPRESS THIS 0?
(1) 004324 001403      BEQ      5$      ;: BR IF YES
(1) 004326 005204      4$: INC      R4      ;: DON'T SUPPRESS ANYMORE 0'S
(1) 004330 052703 000060  BIS      #'0,R3 ;: MAKE THIS DIGIT ASCII
(1) 004334 052703 000040  5$: BIS      #' ,R3 ;: MAKE ASCII IF NOT ALREADY
(1) 004340 110367 000040  MOVB     R3,8$   ;: SAVE FOR TYPING
(1) 004344 104401 004404  TYPE     8$     ;: GO TYPE THIS DIGIT
(1) 004350 105367 000032  7$: DECB   $OCNT ;: COUNT BY 1
(1) 004354 003347      BGT      2$     ;: BR IF MORE TO DO
(1) 004356 002402      BLT      6$     ;: BR IF DONE
(1) 004360 005204      INC      R4     ;: INSURE LAST DIGIT ISN'T A BLANK
(1) 004362 000744      BR       2$     ;: GO DO THE LAST DIGIT
(1) 004364 012605  6$: MOV     (SP)+,R5 ;: RESTORE R5
(1) 004366 012604      MOV     (SP)+,R4 ;: RESTORE R4
(1) 004370 012603      MOV     (SP)+,R3 ;: RESTORE R3
(1) 004372 016666 000002 000004  MOV     2(SP),4(SP) ;: SET THE STACK FOR RETURNING
(1) 004400 012616      MOV     (SP)+,(SP)
(1) 004402 000002      RTI                     ;: RETURN
(1) 004404      000      8$: .BYTE   0      ;: STORAGE FOR ASCII DIGIT
(1) 004405      000      .BYTE   0      ;: TERMINATOR FOR TYPE ROUTINE
(1) 004406      000      $OCNT: .BYTE  0      ;: OCTAL DIGIT COUNTER
(1) 004407      000      $DFILL: .BYTE  0      ;: ZERO FILL SWITCH
(1) 004410 000000      $OMODE: .WORD  0      ;: NUMBER OF DIGITS TO TYPE
669

```

.SBTTL POWER DOWN AND UP ROUTINES

```

(1)
(2)
(1)
:*****
: POWER DOWN ROUTINE
(1) 004412 012737 004552 000024 $PWRDN: MOV     # $ILLUP,@#PWRVEC ;: SET FOR FAST UP
(1) 004420 012737 000340 000026      MOV     #340,@#PWRVEC+2 ;: PRIO:7
(3) 004426 010046      MOV     R0,-(SP) ;: PUSH R0 ON STACK
(3) 004430 010146      MOV     R1,-(SP) ;: PUSH R1 ON STACK
(3) 004432 010246      MOV     R2,-(SP) ;: PUSH R2 ON STACK
(3) 004434 010346      MOV     R3,-(SP) ;: PUSH R3 ON STACK
(3) 004436 010446      MOV     R4,-(SP) ;: PUSH R4 ON STACK
(3) 004440 010546      MOV     R5,-(SP) ;: PUSH R5 ON STACK
(3) 004442 017746 174332  MOV     @SWR,-(SP) ;: PUSH @SWR ON STACK
(1) 004446 010667 000104  MOV     SP,$SAVR6 ;: SAVE SP
(1) 004452 012737 004464 000024  MOV     # $PWRUP,@#PWRVEC ;: SET UP VECTOR
(1) 004460 000000      HALT
(1) 004462 000776      BR      -2      ;: HANG UP

```

```

(2)
(1)
:*****
: POWER UP ROUTINE
(1) 004464 012737 004552 000024 $PWRUP: MOV     # $ILLUP,@#PWRVEC ;: SET FOR FAST DOWN
(1) 004472 016706 000060      MOV     $SAVR6,SP ;: GET SP
(1) 004476 005067 000054      CLR     $SAVR6 ;: WAIT LOOP FOR THE TTY
(1) 004502 005267 000050  1$: INC     $SAVR6 ;: WAIT FOR THE INC
(1) 004506 001375      BNE     1$     ;: OF WORD
(3) 004510 012677 174264      MOV     (SP)+,@SWR ;: POP STACK INTO @SWR
(3) 004514 012605      MOV     (SP)+,R5 ;: POP STACK INTO R5
(3) 004516 012604      MOV     (SP)+,R4 ;: POP STACK INTO R4
(3) 004520 012603      MOV     (SP)+,R3 ;: POP STACK INTO R3
(3) 004522 012602      MOV     (SP)+,R2 ;: POP STACK INTO R2
(3) 004524 012601      MOV     (SP)+,R1 ;: POP STACK INTO R1
(3) 004526 012600      MOV     (SP)+,R0 ;: POP STACK INTO R0

```

```

(1) 004530 012737 004412 000024      MOV      #SPWRDN,2#PWRVEC ;;SET UP THE POWER DOWN VECTOR
(1) 004536 012737 000340 000026      MOV      #340,2#PWRVEC+2 ;;PRIO:7
(1) 004544 104401                      TYPE                      ;;REPORT THE POWER FAILURE
(1) 004546 004560      SPWRMG: .WORD  $POWER          ;;POWER FAIL MESSAGE POINTER
(1) 004550 000002      RTI
(1) 004552 000000      $ILLUP: HALT                ;;THE POWER UP SEQUENCE WAS STARTED
(1) 004554 000776      BR          .-2              ;;BEFORE THE POWER DOWN WAS COMPLETE
(1) 004556 000000      $$AVR6: 0
(1) 004560 005015 047520 042527      $POWER: .ASCIZ <15><12>"POWER" ;;PUT THE SP HERE
(1) 004566 000122

```

(1) .EVEN
670 .SBTTL TRAP DECODER

(1) ;:*****
(1) ;:THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
(1) ;:AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
(1) ;:OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
(1) ;:GO TO THAT ROUTINE.

```

(1) 004570 010046      $TRAP: MOV      RO, -(SP)          ;;SAVE RO
(1) 004572 016600 000002      MOV      2(SP),RO          ;;GET TRAP ADDRESS
(1) 004576 005740      TST      -(RO)            ;;BACKUP BY 2
(1) 004600 111000      MOV      (RO),RO          ;;GET RIGHT BYTE OF TRAP
(1) 004602 006300      ASL      RO                ;;POSITION FOR INDEXING
(1) 004604 016000 004624      MOV      $TRPAD(RO),RO    ;;INDEX TO TABLE
(1) 004610 000200      RTS      RO                ;;GO TO ROUTINE

```

(1) ;:THIS IS USE TO HANDLE THE "GETPRI" MACRO

```

(1) 004612 011646      $TRAP2: MOV      (SP),-(SP)    ;;MOVE THE PC DOWN
(1) 004614 016666 000004 000002      MOV      4(SP),2(SP)      ;;MOVE THE PSW DOWN
(1) 004622 000002      RTI                        ;;RESTORE THE PSW

```

(1) .SBTTL TRAP TABLE

(1) ;:THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
(1) ;:BY THE "TRAP" INSTRUCTION.

	ROUTINE
(3) 004624 004612	\$TRPAD: .WORD \$TRAP2
(3) 004626 003424	\$TYPE ;;CALL=TYPE TRAP+1(104401) TTY TYPEOUT ROUTINE
(3) 004630 004210	\$TYPOC ;;CALL=TYPOC TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
(3) 004632 004164	\$TYPOS ;;CALL=TYPOS TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)
(3) 004634 004224	\$TYPON ;;CALL=TYPON TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)
(3) 004636 002640	\$RDCHR ;;CALL=RDCHR TRAP+5(104405) TTY TYPEIN CHARACTER ROUTINE
(3) 004640 002760	\$RDLIN ;;CALL=RDLIN TRAP+6(104406) TTY TYPEIN STRING ROUTINE
(3) 004642 003264	\$RDOCT ;;CALL=RDOCT TRAP+7(104407) READ AN OCTAL NUMBER FROM TTY

```

671
672 004644 177777      TXCRC: -1
673 004646 000571      571 ;M9301 - YA VERSION
674 004650 000457      457 ;M9301 - YB VERSION
675 004652 000243      243 ;M9301 - YC VERSION

```

676	004654	000635	635
677	004656	000207	207
678	004660	000670	670
679	004662	000132	132
680	004664	000374	374
681	004666	000533	533
682	004670	000536	536
683	004672	177777	-1
684	004674	177777	-1
685	004676	177777	-1
686	004700	177777	-1
687	004702	177777	-1
688	004704	177777	-1
689			
690	004706	177777	-1
691	004710	133725	133725
692	004712	017563	17563
693	004714	141744	141744
694	004716	047613	47613
695	004720	114175	114175
696	004722	146126	146126
697	004724	132161	132161
698	004726	143466	143466
699	004730	036104	036104
700	004732	125411	125411
701	004734	177777	-1
702	004736	177777	-1
703	004740	177777	-1
704	004742	177777	-1
705	004744	177777	-1
706	004746	177777	-1
707	004750	177777	-1
708			
709	004752	177777	-1
710	004754	001000	1000
711	004756	001000	1000
712	004760	001000	1000
713	004762	001000	1000
714	004764	001000	1000
715	004766	004000	4000
716	004770	001000	1000
717	004772	001000	1000
718	004774	001000	1000
719	004776	000734	734
720	005000	177777	-1
721	005002	177777	-1
722	005004	177777	-1
723	005006	177777	-1
724	005010	177777	-1
725	005012	177777	-1
726			
727	005014	177777	-1
728	005016	173000	173000
729	005020	173000	173000
730	005022	173000	173000
731	005024	173000	173000

TXLPC:

TDLN1:

TRMSA1:

:M9400 - YA(OR YC) VERSION
:M9301 - YF VERSION
:M7942 - YB VERSION
:M9301 - YD VERSION
:M9400 - YH (OR YK) VERSION
:M9311 VERSION
:M9301 - YH VERSION

:M9301 - YA VERSION
:M9301 - YB VERSION
:M9301 - YC VERSION
:M9400 - YA(OR YC) VERSION
:M9301 - YF VERSION
:M7942 - YB VERSION
:M9301 - YD VERSION
:M9400 - YH(OR YK) VERSION
:M9311 VERSION
:M9301 - YH VERSION

:M9301 - YA VERSION
:M9301 - YB VERSION
:M9301 - YC VERSION
:M9400 - YA(OR YC) VERSION
:M9301 - YF VERSION
:M7942 - YB VERSION
:M9301 - YD VERSION
:M9400 - YH(OR YK) VERSION
:M9311 VERSION
:M9301 - YH VERSION

:M9301 - YA VERSION
:M9301 - YB VERSION
:M9301 - YC VERSION
:M9400 - YA(OR YC) VERSION

732	005026	173000	173000	;M9301 - YF VERSION
733	005030	170000	170000	;M7942 - YB VERSION
734	005032	173000	173000	;M9301 - YD VERSION
735	005034	173000	173000	;M9400 - YH(OR YK) VERSION
736	005036	163000	163000	;M9311 VERSION
737	005040	173000	173000	;M9301 - YH VERSION
738	005042	177777	-1	
739	005044	177777	-1	
740	005046	177777	-1	
741	005050	177777	-1	
742	005052	177777	-1	
743	005054	177777	-1	
744				
745	005056	177777	TDLN2: -1	
746	005060	001000	1000	;M9301 - YA VERSION
747	005062	001000	1000	;M9301 - YB VERSION
748	005064	001000	1000	;M9301 - YC VERSION
749	005066	001000	1000	;M9400 - YA(OR YC) VERSION
750	005070	001000	1000	;M9301 - YF VERSION
751	005072	000000	0	;M7942 - YB VERSION
752	005074	001000	1000	;M9301 - YD VERSION
753	005076	001000	1000	;M9400 - YH(OR YK) VERSION
754	005100	001000	1000	;M9311 VERSION
755	005102	000764	764	;M9301 - YH VERSION
756	005104	177777	-1	
757	005106	177777	-1	
758	005110	177777	-1	
759	005112	177777	-1	
760	005114	177777	-1	
761	005116	177777	-1	
762				
763	005120	177777	TRMSA2: -1	
764	005122	165000	165000	;M9301 - YA VERSION
765	005124	165000	165000	;M9301 - YB VERSION
766	005126	165000	165000	;M9301 - YC VERSION
767	005130	165000	165000	;M9400 - YA(OR YC) VERSION
768	005132	165000	165000	;M9301 - YF VERSION
769	005134	000000	0	;M7942 - YB VERSION
770				
771	005136	165000	165000	;M9301 - YD VERSION
772	005140	165000	165000	;M9400 - YH(OR YK) VERSION
773	005142	166000	166000	;M9311 VERSION
774	005144	165000	165000	;M9301 - YH VERSION
775	005146	177777	-1	
776	005150	177777	-1	
777	005152	177777	-1	
778	005154	177777	-1	
779	005156	177777	-1	
780	005160	177777	-1	
781				
782	005162	177777	TMSG: -1	
783	005164	006034	MSG1	
784	005166	006051	MSG2	
785	005170	006066	MSG3	
786	005172	006103	MSG4	
787	005174	006123	MSG5	

C03

.MAIN. MACY11 27(1006) 09-FEB-77 10:15 PAGE 7-19
 DZM9AC.P11 09-FEB-77 10:02 TRAP TABLE

SEG 0028

788	005176	006140			MSG6		;M7942 - YB VERSION
789	005200	006155			MSG7		
790	005202	006172			MSG10		
791	005204	006216			MSG11		
792	005206	006226			MSG12		;M9301 - YH VERSION
793	005210	177777			-1		
794	005212	177777			-1		
795	005214	177777			-1		
796	005216	177777			-1		
797	005220	177777			-1		
798	005222	177777			-1		
799							
800							
801	005224	005015	030122	020115	TITL:	.ASCIZ	<15><12>/ROM TEST/
	005232	042524	052123	000			
802	005237	015	052012	050131	GETCRC:	.ASCIZ	<15><12>/TYPE CRC VALUE: /
	005244	020105	051103	020103			
	005252	040526	052514	035105			
	005260	020040	000				
803	005263	015	052012	050131	GETLPC:	.ASCIZ	<15><12>/TYPE LPC VALUE: /
	005270	020105	050114	020103			
	005276	040526	052514	035105			
	005304	020040	000				
804	005307	015	042412	050130	EXCRMG:	.ASCIZ	<15><12>/EXPECTED CRC = /
	005314	041505	042524	020104			
	005322	051103	020103	020075			
	005330	000040					
805	005332	005015	042412	050130	EXLPMG:	.ASCIZ	<15><12><12>/EXPECTED LPC = /
	005340	041505	042524	020104			
	005346	050114	020103	020075			
	005354	000040					
806	005356	005015	047503	050115	ACCRMG:	.ASCIZ	<15><12>/COMPUTED CRC = /
	005364	052125	042105	041440			
	005372	041522	036440	020040			
	005400	000					
807	005401	015	041412	046517	ACLPMG:	.ASCIZ	<15><12>/COMPUTED LPC = /
	005406	052520	042524	020104			
	005414	050114	020103	020075			
	005422	000040					
808	005424	005015	042412	042116	EOTST:	.ASCIZ	<15><12><12>/END OF TEST/
	005432	047440	020106	042524			
	005440	052123	000				
809	005443	015	052012	050131	SA1:	.ASCIZ	<15><12>/TYPE STARTING ADDR. OF 1ST ROM ADDR. SPACE: /
	005450	020105	052123	051101			
	005456	044524	043516	040440			
	005464	042104	027122	047440			
	005472	020106	051461	020124			
	005500	047522	020115	042101			
	005506	051104	020056	050123			
	005514	041501	035105	020040			
	005522	000					
810	005523	015	052012	050131	SA2:	.ASCIZ	<15><12>/TYPE STARTING ADDR. OF 2ND ROM ADDR. SPACE: /
	005530	020105	052123	051101			
	005536	044524	043516	040440			
	005544	042104	027122	047440			
	005552	020106	047062	020104			

	005560	047522	020115	042101	
	005566	051104	020056	050123	
	005574	041501	035105	020040	
	005602	000			
811	005603	015	052012	050131	SIZE1: .ASCIZ <15><12>/TYPE LENGTH (BYTES) OF 1ST ROM ADDR. SPACE: /
	005610	020105	042514	043516	
	005616	044124	024040	054502	
	005624	042524	024523	047440	
	005632	020106	051461	020124	
	005640	047522	020115	042101	
	005646	051104	020056	050123	
	005654	041501	035105	020040	
	005662	000			
812	005663	015	052012	050131	SIZE2: .ASCIZ <15><12>/TYPE LENGTH (BYTES) OF 2ND ROM ADDR. SPACE: /
	005670	020105	042514	043516	
	005676	044124	024040	054502	
	005704	042524	024523	047440	
	005712	020106	047062	020104	
	005720	047522	020115	042101	
	005726	051104	020056	050123	
	005734	041501	035105	020040	
	005742	000			
813	005743	015	000012		CARLF: .ASCIZ <15><12>
814	005746	020040	000		SP2: .ASCIZ / /
815	005751	015	040412	042104	TYPHDR: .ASCIZ <15><12>/ADDRESS DATA/
	005756	042522	051523	020040	
	005764	020040	020040	020040	
	005772	020040	020040	020040	
	006000	042040	052101	000101	
816	006006	020072	000040		COLON: .ASCIZ /: /
817	006012	005015	047125	047113	AUTERM: .ASCIZ <15><12>/UNKNOWN MODULE /
	006020	053517	020116	047515	
	006026	052504	042514	000040	
818					
819	006034	005015	034515	030063	MSG1: .ASCIZ <15><12>/M9301 - YA/
	006042	020061	020055	040531	
	006050	000			
820	006051	015	046412	031471	MSG2: .ASCIZ <15><12>/M9301 - YB/
	006056	030460	026440	054440	
	006064	000102			
821	006066	005015	034515	030063	MSG3: .ASCIZ <15><12>/M9301 - YC/
	006074	020061	020055	041531	
	006102	000			
822	006103	015	046412	032071	MSG4: .ASCIZ <15><12>/M9400 - YA,YC/
	006110	030060	026440	054440	
	006116	026101	041531	000	
823	006123	015	046412	031471	MSG5: .ASCIZ <15><12>/M9301 - YF/
	006130	030460	026440	054440	
	006136	000106			
824	006140	005015	033515	032071	MSG6: .ASCIZ <15><12>/M7942 - YB/
	006146	020062	020055	041131	
	006154	000			
825	006155	015	046412	031471	MSG7: .ASCIZ <15><12>/M9301 - YD/
	006162	030460	026440	054440	
	006170	000104			
826	006172	005015	034515	030064	MSG10: .ASCIZ <15><12>/M9400 - YH(OR YK)/

E03

.MAIN. MACY11 27(1006) 09-FEB-77 10:15 PAGE 7-21
DZM9AC.P11 09-FEB-77 10:02 TRAP TABLE

SEQ 0030

	006200	020060	020055	044131	
	006206	047450	020122	045531	
	006214	000051			
827	006216	005015	034515	030463	MSG11: .ASCIZ <15><12>/M9311/
	006224	000061			
828	006226	005015	034515	030063	MSG12: .ASCIZ <15><12>/M9301 - YH/
	006234	020061	020055	044131	
	006242	000			
829		000001			.END

ABASE = 000000	413						
ACCRMG 005356	501	806*					
ACDW1 = 000000	413						
ACDW2 = 000000	413						
ACLPMG 005401	519	807*					
ACPUOP= 000000	413						
ACTCRC 001022	406*	470*	493	502	544	574*	647
ACTLPC 001024	407*	471*	512	520	599	606*	
ADDW0 = 000000	413						
ADDW1 = 000000	413						
ADDW10= 000000	413						
ADDW11= 000000	413						
ADDW12= 000000	413						
ADDW13= 000000	413						
ADDW14= 000000	413						
ADDW15= 000000	413						
ADDW2 = 000000	413						
ADDW3 = 000000	413						
ADDW4 = 000000	413						
ADDW5 = 000000	413						
ADDW6 = 000000	413						
ADDW7 = 000000	413						
ADDW8 = 000000	413						
ADDW9 = 000000	413						
ADEVCT= 000000	413						
ADEVN = 000000	413						
RENV = 000000	413						
REVM = 000000	413						
RFATAL= 000000	413						
AMADR1= 000000	413						
AMADR2= 000000	413						
AMADR3= 000000	413						
AMADR4= 000000	413						
AMAMS1= 000000	413						
AMAMS2= 000000	413						
AMAMS3= 000000	413						
AMAMS4= 000000	413						
AMSGAD= 000000	413						
AMSGLG= 000000	413						
AMSGTY= 000000	413						
AMTYP1= 000000	413						
AMTYP2= 000000	413						
AMTYP3= 000000	413						
AMTYP4= 000000	413						
APASS = 000000	413						
APRIOR= 000000	413						
APTCSU= 000040	666	667*					
APTEMV= 000001	666	667*					
APTSIZ= 000200	418	667*					
APTSPO= 000100	666	667*					
ASWREG= 000000	413						
ATESTN= 000000	413						
AUNIT = 000000	413						
AUSMR = 000000	413						
AUTACT 002522	492	643*					
AUTERM 006012	659	817*					

AUTERR	002612	646	658#					
AUT1	002524	644#	648					
AUT2	002570	651*	653#					
AUT3	002572	650	654#					
AVECT1=	000000	413						
AVECT2=	000000	413						
BIT0 =	000001	383#						
BIT00 =	000001	383#						
BIT01 =	000002	383#						
BIT02 =	000004	383#						
BIT03 =	000010	383#						
BIT04 =	000020	383#						
BIT05 =	000040	383#						
BIT06 =	000100	383#						
BIT07 =	000200	383#						
BIT08 =	000400	383#						
BIT09 =	001000	383#						
BIT1 =	000002	383#						
BIT10 =	002000	383#						
BIT11 =	004000	383#						
BIT12 =	010000	383#						
BIT13 =	020000	383#						
BIT14 =	040000	383#						
BIT15 =	100000	383#						
BIT2 =	000004	383#						
BIT3 =	000010	383#						
BIT4 =	000020	383#						
BIT5 =	000040	383#						
BIT6 =	000100	383#						
BIT7 =	000200	383#						
BIT8 =	000400	383#						
BIT9 =	001000	383#						
BPTVEC=	000014	383#						
CARLF	005743	626	631	813#				
CHECK	001510	436	438	440	467#			
CHO	001566	473	480#					
CH1	001644	482	491	493#				
CK1	001736	494	507	512#	656			
CK2	002030	513	525	530#				
CLLAST	002204	567	569#					
CLP0	002252	578#	584					
CLP1	002264	579	581#					
CLP2	002316	588	590#					
CLO	002102	545#	551	568				
CL1	002146	559#						
CL2	002170	556	565#					
CL3	002122	547	552#					
COLON	006006	635	816#					
CR =	000015	383#	666					
CRC	002074	475	483	544#				
CRLF =	000200	383#	666					
DATLN1	001006	400#	432*	458*	472	477	612	
DATLN2	001012	402#	434*	466*	481	485	617	
DDISP =	177570	383#	418					
DISPLA	001002	398#	418*					
DISPRE	000174	385#	418					

H03

.MAIN. MACY11 27(1006) 09-FEB-77 10:15 PAGE 8-2
 DZMSAC.P11 09-FEB-77 10:02 CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0033

DSMR =	177570	393#	418					
EMTVEC=	000030	383#						
ENDOT	002446	619	621#					
EOTST	005424	533	622	808#				
ERRVEC=	000004	383#	418*					
EXCRC	001016	404#	431*	446*	493	498	654*	
EXCRMG	005307	497	804#					
EXLPC	001020	405#	430*	450*	512	516	655*	
EXLPMG	005332	515	805#					
GETCRC	005237	444	802#					
GETIN	001370	428	437#					
GETLPC	005263	448	803#					
GET2	001440	442	451#					
GNS =	***** U	385	670					
GOAGIN	002070	536	542#					
HT =	000011	383#	666					
IOTVEC=	000020	383#						
LF =	000012	383#	666					
LPC	002344	479	487	599#				
LPC1	002352	600#	605					
LPC2	002366	602	604#					
MSG1	006034	783	819#					
MSG10	006172	790	826#					
MSG11	006216	791	827#					
MSG12	006226	792	828#					
MSG2	006051	784	820#					
MSG3	006066	785	821#					
MSG4	006103	786	822#					
MSG5	006123	787	823#					
MSG6	006140	788	824#					
MSG7	006155	789	825#					
PARCNT	001026	408#	576*	580*	587			
PARITY	002242	552	576#					
PIRQ =	177772	383#						
PIRQVE=	000240	383#						
PRO =	000000	383#						
PR1 =	000040	383#						
PR2 =	000100	383#						
PR3 =	000140	383#						
PR4 =	000200	383#						
PR5 =	000240	383#						
PR6 =	000300	383#						
PR7 =	000340	383#						
PS =	177776	383#	393#					
PSW =	177776	383#						
PWRVEC=	000024	383#	418*	669*				
ROCHR =	104405	664	670#					
ROLIN =	104406	665	670#					
ROOCT =	104407	445	449	453	457	461	465	670#
RESTRT	001274	420	424#	542				
RESVEC=	000010	383#						
ROMSA1	001004	399#	433*	454*	474	476	611	
ROMSA2	001010	401#	435*	462*	480	484	616	
SA1	005443	452	809#					
SA2	005523	460	810#					
SIZE1	005603	456	811#					

COMMEN	383#																			
ENDCOM	383#																			
ERROR	383#																			
ESCAPE	383#																			
GETPRI	383#																			
GETSMR	383#																			
MULT	383#																			
NEWTST	383#																			
POP	380#	383#	665	667	669															
PUSH	380#	383#	665	667	669															
REPORT	383#																			
SCOPE	383#																			
SETPRI	383#																			
SETTRA	670#																			
SETUP	380#	383#	418																	
SKIP	383#																			
SLASH	383#																			
SPACE	383#																			
STARS	380#	383#	411	412	413	664	665	666	667	668	669	670								
SMRSU	383#	418#																		
TRMTRP	670#																			
TYPBIN	383#																			
TYPDEC	383#																			
TYPNAM	383#																			
TYPNUM	383#																			
TYPOCS	383#																			
TYPOCT	383#																			
TYPTXT	383#																			
SSSCA	383#																			
SSNEW	383#																			
SSSET	670#																			
SSSETH	418#																			
SSSKIP	383#																			
.EQUAT	380#	383																		
.SETUP	380#	386																		
.SACT1	381#	411																		
.SAPT8	381#	413																		
.SAPTH	381#	412																		
.SAPTY	381#	667																		
.SCATC	379#	385																		
.SPOWE	379#	669																		
.SRDOC	379#	665																		
.SREAD	379#	664																		
.STRAP	379#	670																		
.STYPE	379#	666																		
.STYPO	379#	668																		

. ABS. 006243 000

ERRORS DETECTED: 0
 DEFAULT GLOBALS GENERATED: 0

DZM9AC.BIN,DZM9AC.LST/CRF/NL:TOC=DZM9AC.P11
 RUN-TIME: 11 4 .5 SECONDS

M03

.MAIN. MACY11 27(1006) 09-FEB-77 10:15 PAGE 9-1
DZM9AC.P11 09-FEB-77 10:02 CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0038

RUN-TIME RATIO: 272/16=16.8
CORE USED: 20K (40 PAGES)

N03