

.TITLE GT40 ROM BOOTSTRAP TEST MAINDEC-11-CDGTD-D

.REM

IDENTIFICATION

PRODUCT CODE: AC-7924D-MC
PRODUCT NAME: CDGTDD0 GT40 ROM VERIFY
DATE: AUG. 1979
MAINTAINER: DIAGNOSTIC GROUP

COPYRIGHT (C) 1975, 1976, 1979 DIGITAL EQUIPMENT CORP., MAYNARD, MASS.

THIS SOFTWARE IS FURNISHED TO PURCHASER UNDER A LICENSE FOR USE ON A SINGLE COMPUTER SYSTEM AND CAN BE COPIED (WITH INCLUSION OF DEC'S COPYRIGHT NOTICE) ONLY FOR USE IN SUCH SYSTEM, EXCEPT AS MAY OTHERWISE BE PROVIDED IN WRITING BY DEC.

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION.

DEC ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF OF ITS SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.

1. ABSTRACT

THIS VERSION OF THE PROGRAM SUPPORTS NON-SWITCH REGISTER CPU'S.
FOR THESE CPU'S, THE SWITCH REGISTER CAN BE CHANGED BY CHANGING
THE CONTENTS OF SWREG (170).

THE CDGTD DIAGNOSTIC PROGRAM IS WRITTEN TO BE USED AS AN AID
TO HARDWARE DEBUGGING AND MAINTENANCE OF THE GT40 ROM
BOOTSTRAP LOADER VERSION 1, 2 OR 3.

THE AVAILABLE TESTS ARE
PRG0 - LOGIC TESTS
PRG1 - ROM DATA DUMP TO THE CONSOLE TELETYPE

2. REQUIREMENTS

2.1 EQUIPMENT

GT40 DISPLAY PROCESSOR WITH ROM BOOTSTRAP VERSION 1, 2 OR 3.

2.2 STORAGE

THIS PROGRAM USES MEMORY LOCATIONS 0-7776 + 16000-16776(8).

3. LOADING PROCEDURE

PROCEDURE FOR A NORMAL BINARY TAPE SHOULD BE FOLLOWED.

4. STARTING PROCEDURE

4.1 STARTING ADDRESSES

0200 PROGRAM 0, ROM LOGIC TEST.
0204 PROGRAM 1, ROM DATA DUMP ON CONSOLE TTY.

4.2 SWITCH SETTINGS

CONSOLE SW 11=0	NORMAL RUN (64. INTERACTIONS/TEST)
CONSOLE SW 11=1	SUPPRES SUBPROGRAM INTERACTIONS
CONSOLE SW 08=0	TEST AS VERSION 2 OR 3 ROM (512. WORDS)
CONSOLE SW 08=1	TEST AS VERSION 1 ROM (256. WORDS)

5. PROGRAM DESCRIPTIONS

5.1 PRG0 - LOGIC TESTS

THE LOGIC TESTS CONSIST OF 4 ROUTINES TO TEST THE GT40 ROM
BOOTSTRAP LOGIC

5.1.1 ROUTINE DESCRIPTIONS

ROUTINE	TESTS
T1	ADDRESSABILITY OF GT40 ROM BOOTSTRAP
T2	DATA RELIABILITY
T3	THAT GT40 ROM BOOTSTRAP TIMES OUT WHEN REFERENCED BY A DATIP BUS CYCLE
T4	THAT DATA READ FROM THE ROM IS CORRECT

5.2 PRG1 - ROM DATA DUMP

THIS PROGRAM TYPES OUT THE 512./256. WORDS OF ROM DATA ON THE
CONSOLE TELETYPE AND HALTS.

6. ERRORS

THE PROGRAM WILL ONLY HALT ON ERROR. THE PROGRAM DOES NOT
CONTAIN FACILITIES FOR REPORTING ERROR CONDITIONS.
TO PLACE THE PROGRAM INTO A SCOPE LOOP, REPLACE THE ERROR
HALT WITH A NOP.

7. EXECUTION TIME

PRG0 TAKES APPROX. 5 SECONDS PER PASS.
PROG1 N/A
PROG2 N/A

8. XXDP/ACT/APT

THE PROGRAM IS CHAINABLE UNDER XXDP OR ACT BUT WILL NOT
SCRIPT UNDER APT.

-

156
157
158
159
160
161
162
169 000000
170 000034 000034
171 000034 002140
172 000036 000000
173 000046 000046
174 000046 001600
175 000052 000052
176 000052 000000
177
178 104400
179 177564
180 177566
181 177776
182 177570
183 000004
184 000500
185 000170
186 000170 000000
187 000172 177570
188 000200 000200
189 000200 000137 001040
190 000204 000137 001614

.LIST ME,BIN,SEQ,LD
.ENABL ABS,AMA
:LOAD ADDRESS=0200
:DEPRESS START
:STACK POINTER IS AT 500
:EQUATE STATEMENTS
SCOPE=TRAP
TPCSR=177564
TPDBR=177566
PSW=177776
DSWR=177570
ERRVEC=4
STKPTR=500
SWREG: .WORD 0
SWR: DSWR
. =200
JMP PRMTRS
JMP PRG1
.LIST
. =34
SCOPEC
0
. =46
LOGICAL
. =52
0
:END ADDRESS FOR ACT/XXDP
:NO MANUAL INTERVENTION REQUIRED
:ADDRESS OF SWITCH REGISTER
:INITIAL STACK SETTING


```

192
193          001000
194 001000 166000          ROMADD: 166000          ;ROM ADDRESS
195 001002 001000          WORDS: 512.          ;ROM LENGTH
196 001004 006000          IMAGE: START      ;256.          ;ROM IMAGE
197 001006 172002          DSR: 172002      ;STARTA        ;DISPLAY STATUS REGISTER
198 001010 000000          ROSAV: 0          ;CONTENTS OF REGISTERS
199 001012 000000          R1SAV: 0
200 001014 000000          R2SAV: 0
201 001016 000000          R3SAV: 0
202 001020 000000          R4SAV: 0
203 001022 000000          R5SAV: 0
204 001024 000010          FILLER: 10        ;# OF FILLER CHAR
205 001026 000010          FILCNT: 10
206 001030 000000          ICNT: 0
207 001032 000000          DUMP: 0
208 001034 000000          CHARA: 0
209 001036 000000          TERM: 0
210 001040 012706 000500  PRMTRS: MOV      #STKPTR,%6  ;SET STACK PTR
211 001044 004737 002424  JSR      PC,SWITCH  ;CHECK ROM VERSION
212
213          ;PROGRAM 0 LOGIC TESTS
214
215 001050 005037 001030  PRGO:  CLR      ICNT          ;CLEAR PASS COUNT
216 001054 012706 000500  PRGOR: MOV      #STKPTR,%6
217 001060 012737 001054 002214  MOV      #PRGOR,RETURN  ;SET RETURN ADDRESS FOR SCOPE
218
219          ;TEST1 TEST ABILITY TO REFERENCE ROM WITHOUT TIMING OUT
220
221 001066 013700 001000  T1:     MOV      ROMADD,%0    ;GET ROM ADDRESS
222 001072 013701 001002  MOV      WORDS,%1          ;GET ADDRESS COUNTER
223 001076 012737 001136 000004  MOV      #ERROR1,4        ;SET UP TIME OUT VECTOR
224 001104 011003  T1A:    MOV      (0),%3        ;REFERENCE
225 001106 005720  TST      (0)+              ;ROM
226 001110 064037 001032  ADD      -(0),DUMP        ;
227 001114 021010  CMP      (0),(0)          ;
228 001116 132020  BITB    (0)+,(0)+        ;
229 001120 164037 001032  SUB      -(0),DUMP        ;
230 001124 062700 000002  ADD      #2,%0           ;INCREMENT POINTER
231 001130 005301  DEC      %1              ;DECREMENT ADDRESS COUNTER
232 001132 001364  BNE     T1A              ;BRANCH IF NOT FINISHED
233 001134 000405  BR      T1B              ;GO TO SCOPE LOOP
234 001136 022626  ERROR1: CMP     (6)+,(6)+  ;REPOSITION STACK
235 001140 004737 002372  JSR     PC,SAVE05        ;SAVE R0-R5
236 001144 000000  HALT                    ;ERROR, TIME-OUT ON ROM ADDRESS
237 001146 000756  BR      T1A              ;LOOP ON ERROR
238 001150 104400  T1B:    SCOPE
239

```



```

241
242 ;TEST2 TEST THAT ROM DATA CAN BE READ RELIABLY.
243
244 001152 013700 001000 T2: MOV ROMADD,%0 ;GET ROM ADDRESS
245 001156 013701 001002 MOV WORDS,%1 ;GET ADDRESS COUNTER
246 001162 012737 000006 000004 MOV #6,4 ;INITIALIZE TIME OUT VECTOR
247 001170 005037 001032 T2A: CLR DUMP ;INITIALIZE DUMP
248 001174 011003 MOV (0),%3 ;GET DATA
249 001176 062037 001032 ADD (0)+,DUMP ;ADD DATA TO DUMP
250 001202 163703 001032 SUB DUMP,%3 ;SUBTRACT DATA FROM DATA
251 001206 001404 BEQ T2B ;BRANCH IF EQUAL
252 001210 004737 002372 ERROR2: JSR PC,SAVE05 ;SAVE R0-R5
253 001214 000000 HALT ;DATA ERROR
254 001216 000764 BR T2A ;LOOP ON ERROR
255 001220 044037 001032 T2B: BIC -(0),DUMP ;CLEAR DUMP BITS
256 001224 001404 BEQ T2C ;BRANCH IF EQUAL TO 0
257 001226 004737 002372 JSR PC,SAVE05 ;SAVE R0-R5
258 001232 000000 HALT ;DATA ERROR
259 001234 000771 BR T2B ;LOOP ON ERROR
260 001236 021010 T2C: CMP (0),(0) ;COMPARE DATA
261 001240 001404 BEQ T2D ;BRANCH IF EQUAL
262 001242 004737 002372 JSR PC,SAVE05 ;SAVE R0-R5
263 001246 000000 HALT ;DATA ERROR
264 001250 000772 BR T2C ;LOOP ON ERROR
265 001252 122040 T2D: CMPB (0)+,-(0) ;COMPARE DATA (BYTE OPERATION)
266 001254 001404 BEQ T2E ;BRANCH IF EQUAL
267 001256 004737 002372 JSR PC,SAVE05 ;SAVE R0-R5
268 001262 000000 HALT ;DATA ERROR
269 001264 000772 BR T2D ;LOOP ON ERROR
270 001266 005720 T2E: TST (0)+ ;INCREMENT ADDRESS POINTER
271 001270 005301 DEC %1 ;DECREMENT ADDRESS COUNTER
272 001272 001336 BNE T2A ;RETURN IF NOT DONE
273 001274 104400 SCOPE
274

```



```

276
277
278
279
280 001276 012706 000500 T3: MOV #STKPTR,%6 ;SET STACK PTR
281 001302 013700 001000 MOV ROMADD,%0 ;GET ROM ADDRESS
282 001306 013701 001002 MOV WORDS,%1 ;GET ADDRESS COUNTER
283 001312 012737 001332 000004 T3AA: MOV #T3B,%4 ;SET UP TIME OUT VECTOR
284 001320 010010 T3A: MOV %0,(0) ;ATTEMPT TO ALTER DATA
285 001322 004737 002372 JSR PC,SAVE05 ;SAVE R0-R5
286 001326 000000 HALT ;HERE IF DID NOT TIME OUT
287 001330 000773 BR T3A ;LOOP ON ERROR
288 001332 012737 001354 000004 T3B: MOV #T3D,%4 ;SET UP TIME OUT VECTOR
289 001340 022626 CMP (6)+,(6)+ ;REPOSITION STACK
290 001342 005210 T3C: INC (0) ;ATTEMPT TO ALTER DATA
291 001344 004737 002372 JSR PC,SAVE05 ;SAVE R0-R5
292 001350 000000 HALT ;HERE IF DID NOT TIME OUT
293 001352 000773 BR T3C ;LOOP ON ERROR
294 001354 012737 001400 000004 T3D: MOV #T3F,%4 ;SET UP TIME OUT VECTOR
295 001362 022626 CMP (6)+,(6)+ ;REPOSITION STACK
296 001364 005077 177410 T3E: CLR @ROMADD ;ATTEMPT TO ALTER DATA
297 001370 004737 002372 JSR PC,SAVE05 ;SAVE R0-R5
298 001374 000000 HALT ;HERE IF DID NOT TIME OUT
299 001376 000772 BR T3E ;LOOP ON ERROR
300 001400 005720 T3F: TST (0)+ ;INCREMENT ADDRESS POINTER
301 001402 022626 CMP (6)+,(6)+ ;REPOSITION STACK
302 001404 005301 DEC %1 ;DECREMENT ADDRESS COUNTER
303 001406 001341 BNE T3AA ;RETURN IF NOT DONE
304 001410 012737 000006 000004 MOV #6,@#4 ;RESTORE TIME OUT TRAP
305 001416 104400 SCOPE ;SCOPE LOOP
306
307 ;COMPARE THE ROM DATA TO THE IMAGE DATA
308 ;
309 ; R0=WORD NUMBER
310 ; R1=GOOD ADDRESS
311 ; R2=GOOD DATA
312 ; R3=BAD ADDRESS
313 ; R4=BAD DATA
314 001420 012700 000000 T4: MOV #0,%0 ;SET UP INITIAL WORD COUNT
315 001424 013701 001004 MOV IMAGE,%1 ;SET UP STARTING ADDRESS OF ROM IMAGE
316 001430 013703 001000 MOV ROMADD,%3 ;SET UP STARTING ROM ADDRESS
317 001434 011102 T4A: MOV (%1),%2 ;READ EXPECTED VALUE
318 001436 011304 MOV (%3),%4 ;READ ROM VALUE
319 001440 020204 CMP %2,%4 ;COMPARE EXPECTED TO THE VALUE READ
320 001442 001422 BEQ T4E ;BRANCH IF CORRECT
321 001444 012705 002536 MOV #TABLE0,%5 ;LOAD PATCH POINTER
322 001450 020015 T4B: CMP %0,(%5) ;TEST IF PATCHED LOCATION
323 001452 001406 BEQ T4C ;BR IF IT IS
324 001454 020527 002652 CMP %5,#TABLEX ;TEST IF END OF PATCH TABLE
325 001460 001407 BEQ T4D ;BR IF END OF TABLE * MUST BE A BAD ROM WORD
326 001462 062705 000004 ADD #4,%5 ;ADJUST PATCH POINTER
327 001466 000770 BR T4B ;TRY AGAIN
328 001470 005725 T4C: TST (%5)+ ;POINT TO PATCH WORD VALUE
329 001472 011502 MOV (%5),%2 ;GET PATCHED WORD VALUE
330 001474 020204 CMP %2,%4 ;COMPARE EXPECTED TO VALUE READ
331 001476 001404 BEQ T4E ;BR IF CORRECT

```



```

332 001500 004737 002372      T4D:  JSR      PC,SAVE05      ;SAVE CONTENTS OF R0 - R5
333 001504 000000              HALT                    ;ERROR, ROM VALUE FAILED TO EQUAL EXPECTED
334 001506 000752              BR       T4A
335
336 001510 022123              T4E:  CMP      (%1)+,(%3)+    ;INCREMENT ADDRESSES POINTERS
337 001512 005200              INC      %0              ;INCREMENT WORD COUNT
338 001514 023700 001002      CMP      WORDS,%0        ;COMPARE IF END WORD
339 001520 001345              BNE     T4A              ;BRANCH IF NOT LAST WORD
340 001522 104400              SCOPE
341
342 001524 005237 001030      END:   INC      ICNT        ;INCREMENT PASS COUNT
343 001530 012777 000001 177250  MOV     #1,@DSR          ;RING THE GT40 BELL
344 001536 012737 000207 177566  DONE0: MOV     #207,@#TPDBR   ;RING THE TELETYPE BELL
345 001544 105737 177564      TSTB   @#TPCSR
346 001550 100375              BPL     -4
347 001552 012737 000207 177566  MOV     #207,@#TPDBR
348 001560 105737 177564      1$:   TSTB   @#TPCSR
349 001564 100375              BPL     1$
350 001566 013700 000042      MOV     @#42,%0          ;RETURN TO XXDP/ACT MONITOR?
351 001572 001406              BEQ     DONE1
352 001574 000005              RESET
353 001576 000005              RESET
354 001600 004710              LOGICAL: JSR      7,(0)      ;RETURN!
355 001602 000240              NOP
356 001604 000240              NOP
357 001606 000240              NOP
358 001610 000137 001050      DONE1: JMP     PRG0
359
  
```



```

361
362 ;THIS PROGRAM TYPES OUT ROM DATA
363
364 001614 012706 000500 PRG1: MOV #STKPTR,%6 ;INITIALIZE STACK
365 001620 012737 000006 000004 MOV #6,%4 ;SET UP BUSS ERROR
366 001626 004737 002424 JSR PC,SWITCH
367 001632 004537 001776 JSR 5,TYPEM
368 001636 002361 M8
369 001640 004537 001776 JSR 5,TYPEM ;TYPE MESSAGE
370 001644 002336 M7 ;'ROM DATA'
371 001646 013701 001002 MOV WORDS,%1 ;GET # OF WORDS
372 001652 013700 001000 PRG1A: MOV ROMADD,%0 ;GET STARTING ADDRESS
373 001656 012702 000010 MOV #10,%2 ;GET ADDRESS INDICATOR
374 001662 105737 177564 TSTB TPCSR ;WAIT FOR
375 001666 100375 BPL -4 ;TELEPRINTER FLAG
376 001670 010037 002230 PRG1B: MOV %0,D2BTYP ;GET ADDRESS
377 001674 004737 002232 JSR 7,02A ;AND TYPE IT
378 001700 004537 001776 JSR 5,TYPEM ;TYPE
379 001704 002365 M9 ;CR/LF
380 001706 012037 002230 PRG1C: MOV (0)+,D2BTYP ;TYPE
381 001712 004737 002232 JSR 7,02A ;DATA
382 001716 105737 177564 TSTB TPCSR ;WAIT FOR
383 001722 100375 BPL -4 ;TELEPRINTER FLAG
384 001724 012737 000040 177566 MOV #' ',TPDBR ;TYPE SPACE
385 001732 005301 DEC %1 ;ALL DATA TYPED
386 001734 001410 BEQ PRG1D ;GO TO FINISH
387 001736 005302 DEC %2
388 001740 001362 BNE PRG1C ;RETURN TO PRG1B
389 001742 012702 000010 MOV #10,%2 ;GET ADDRESS INDICATOR
390 001746 004537 001776 JSR 5,TYPEM ;TYPE
391 001752 002361 M8 ;CR/LF
392 001754 000745 BR PRG1B ;RETURN TO PRG1B
393 001756 004537 001776 PRG1D: JSR 5,TYPEM
394 001762 002361 M8
395 001764 004537 001776 JSR 5,TYPEM
396 001770 002361 M8
397 001772 000000 HALT
398 001774 000707 BR PRG1
399
400 ;ROUTINE TO LOOP ON A SINGLE ADDRESS
401

```



```

403 ;ROUTINE TO TYPE A MESSAGE
404
405 001776 010026 TYPEM: MOV %0,(6)+ ;SAVE REGISTER 0
406 002000 012500 MOV (5)+,%0 ;PLACE MESSAGE ADDRESS IN R0
407 002002 112037 001036 MOVB (0)+,TERM ;GET TERMINATOR CHARACTER
408 002006 112037 001034 TYPEMA: MOVB (0)+,CHARA ;GET NEXT CHARACTER
409 002012 123737 001034 001036 CMPB CHARA,TERM ;WAS NEXT CHARACTER THE TERM
410 002020 001005 BNE TYPEMB ;CHARACTER
411 002022 014600 MOV -(6),%0 ;RESTORE R0
412 002024 105737 177564 TSTB TPCSR
413 002030 100375 BPL .-4
414 002032 000205 RTS 5 ;AND EXIT
415 002034 123727 001034 000045 TYPEMB: CMPB CHARA,#'% ;WAS CHARACTER %
416 002042 001027 BNE TYPEMC
417 002044 105737 177564 TSTB TPCSR ;TEST TELEPRINTER FLAG
418 002050 100375 BPL .-4 ;AND WAIT FOR DONE
419 002052 012737 000215 177566 MOV #215,TPDBR ;LOAD TELEPRINTER WITH CAR. RET
420 002060 013737 001024 001026 MOV FILLER,FILCNT ;LOAD FILLER COUNT
421 002066 000403 BR 1$
422 002070 012737 000006 177566 2$: MOV #6,TPDBR ;PRINT FILLER CHAR
423 002076 105737 177564 1$: TSTB TPCSR ;TEST TELEPRINTER FLAG
424 002102 100375 BPL .-4 ;AND WAIT FOR DONE
425 002104 005337 001026 DEC FILCNT ;FINISHED FILLERS ?
426 002110 001367 BNE 2$ ;BR IF NOT
427 002112 012737 000212 177566 MOV #212,TPDBR ;LOAD TELEPRINTER WITH LINE FEED
428 002120 000732 BR TYPEMA ;GET NEXT CHARACTER
429 002122 105737 177564 TYPEMC: TSTB TPCSR ;TEST TELEPRINTER FLAG
430 002126 100375 BPL .-4 ;AND WAIT FOR DONE
431 002130 013737 001034 177566 MOV CHARA,TPDBR ;LOAD TELEPRINTER BUFFER
432 002136 000723 BR TYPEMA ;AND GET NEXT CHARACTER
433
434 ;SCOPE ROUTINE. THIS ROUTINE IS ENTERED AT THE END OF EACH SUBTEST.
435
436 002140 032777 040000 176024 SCOPEC: BIT #4000,@SWR ;TEST SR FOR SCOPE
437 002146 001023 BNE SCOPEB ;YES SCOPE
438 002150 032777 004000 176014 BIT #4000,@SWR ;TEST FOR ITERATION
439 002156 001007 BNE SCOPEG ;INHIBIT ITERATION
440 002160 023737 002212 002210 CMP SCOPEF,ICOUNT ;ITERATION COMPLETE
441 002166 001403 BEQ SCOPEG ;ITERATION COMPLETE GO TO SCOPEG
442 002170 005237 002212 INC SCOPEF ;INCREMENT ITERATION COUNT
443 002174 000410 BR SCOPEB ;GO TO SCOPEB
444 002176 005037 002212 SCOPEG: CLR SCOPEF ;CLEAR ITERATION COUNT
445 002202 011637 002214 MOV @%6,RETURN ;GET ADDRESS OF NEXT TEST
446 002206 000002 RTI ;EXIT
447 002210 000100 ICOUNT: 100
448 002212 000000 SCOPEF: 0 ;CONTAINS SUBTEST ITERATION COUNT
449 002214 000000 RETURN: .WORD 0 ;CONTAINS RETURN PC FOR SCOPE
450 002216 005726 SCOPEB: TST (6)+ ;POP PC
451 002220 012637 177776 MOV (6)+,PSW ;RESTORE CONDITION CODES
452 002224 000177 177764 JMP @RETURN
453

```



```

455
456
457           ;THIS ROUTINE CONVERTS AN OCTAL NUMBER TO ASCII AND TYPES IT ON THE TTY.
458
459 002230 000000          D2BTYP: 0
460 002232 013746 177564 O2A:  MOV     TPCSR,-(6)      ;SAVE TPCSR
461 002236 010246          MOV     %2,-(6)      ;SAVE R2
462 002240 010146          MOV     %1,-(6)      ;SAVE R1
463 002242 010046          MOV     %0,-(6)      ;SAVE R0
464 002244 013700 002230  MOV     D2BTYP,%0    ;GET DATA TO BE TYPED
465 002250 012701 000006  MOV     #6,%1        ;GET COUNTER
466 002254 005002          CLR     %2           ;CLEAR WORKING REGISTER
467 002256 006100          ROL    %0           ;MOV FIRST BIT (MSB) INTO
468 002260 006102          ROL    %2           ;R2
469 002262 062702 000260  O2AA:  ADD     #260,%2     ;FORM ASCII CODE
470 002266 105737 177564  TSTB   TPCSR        ;TEST TELEPRINTER
471 002272 100375          BPL    .-4          ;FLAG AND WAIT UNTIL DONE
472 002274 010237 177566  MOV     %2,TPDBR    ;LOAD TELEPRINTER BUFFER
473 002300 005002          CLR    %2           ;CLEAR WORKING REGISTER
474 002302 006100          ROL    %0           ;ROTATE THE
475 002304 006102          ROL    %2           ;NEXT
476 002306 006100          ROL    %0           ;OCTAL CHARACTER
477 002310 006102          ROL    %2           ;INTO
478 002312 006100          ROL    %0           ;REGISTER
479 002314 006102          ROL    %2           ;TWO
480 002316 005301          DEC    %1           ;DECREMENT COUNTER
481 002320 001360          BNE    O2AA        ;GO TO O2AA IF NOT 0
482 002322 012600          MOV    (6)+,%0     ;FINISHED. RESTORE REGISTERS
483 002324 012601          MOV    (6)+,%1     ;
484 002326 012602          MOV    (6)+,%2     ;
485 002330 012637 177564  MOV    (6)+,TPCSR  ;AND TPCSR
486 002334 000207          RTS     7           ;AND EXIT
487
488
489 002336 022500 052107 032055 ;ASCII MESSAGES
002344 020060 047522 020115 M7:  .ASCII  '%GT-40 ROM DATA%'
002352 040504 040524 022445
002360 100
490 002361 100 022445 100 M8:  .ASCII  '%a%'
491 002365 100 020040 100 M9:  .ASCII  '% a %'
492 002372
493
494           ;SUBROUTINE TO SAVE R0-R5 NEAR LOC. 1000
495 002372 010037 001010  SAVE05: MOV    R0,R0SAV
496 002376 010137 001012  MOV    R1,R1SAV
497 002402 010237 001014  MOV    R2,R2SAV
498 002406 010337 001016  MOV    R3,R3SAV
499 002412 010437 001020  MOV    R4,R4SAV
500 002416 010537 001022  MOV    R5,R5SAV
501 002422 000207          RTS     PC           ;EXIT
    
```

```

503 002424 013746 000004          SWITCH: MOV    @#ERRVEC,-(SP)    ;SAVE VECTORS CONTENTS
504 002430 012737 002456 000004    MOV    #1$,@#ERRVEC    ;SET UP FOR TRAP
505 002436 012737 177570 000172    MOV    #DSWR,@#SWR    ;SET UP TO TEST FOR SWITCH REGISTER
506 002444 022777 177777 175520    CMP    #-1,@SWR      ;TEST FOR SWITCH REGISTER
507 002452 001005          BNE    3$             ;SWITCH REGISTER IS PRESENT
508 002454 000401          BR     2$             ;NO SWITCH REGISTER
509 002456 022626          1$:  CMP    (SP)+,(SP)+    ;POP 2 WORDS OFF STACK
510 002460 012737 000170 000172    2$:  MOV    #SWREG,@#SWR ;SET UP FOR SOFTWARE SWITCH REGISTER
511 002466 012637 000004          3$:  MOV    (SP)+,@#ERRVEC ;RESTORE VECTORS CONTENTS
512 002472 032777 000400 175472    BIT    #400,@SWR     ;TEST BIT 8
513 002500 001007          BNE    4$             ;BR IF VERSION 1
514 002502 012737 001000 001002    MOV    #512,WORDS    ;SET UP VERSION 2 LENGTH
515 002510 012737 006000 001004    MOV    #START,IMAGE  ;SET UP VERSION 2 STARTING ADD.
516 002516 000406          BR     5$
517 002520 012737 000400 001002    4$:  MOV    #256,WORDS    ;SET UP VERSION 1 LENGTH
518 002526 012737 016000 001004    MOV    #STARTA,IMAGE ;SET UP VERSION 1 STARTING ADD.
519 002534 000207          5$:  RTS    PC

```

520
 521
 522 .SBTTL ROM VERSION 3 PATCH VALUE FOR VERSION 2

```

523 002536 000001 000137          TABLE: .WORD 1,137
524 002542 000002 167636          .WORD 2,167636
525 002546 000003 000000          .WORD 3,0
526 002552 000327 000137          .WORD 215.,137
527 002556 000330 167620          .WORD 216.,167620
528 002562 000710 105737          .WORD 456.,105737
529 002566 000711 175614          .WORD 457.,175614
530 002572 000712 100375          .WORD 458.,100375
531 002576 000713 112337          .WORD 459.,112337
532 002602 000714 175616          .WORD 460.,175616
533 002606 000715 000203          .WORD 461.,203
534 002612 000716 000000          .WORD 462.,0
535 002616 000717 005037          .WORD 463.,5037
536 002622 000720 177776          .WORD 464.,177776
537 002626 000721 012737          .WORD 465.,12737
538 002632 000722 010007          .WORD 466.,10007
539 002636 000723 175610          .WORD 467.,175610
540 002642 000724 000137          .WORD 468.,137
541 002646 000725 166010          .WORD 469.,166010

```

542 002652 000000 TABLEX: 0
 543 .SBTTL ROM VERSION 2 VALUES

545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598

```
*****  
: EXCEPT FOR THE NEW ORGIN ADDRESS AND SEVERAL '!160000'  
: FOR ADDRESS FUDGING THIS IS AN EXACT COPY OF THE CONTENTS  
: OF THE GT-40 BOOTSTRAP VERSION #2  
:*****
```

.TITLE SCROLLING ROM BOOTSTRAP FOR THE GT40

; BOOTGT.T16 OCT 10, 1973

; COPYRIGHT 1973, DIGITAL EQUIPMENT CORPORATION
; 146 MAIN STREET
; MAYNARD, MASSACHUSETTS 01754

; WRITTEN BY JACK BURNES.

; THIS PROGRAM IS THE SECOND VERSION THE THE ROM BOOTSTRAP FOR
; THE GT40 DISPLAY TERMINAL. IT INCLUDES SCROLLING AND AN END OF
; MEMORY SEARCH FOR THE LOADER.

.ENABL ABS,AMA ;ASSEMBLER DIRECTIVES FOR ABSOLUTE BINARY OUTPUT
; NOTE: USE 'MACDLX' TO ASSEMBLE THIS PROGRAM.

.SBTTL DEFINITION SECTION
.PAGE

599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652

REGISTER DEFINITIONS

BASIC DEFINITIONS

000000	R0=%0	;DEFINE STANDARD VALUES.
000001	R1=%1	
000002	R2=%2	
000003	R3=%3	
000004	R4=%4	
000005	R5=%5	
000006	SP=%6	
000007	PC=%7	

GT40 DEFINITIONS

000000	CHAR=R0	;CONTAINS THE INPUT CHARACTER.
000001	POINTR=R1	;POINTS TO NEXT INSERTION BYTE IN DISPLAY BUFFER
000002	TABCNT=R2	;CHARACTER COUNTER FOR THE 'TAB' FEATURE.
000003	SCAN=R3	;GENERALLY CONTAINS A POINTER WHICH
		;IS USED WHEN SCANNING MEMORY FOR SOMETHING.
000004	HOLD=R4	;TYPICALLY A TEMPORARY WHICH IS USED TO RETAIN
		;A VALUE FOR A SHORT TIME.
000005	COUNTR=R5	;TYPICALLY USED AS A COUNTER.

LOADER DEFINITIONS

000000	L.BYT=CHAR	;CHARACTER INPUT FOR THE LOADER.
000001	L.ADR=POINTR	;CURRENT MEMORY ADDRESS TO BE LOADED.
000002	L.BC=TABCNT	;NUMBER OF DATA ITEMS TO LOAD.
000005	L.CKSM=COUNTR	;CHECKSUM ON THE INPUT DATA.
000003	INDEX=SCAN	;INDICATES HOW TO ASSEMBLE THE 8 BIT CHARACTER.

.PAGE

653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705

:
:

MAJOR SYSTEM DEFINITIONS

166000	ORIGIN=166000	;ORIGIN OF THE BOOTSTRAP.
175610	DL11IS=175610	;INPUT STATUS REGISTER OF DL11
175612	DL11IB=DL11IS+2	;INPUT CHARACTER FROM DL11
175614	DL11OS=DL11IB+2	;OUTPUT STATUS OF THE DL11
175616	DL11OB=DL11OS+2	;OUTPUT CHARACTER TO THE DL11
177560	KBDIS=177560	;KEYBOARD INPUT STATUS
177562	KBDIB=KBDIS+2	;CURRENT CHARACTER FROM KEYBOARD.
172000	GT40PC=172000	;GT40 PROGRAM COUNTER.
172002	GT40SR=GT40PC+2	;GT40 STATUS REGISTER ADDRESS.
001000	BSTART=1000	;START OF THE DISPLAY BUFFER
007000	BLIMIT=7000	;APPROXIMATE END OF THE DISPLAY BUFFER.
007776	TMPEND=7776	;LOCATION OF INITIALIZATION STACK.
000004	CORSTR=4	;LOCATION OF PDP-11 TRAP VECTOR.
007012	JMPADD=BLIMIT+10.	;WHERE THE POINTER IS TO FIRST CHAR ON SCREEN
000040	NUMLIN=32.	;NUMBER OF LINES ON TEXT TO SHOW ON THE SCREEN
005015	CRLF=5015	;CARRIAGE RETURN - LINE FEED
000175	ALTMOD=175	;THE 'KEY' CHARACTER [I.E. ALTMODE].
160000	DISJMP=160000	;THE GT40 JMP INSTRUCTION
173000	DISTOP=173000	;THE GT40 STOP DISPLAY INSTRUCTION.

.SBTTL INITIALIZATION AND RESTART CODE
.PAGE

```

706
707
708
709          :          GT40 BOOTSTRAP CODE
710          :          -----
711
712
713
714
715          006000          .=6000
716
717          :          .=ORIGIN          ;DEFINE ORIGIN OF THE BOOTSTRAP.
718
719
720
721
722
723
724          :          COLD INITIALIZATION CODE
725          :          -----
726
727
728
729 006000 000005          START: RESET          ;RESET ALL HARDWARE NOW.
730 006002 012737 000007 175610          MOV #7,DL11IS          ;INITIALIZE DL-11 INPUT NOW.
731 006010 012706 007776          MOV #TMPEND,SP          ;ESTABLISH A GOOD TEMPORARY STACK
732                                     ;POINTER FOR CORE SEARCH.
733 006014 005237 175614          INC DL110S          ;SET BREAK BIT
734 006020 004337 166652          JSR SCAN,OUTLIT!160000 ;FOR 2 CHARACTER TIMES
735 006024 000000          .WORD 0          ;SEND TWO ZERO'S
736
737 006026 012703 000004          MOV #CORSTR,SCAN          ;GET ADDRESS OF BAD CORE TRAP VECTOR.
738 006032 012723 166042          MOV #NOTHERE!160000,(SCAN)+ ;AND INSERT A POINTER TO US THERE.
739
740 006036 005023          ENDCOR: CLR (SCAN)+          ;NOW CLEAR ALL OF MEMORY BEYOND THE POINTER,
741 006040 000776          BR ENDCOR          ;UNTIL WE RUN OUT OF MEMORY AND TRAP.
742
743
744 006042 005743          NOTHER: TST -(SCAN)          ;WHEN WE TRAP OUT, WE COME HERE.
745                                     ;WE BACK UP POINTER TO GOOD CORE.
746                                     ;NOTE THAT IF WE TRAP OUT AGAIN, IT
747                                     ;IS STILL OK, BECAUSE WE WILL LOOP
748                                     ;UNTIL WE GET A GOOD CORE ADDRESS.
749 006044 010306          MOV SCAN,SP          ;WHEN WE GET ONE, THAT IS LAST LOCATION
750                                     ;IN THE MACHINE, AND HENCE OUR SP.
751 006046 105737 175614          1$: TSTB DL110S          ;SEE IF BREAK IS DONE
752 006052 100375          BPL 1$          ;NO GO BACK
753 006054 005037 175614          CLR DL110S          ;CLEAR BREAK BIT
754
755
756
757
758
759
760          :          RESTART INITIALIZATION CODE WHEN COMMUNICATIONS IS WORKING.
761          :          -----
  
```



```

762
763
764
765 006060 052706 007776      RESTRT: BIS      #TMPEND,SP      ;FORCE THE SP TO LIMIT OF EXISTING CORE.
766
767
768 006064 012703 006700      MOV      #BLIMIT-NUMLIN-NUMLIN,SCAN      ;NOW WE WILL FILL THE KEY AREAS OF THE
769 006070 012702 000040      MOV      #NUMLIN,TABCNT      ;DISPLAY BUFFER WITH INITIAL CR-LF'S.
770
771 006074 012723 005015      SETLP1: MOV      #CRLF,(SCAN)+      ;INSERT A CRLF NOW.
772 006100 005302              DEC      TABCNT      ;AND LOOP UNTIL DONE.
773 006102 003374              BGT      SETLP1      ;THUS DISPLAY CORE IS ALMOST CORRECT.
774
775
776 006104 012703 166432      MOV      #SETUP!160000,SCAN      ;NOW WE WILL INITIALIZE CORE FOR THE
777                                ;DISPLAY. PICK UP POINTER TO LIST.
778
779 006110 012302      SETLP2: MOV      (SCAN)+,TABCNT      ;GET NUMBER OF ITEMS TO INSERT.
780 006112 001405      BEQ      SETDUN      ;IF ZERO, WE ARE DONE.
781 006114 012301      MOV      (SCAN)+,POINTR      ;PICK UP FIRST CORE ADDRESS POINTER.
782
783 006116 012321      SETLP3: MOV      (SCAN)+,(POINTR)+      ;MOVE OVER A DATA ITEM NOW.
784 006120 005302      DEC      TABCNT      ;ALL DONE?
785 006122 003375      BGT      SETLP3      ;NOPE. MOVE OVER THE NEXT.
786 006124 000771      BR       SETLP2      ;YES. GET NEXT MAJOR LIST TO INSERT.
787
788
789 006126 012701 006776      SETDUN: MOV      #BLIMIT-2,POINTR      ;ESTABLISH THE BUFFER POINTER NOW.
790
791
792
793
794
795
796
797
798
799
800
801
802

```

```

.SBTTL VT05 SIMULATOR
.PAGE

```

803
 804
 805
 806
 807
 808
 809
 810
 811
 812
 813
 814
 815
 816
 817
 818
 819
 820
 821
 822
 823
 824
 825
 826
 827
 828
 829
 830
 831
 832
 833
 834
 835
 836
 837
 838
 839
 840
 841
 842
 843
 844
 845
 846
 847
 848
 849
 850
 851
 852
 853
 854
 855
 856
 857
 858

VT05 (SCROLLING) PORTION OF THE BOOTSTRAP

```

NXTCHR: JSR PC,GETCHR!160000          ;GET A CHARACTER NOW.
          CMP CHAR,#177              ;IS IT OUT OF RANGE?
          BGE NXTCHR                 ;YEP. GET ANOTHER ONE.
          CMP CHAR,#40               ;IS IT A PRINTING CHARACTER?
          BGE NORMAL                 ;YES. IT'S A NORMAL PRINTING CHARACTER.
          MOV CHAR,SCAN              ;MOVE IT OVER SO WE CAN PLAY WITH IT.
          SUB #7,SCAN                ;BIAS SO THAT BELL [7] IS ZERO.
          CMP SCAN,#7                ;IF CHARACTER IS LESS THEN BELL OR
          BHIS NXTCHR                ;GREATER THEN CR, THEN IGNORE.
          ASL SCAN                    ;IF GOOD, MAKE IT WORD INDEX.
          ADD SCAN,PC                 ;AND GO TO THE CORRECT ROUTINE.

          BR BELL                     ;7=BELL
          BR NORMAL                   ;10=BACKSPACE
          BR TAB                       ;11=TAB
          BR LF                        ;12=LINE FEED [LF]
          BR VT                        ;13=VERTICAL TAB [VT]
          BR FF                        ;14=FORM FEED [FF]
          BR CR                        ;15=CARRIAGE RETURN [CR]

CR:      MOV #-1,TABCNT               ;RESET TAB POSITION ON A CR, AND
          ;FALL THROUGH TO INSERT THE CHARACTER.

NORMAL: JSR PC,INSERT!160000         ;INSERT THE CHARACTER IN THE BUFFER.
          INC TABCNT                 ;UPDATE TAB POSITION NOW.
          BR NXTCHR                  ;AND GET NEXT CHARACTER.

TAB:     MOV #40,CHAR                 ;ON A TAB, INSERT BLANKS UNTIL THE
          JSR PC,INSERT!160000         ;NEXT CHARACTER POSITION IS A MULTIPLE
          INC TABCNT                 ;OF .8.
          BIT #7,TABCNT               ;ARE WE DONE YET?
          BNE TAB                     ;NOPE.
          BR NXTCHR                   ;YES.

VT:      MOV# (PC),COUNTR             ;THIS PUTS THE LOW BYTE OF THE
          BR FFLOOP                   ;BRANCH CODE IN COUNTR-SAVE A WORD

BELL:   CLR GT40SR                   ;RING BELL -WRITE IN GT40SR
          BR NXTCHR                   ;AND LOOP BACK

FF:     MOV #NUMLIN,COUNTR           ;FORM FEED IS DONE BY INSERTING LF'S.
  
```



```

859
860 006262 0127C3 000012      FFLOOP: MOV      #12,CHAR      ;MAKE THE CHARACTER A LINEFEED.
861 006266 004737 166304      JSR      PC,LFSUB!160000      ;DO A LINEFEED.
862 006272 005305              DEC      COUNTR              ;DONE?
863 006274 003372              BGT      FFLOOP              ;NOPE. KEEP SENDING THEM.
864 006276 C00715              BR       NXTCHR              ;YES. NOW RETURN. DO NOT FALL THROUGH.
865
866
867 006300 012746 166132      LF:      MOV      #NXTCHR!160000,-(SP)      ;RETURN TO NXTCHR AFTER PROCESSING
868                                         ;THE LF BY FAKING A JSR.
869
870 006304 013703 007012      LFSUB:  MOV      JMPADD,SCAN      ;GET POINTER TO FIRST CHAR ON SCREEN
871
872 006310 122300              LFLOOP: CMPB     (SCAN)+,CHAR      ;AND LOOK FOR A LINEFEED.
873 006312 001406              BEQ      LFOUND              ;GOT IT. SEARCH HAS ENDED.
874 006314 020327 007000      CMP      SCAN,#BLIMIT        ;ARE WE AT END OF BUFFER?
875 006320 103773              BLO      LFLOOP              ;NOPE. KEEP ON LOOKING.
876 006322 012703 001000      MOV      #BSTART,SCAN        ;IF AT TOP, RESET TO BOTTOM OF BUFFER
877 006326 000770              BR       LFLOOP              ;AND KEEP ON LOOKING.
878
879 006330 005203              LFOUND: INC      SCAN          ;WE'VE GOT THE LINE FEED. STOP SHOWING
880 006332 042703 000001      BIC      #1,SCAN            ;FIRST LINE BY CHANGING THE 'DISJMP'
881 006336 010337 007012      MOV      SCAN,JMPADD         ;INSTRUCTION TO FIRST CHAR BEYOND LF.
882 006342 004737 166350      JSR      PC,INSERT!160000      ;INSERT THE LF IN THE BUFFER.
883 006346 005000              CLR      CHAR                ;AND THEN INSERT ONE NULL CHARACTER BECAUSE
884                                         ;THE 'DISJMP' ADDRESS MUST BE EVEN, AND
885                                         ;THIS GUARANTEES WE WILL NOT LOSE A
886                                         ;A GOOD DATA CHARACTER. WE FALL THROUGH
887                                         ;TO INSERT THE NULL IN THE BUFFER.
888
889
890 006350 110021              INSERT: MOVB     CHAR,(POINTR)+      ;STICK IN THE CHARACTER NOW.
891 006352 032701 000001      BIT      #1,POINTR          ;IS NEXT POSITION EVEN OR ODD?
892 006356 001021              BNE      INSRTX              ;ODD. NO PROBLEMS. SPACE IS ALLOCATED.
893 006360 020127 007000      CMP      POINTR,#BLIMIT      ;EVEN. ARE WE AT THE END OF THE BUFFER?
894 006364 103410              BLO      INSRTL              ;NO. JUST MAKE ROOM FOR ANOTHER WORD.
895 006366 010103              MCV      POINTR,SCAN         ;AT THE END. MOVE THE STUFF TO THE
896 006370 012701 001000      MOV      #BSTART,POINTR      ;BEGINNING OF THE BUFFER.
897 006374 004737 166406      JSR      PC,INSRTL!160000      ;CALL THE ROUTINE TO SAVE SPACE.
898 006400 005023              CLR      (SCAN)+            ;AND CLEAR UP THE INSTRUCTIONS AT THE
899 006402 005013              CLR      (SCAN)              ;END OF THE BUFFER.
900 006404 000207              RTS      PC                  ;AND THEN RETURN.
901
902 006406 022121              INSRTL: CMP      (POINTR)+,(POINTR)+      ;BYPASS THE 'DISJMP' BY ADDING 4 TO POINTR.
903 006410 012711 166474      MOV      #HEADER!160000,(POINTR) ;NOW INSERT THE DISJMP INSTRUCTION TO OUR HEADER
904 006414 012741 160000      MOV      #DISJMP,-(POINTR)    ;AND IT'S ADDRESS (PUT THEM IN BACKWARDS).
905 006420 005041              CLR      -(POINTR)          ;MAKE AVAILABLE A NEW CHARACTER SPOT.
906
907 006422 000207              INSRTX: RTS      PC          ;FINALLY RETURN TO THE CALLER.
908
909
910
911
912
913 006424 012737 001000 172000 GTBUSE: MOV      #BSTART,GT40PC      ;ON A BUS ERROR, WE MERELY RESTART THE GT40 AT
914

```

```

915                                     ;THE RTI FOR THIS ROUTINE
916                                     ;IS THE FIRST WORD OF THE TABLE
917                                     ;BELOW-IT SAVES A WORD!
918
919
920
921
922
923
924
925
926
927
928
929                                     :
930                                     :
931                                     :
932                                     :
933                                     :
934 006432 000002      SETUP: .WORD 2          ;INITIALIZE 2 WORDS.--ALSO RTI FROM ABOVE
935 006434 000330      .WORD 330          ;STARTING AT LOCATION 330
936 006436 166424      .WORD GTBUSE!160000 ;FIRST WORD IS POINTER TO BUS ERROR ROUT
937 006440 000200      .WORD 200          ;SECOND WORD IS NEW STATUS WORD ON INTERRUPT.
938
939 006442 000007      .WORD 7          ;INITIALIZE THE END OF THE BUFFER TO
940 006444 006776      .WORD BLIMIT-2    ;A CLEAR SPACE TO INSERT THE CHARACTER.
941 006446 000000      .WORD 0          ;THIS IS THE 'RUNNING' START. THIS IS
942 006450 160000 166474 .WORD DISJMP,HEADER!160000 ;FOLLOWED BY A DISJMP TO OUR HEADER BLOC
943 006454 160000 001000 .WORD DISJMP,BSTART ;AND THEN A DISJMP TO THE START OF THE BUFFER
944 006460 160000 006700 .WORD DISJMP,BLIMIT-NUMLIN-NUMLIN ;AND A DISJMP TO THE FIRST CHAR ON SCREE
945
946 006464 000001      .WORD 1          ;FINALLY START THE GT40 GOING AT
947 006466 172000      .WORD GT40PC     ;THE POSITION INSTRUCTION IN THE
948 006470 166474      .WORD HEADER!160000 ;HEADER BLOCK.
949
950 006472 000000      .WORD 0          ;END OF INIT CODE
951                                     :
952                                     :
953                                     :
954 006474 103334      HEADER: .WORD 103334 ;ENABL CHAR MODE,BLINKING
955 006476 000177      .WORD 177        ;A BLINKING BOX-RUB OUT!
956 006500 116124      .WORD 116124    ;GO TO POINT MODE
957 006502 171340      .WORD 171340    ;LOAD STATUS REGISTER
958 006504 000000 001352 .WORD 0,1352 ;POINT TO UPPER LEFT
959 006510 103324      .WORD 103324    ;BACK TO CHAR MODE
960 006512 160000 007010 .WORD DISJMP,JMPADD-2 ;AND TO THE CHANGING JMP INST.
961
962 .SBTTL COMMUNICATIONS AND MISC. SUPPORT ROUTINES
963 .PAGE
  
```



```

964
965
966
967      :
968      :
969      :
970
971
972
973
974
975      :
976      :
977
978
979
980 006516 105737 175610      GETDL: TSTB   DL11IS      ;CHECK THE HOST INPUT STATUS.
981 006522 100011              BPL     GETDL1      ;HOST DID NOT SEND ANYTHING, YET.
982 006524 113700 175612      MOVB   DL11IB,CHAR  ;HOST SENT US A CHARACTER. PROCESS IT.
983 006530 012737 000007 175610  MOV    #7,DL11IS   ;REENABLE THE HOST TELECOMMUNICATIONS.
984 006536 042700 177600      BIC    #-200,CHAR  ;MAKE CHARACTER JUST SEVEN BITS.
985 006542 001765              BEQ    GETDL        ;IF NULL, IGNORE IT.
986 006544 000207              RTS     PC          ;ELSE RETURN NOW.
987
988 006546 105737 177560      GETDL1: TSTB   KBDIS      ;DID USER TYPE A CHARACTER?
989 006552 100361              BPL   GETDL        ;NO. GO BACK AND CHECK HOST MACHINE.
990 006554 113737 177562 175616  MOVB  KBDIB,DL110B  ;MOVE THE CHARACTER TO THE HOST.
991 006562 000755              BR    GETDL        ;AND CHECK AGAIN FOR INPUT.
992
993
994
995
996
997      :
998      :
999
1000
1001
1002 006564 004737 166516      GETCHR: JSR    PC,GETDL!160000 ;GET A CHARACTER FROM THE HOST NOW.
1003 006570 020027 000175      CMP    CHAR,#ALTMOD ;IS IT AN 'ALTMODE'
1004 006574 001025              BNE   GETEXT       ;NO. EXIT NOW.
1005
1006 006576 004737 166516              JSR    PC,GETDL!160000 ;YES. GET ANOTHER ONE NOW.
1007 006602 020027 000114      CMP    CHAR,#'L'    ;IS IT AN 'L'
1008 006606 001501              BEQ   LOADER       ;YES. START LOADING NOW.
1009 006610 020027 000122      CMP    CHAR,#'R'    ;IS IT AN 'R'
1010 006614 001015              BNE   GETEXT       ;NO. IGNORE THE ALTMODE AND JUST RETURN THE CHAR
1011
1012 006616 012737 173000 007010      MOV    #DISTOP,JMPADD-2 ;YES. RESET. STOP DISPLAY BY INSERTING A 'DISTOP
1013 006624 000137 166060      PRESTR: JMP   RESTRT!160000 ;INSTRUCTION IN THE BUFFER, AND RESTART.
1014
1015
1016
1017
1018
1019      :

```

COMMUNICATIONS HANDLING ROUTINES

THE DL-11 HANDLER

THE 'GET CHARACTER' ROUTINE

THE 'GET A SIX BIT CHARACTER' ROUTINE

```

1020 ; -----
1021
1022
1023
1024 006630 004737 166564 GETSIX: JSR PC,GETCHR!160000 ;GET A CHARACTER NOW.
1025 006634 020027 000040 CMP CHAR,#40 ;IS IT A LEGAL PRINTING CHARACTER?
1026 006640 002517 BLT L.BAD ;NOPE. ABORT
1027 006642 020027 000137 CMP CHAR,#137 ;IT'S BIG ENOUGH. IS IT TOO BIG?
1028 006646 003114 BGT L.BAD ;YEP. ABORT.
1029
1030 006650 000207 GETEXT: RTS PC ;RETURN TO THE CALLER.
1031
1032
1033 ; THIS OUTPUTS TWO CHARACTERS VIA A
1034 ; JSR SCAN,OUTLIT
1035 ; 'TWO CHARACTERS'
1036
1037 006652 112337 175616 OUTLIT: MOVB (SCAN)+,DL110B
1038 006656 112337 175616 MOVB (SCAN)+,DL110B ;DOUBLE BUFFERED
1039 006662 000203 RTS SCAN ;RETURN
1040
1041
1042
1043
1044
1045
1046
1047 ; THE 'GET AN EIGHT BIT CHARACTER' ROUTINE
1048 ; -----
1049
1050
1051
1052 ; THIS ROUTINE DIFFERS FROM THE PREVIOUS ROUTINES
1053 ; IN THAT IT WILL TAKE SIX BIT CHARACTERS AND ASSEMBLE
1054 ; THEM FOR THE LOADER TO USE. NOTE THAT FROM THIS POINT
1055 ; ON WE WILL SWITCH TO THE LOADER DEFINITIONS OF THE
1056 ; REGISTERS. THUS THE CHARACTER IS RETURNED IN
1057 ; REGISTER 'L.BYT' RATHER THAN CHAR (THOUGH THEY ARE
1058 ; PHYSICALLY THE SAME).
1059 ;
1060
1061
1062 006664 004737 166630 GET8: JSR PC,GETSIX!160000 ;GET A SIXBIT CHARACTER.
1063 006670 010046 MOV L.BYT,-(SP) ;SAVE IT ON THE STACK.
1064 006672 005723 TST (INDEX)+ ;UPDATE INDEX TO NEXT ITEM (ALL ARE *2)
1065 006674 000163 166676 JMP GET8TB-2!160000(INDEX) ;AND DISPATCH ACCORDING TO THE INDEX.
1066
1067 006700 000404 GET8TB: BR GET81 ;INDEX=2: ASSEMBLE FIRST CHAR
1068 006702 000416 BR GET82 ;INDEX=4: ASSEMBLE SECOND CHAR
1069 006704 000432 BR GET83 ;INDEX=6: ASSEMBLE THIRD AND LAST CHAR
1070 ;INDEX=8: RESET INDEX TO 0 [2] AND RETRY.
1071
1072
1073 006706 012703 000002 GET84: MOV #2,INDEX ;THE FOURTH INDEX IS THE SAME AS THE FIRST
1074 ;INDEX. JUST RESET IT AND FALL THROUGH.
1075

```


1076						
1077	006712	004737	166630	GET81:	JSR PC,GETSIX!160000	:GET ANOTHER CHARACTER NOW.
1078	006716	010004			MOV L.BYT,HOLD	:AND PRESERVE IT FOR NEXT TIME THROUGH.
1079	006720	006300			ASL L.BYT	:NOW THROW AWAY LEFT MOST BITS OF
1080	006722	006300			ASL L.BYT	:THE 8 BIT CHARACTER. NOW MERGE IN
1081	006724	106300			ASLB L.BYT	:THE LEFT TWO BITS OF THE
1082	006726	106116			ROLB (SP)	:NEW SIX BIT CHARACTER WITH THE SIX
1083	006730	106300			ASLB L.BYT	:BITS FROM THE CHARACTER ON THE
1084	006732	106116			ROLB (SP)	:STACK. 1ST CHARACTER IS NOW ASSEMBLED,
1085	006734	012600			MOV (SP)+,L.BYT	:SO WE'LL RETURN IT TO THE USER.
1086	006736	000207			RTS PC	:AND THEN WE SHALL RETURN TO HIM.
1087						
1088						
1089	006740	006300		GET82:	ASL L.BYT	:THE SECOND CHARACTER IS CREATED FROM
1090	006742	006300			ASL L.BYT	:THE 4 RIGHT BITS OF THE PREVIOUS CHARACTER
1091	006744	106300			ASLB L.BYT	:AND THE FOUR MIDDLE BITS OF THE PRESENT
1092	006746	106104			ROLB HOLD	:8 BIT CHARACTER.
1093	006750	106300			ASLB L.BYT	:WE WILL CREATE THE NEW 8 BIT
1094	006752	106104			ROLB HOLD	:IN THIS REGISTER, SINCE IT
1095	006754	106300			ASLB L.BYT	:MORE CONVIENT. WE WILL MOVE OVER THE
1096	006756	106104			ROLB HOLD	:ANSWER AT THE END.
1097	006760	106300			ASLB L.BYT	:ONE MORE TO GO
1098	006762	106104			ROLB HOLD	:DONE.
1099	006764	010400			MOV HOLD,L.BYT	:BRING OVER THE VALUE.
1100	006766	012604			MOV (SP)+,HOLD	:AND REMEMBER THE LAST CHARACTER WE RECEIVED.
1101	006770	000207			RTS PC	:AND RETURN TO THE CALLER.
1102						
1103						
1104	006772	006100		GET83:	ROL L.BYT	:FINAL CHARACTER IS EASY. JUST A
1105	006774	106100			ROLB L.BYT	:SIMPLE MERGER OF LEFT TWO BITS OF
1106	006776	006004			ROR HOLD	:PREVIOUS VALUE WITH RIGHT SIX BITS
1107	007000	106000			RORB L.BYT	:OF LAST (4TH) CHARACTER RECEIVED.
1108	007002	006004			ROR HOLD	
1109	007004	106000			RORB L.BYT	:AND WE ARE DONE.
1110	007006	005726			TST (SP)+	:FINALLY THROW AWAY STACK.
1111	007010	000207			RTS PC	:AND RETURN TO THE CALLER.
1112						
1113						
1114						
1115						
1116						
1117						
1118						
1119						
1120						
1121						
1122						
1123						
1124						
1125						

.SBTTL THE LOADER
 .PAGE

```

1126
1127
1128
1129          :          THE LOADER
1130          :          -----
1131
1132
1133
1134
1135 007012 012737 173000 007010 LOADER: MOV      #DISTOP,JMPADD-2      ;STOP THE GT40 BY INSERTING A 'DISTOP' IN THE LI
1136
1137 007020 005003          CLR      INDEX          ;RESET THE 8 BIT ASSEMBLER TO THE FIRST CHAR
1138
1139
1140 007022 005005          L.LD2:  CLR      L.CKSM          ;CLEAR THE CHECKSUM
1141 007024 004737 167114      JSR      PC,L.PTR!160000      ;GET A BYTE NOW.
1142 007030 105300          DECB    L.BYT          ;IS IT A ONE (HEADER)?
1143 007032 001373          BNE     L.LD2          ;NO. WAIT FOR THE ONE.
1144
1145 007034 004737 167114      JSR      PC,L.PTR!160000      ;YES. SKIP OVER THE NEXT CHARACTER NOW.
1146
1147 007040 004737 167126      JSR      PC,L.GWRD!160000      ;ASSEMBLE A WORD NOW.
1148 007044 010002          MOV     L.BYT,L.BC          ;MOVE OVER TO THE COUNTER.
1149 007046 162702 000004      SUB     #4,L.BC          ;REDUCE TO ACTUAL DATA COUNT.
1150 007052 022702 000002      CMP     #2,L.BC          ;ANY DATA AT ALL?
1151 007056 001433          BEQ     L.JMP          ;NO. MUST BE END
1152 007060 004737 167126      JSR      PC,L.GWRD!160000      ;YES. ASSEMBLE A DATA WORD NOW.
1153 007064 010001          MOV     L.BYT,L.ADR          ;AND THIS MUST BE THE FIRST ADDRESS.
1154
1155
1156 007066 004737 167114      L.LD3:  JSR      PC,L.PTR!160000      ;GET A BYTE OF DATA NOW.
1157 007072 002006          BGE     L.LD4          ;ALL DONE?
1158 007074 105705          TSTB   L.CKSM          ;YEP. COUNTER IS MINUS. CHECK CHECKSUM.
1159 007076 001751          BEQ     L.LD2          ;CHECKSUM GOOD. GET NEXT COMMAND.
1160
1161
1162 007100 004337 166652      L.BAD:  JSR      SCAN,OUTLIT!160000      ;BAD LOAD INFORM HOST
1163 007104 175 102          .BYTE  ALTMOD,'B          ;SEND ALTMODE B
1164 007106 000646          BR     PRESTR1          ;AND RESTART THE DISPLAY.
1165
1166
1167 007110 110021      L.LD4:  MOVB   L.BYT,(L.ADR)+          ;INSERT BYTE INTO MEMORY.
1168 007112 000765          BR     L.LD3          ;AND GET THE NEXT BYTE.
1169
1170
1171
1172 007114 004737 166664      L.PTR:  JSR      PC,GET8!160000      ;ASSEMBLE AN 8 BIT CHARACTER NOW.
1173 007120 060005          ADD     L.BYT,L.CKSM      ;UPDATE THE CHECKSUM NOW.
1174 007122 005302          DEC     L.BC          ;DECREMENT THE CHARACTER COUNTER.
1175 007124 000207          RTS     PC          ;AND RETURN TO THE CALLER NOW.
1176
1177
1178
1179 007126 004737 167114      L.GWRD: JSR      PC,L.PTR!160000      ;ASSEMBLE A WORD. FIRST GET A CHARACTER
1180 007132 010046          MOV     L.BYT,-(SP)        ;AND SAVE IT.
1181 007134 004737 167114      JSR      PC,L.PTR!160000      ;AND THEN GET ANOTHER ONE.

```


1182	007140	000300		SWAB	L.BYT				
1183	007142	052600		BIS	(SP)+,L.BYT				;AND THEN REASSEMBLE THE MESS.
1184	007144	000207		RTS	PC				;WITH THE FEARSOME POWER OF THE 11.
1185									;AND RETURN TO THE CALLER.
1186									
1187									
1188									
1189	007146	004737	167126	L.JMP:	JSR	PC,L.GWRD!160000			;ALL DONE WITH THE LOAD. ASSEMBLE
1190	007152	010046			MOV	L.BYT,-(SP)			;THE STARTING ADDRESS NOW.
1191	007154	004737	167114		JSR	PC,L.PTR!160000			;AND DON'T FORGET TO CHECKSUM IT.
1192	007160	105705			TSTB	L.CKSM			
1193	007162	001346			BNE	L.BAD			;A BAD CHECKSUM. ALL IS EVIL.
1194									
1195	007164	004337	166652		JSR	SCAN,OUTLIT!160000			;GOOD CHKSUM,INFORM HOST
1196	007170	175	107		.BYTE	ALTMOD,'G			;WITH ALTMOD G
1197									
1198	007172	032716	000001		BIT	#1,(SP)			;DO WE WANT TO START EXECUTION?
1199	007176	001401			BEQ	L.JMP1			;YES. AWAY WE GO.
1200									
1201	007200	000000		L.HALT:	HALT				;IF NOT, HALT.
1202									
1203	007202	000136		L.JMP1:	JMP	@(SP)+			;IF GO, THEN GO ALREADY. WHEEEE!
1204									
1205									
1206									
1207									
1208									
1209									
1210									
1211									
1212									
1213									
1214									
1215									
1216									
1217									
1218									
1219									
1220									

.SBTTL THE SELF TEST

.PAGE

:THIS IS GT40 QUICK TEST
:GIVES QUICK VISUAL TEST
:OF CONDITION OF MACHINE
:WITHOUT READING IN DIAG.

1221		
1222		
1223		
1224		
1225	100000	CHAR=100000
1226	104000	SHORTV=104000
1227	110000	LONGV=110000
1228	114000	POINT=114000
1229	120000	GRAPHX=120000
1230	124000	GRAPHY=124000
1231	130000	RELATV=130000
1232		
1233	002000	INT0=2000
1234	002200	INT1=2200
1235	002400	INT2=2400
1236	002600	INT3=2600
1237	003000	INT4=3000
1238	003200	INT5=3200
1239	003400	INT6=3400
1240	003600	INT7=3600
1241		
1242	000100	LPOFF=100
1243	000140	LPON=140
1244	000020	BLKOFF=20
1245	000030	BLKON=30
1246		
1247	000004	LINE0=4
1248	000005	LINE1=5
1249	000006	LINE2=6
1250	000007	LINE3=7
1251		
1252	160000	DJMP=160000
1253	164000	DNOP=164000
1254	170000	STATSA=170000
1255	173400	DSTOP=173400
1256		
1257	000300	LPLITE=300
1258	000200	LPDARK=200
1259	000040	ITAL0=40
1260	000060	ITAL1=60
1261	000004	SYNON=4
1262		
1263		
1264	174000	STATSB=174000
1265		
1266	000100	INCR=100
1267	040000	INTX=40000
1268	001777	MAXX=1777
1269	001377	MAXY=1377
1270	020000	MINUSX=20000
1271	020000	MINUSY=MINUSX
1272	017600	MAXSX=17600
1273	000077	MAXSY=77
1274	000100	MINSUY=100
1275		
1276		

:BRIGHTEST

:STOP INTERRUPT

:ITALICS OFF
:ON
:SYNC ON

:LOAD GRAPH INCR
:INTENSIFY BIT
:BIGGEST X VECTOR
:BIGGEST Y VECTOR
:THE MINUS BIT

:BIGGEST X IN SHORTVEC
:Y IN
:MINUS BIT FOR Y IN SHORTVEC


```

1277 007204 012737 167214 172000      MOV      #FILE0:160000,GT40PC      ;START THE GT40
1278 007212 000001                      WAIT      ;AND WAIT
1279                                     FILE0: POINT!BLKOFF      ;POINT--INVISIBLE
1280 007214 114020                      0
1281 007216 000000                      MAXY
1282 007220 001377
1283
1284 007222 112004                      LONGV!INT0!LINE0      ;DRAW TOP LINE
1285 007224 041777                      INTX!MAXX
1286 007226 000000                      0
1287
1288 007230 112405                      LONGV!INT2!LINE1
1289 007232 040000                      INTX      ;DRAW LINE TO RIGHT
1290 007234 021377                      MINUSX!MAXY
1291
1292 007236 113006                      LONGV!INT4!LINE2
1293 007240 061777                      INTX!MINUSX!MAXX      ;DRAW BOTTOM LINE
1294 007242 000000                      0
1295
1296 007244 113407                      LONGV!INT6!LINE3
1297 007246 040000                      INTX
1298 007250 001377                      MAXY      ;DRAW LINE TO LEFT
1299
1300 007252 114000                      POINT
1301 007254 000400                      400
1302 007256 000500                      500
1303 007260 106200                      SHORTV!INT1
1304 007262 057677                      57677      ;+X+Y
1305 007264 106600                      SHORTV!INT3
1306 007266 077677                      77677      ;+X-Y
1307 007270 107200                      SHORTV!INT5
1308 007272 077777                      77777      ;-X-Y
1309 007274 107600                      SHORTV!INT7
1310 007276 057777                      57777      ;-X+Y
1311
1312 007300 114000                      POINT
1313 007302 001400                      1400
1314 007304 000500                      500
1315 007306 133030                      RELATV!INT4!BLKCN
1316 007310 057677                      57677      ;+X+Y
1317 007312 077677                      77677      ;+X-Y
1318 007314 077777                      77777      ;-X-Y
1319 007316 057777                      57777      ;-X+Y
1320
1321 007320 114000                      POINT
1322 007322 000400                      400
1323 007324 000100                      100
1324 007326 174120                      STATSB!INCR+20      ;TRY GRAPH MODES
1325 007330 114000                      POINT
1326 007332 001000                      1000
1327 007334 000200                      200
1328
1329 007336 120000                      GRAPHX
1330 007340 001010                      1010
1331 007342 001020                      1020
1332 007344 001030                      1030
  
```

1333	007346	001040	1040
1334	007350	001050	1050
1335			
1336	007352	114000	POINT
1337	007354	001000	1000
1338	007356	001200	1200
1339			
1340	007360	124000	GRAPHY
1341	007362	001020	1020
1342	007364	001030	1030
1343	007366	001040	1040
1344	007370	001050	1050
1345	007372	001060	1060
1346			
1347	007374	160000	DJMP
1348	007376	167214	FILE0:160000
1349			
1350			.SBTTL PAPER TAPE BOOT


```

1352
1353
1354
1355          177550
1356          177560
1357
1358
1359 007400 012701 160000
1360 007404 012702 000004
1361 007410 012703 167500
1362 007414 010712
1363 007416 012706 000024
1364 007422 014304
1365 007424 005714
1366 007426 100775
1367 007430 010712
1368 007432 012706 000024
1369 007436 010441
1370
1371 007440 040601
1372 007442 010111
1373 007444 011102
1374 007446 005214
1375 007450 105714
1376 007452 100376
1377 007454 116412 000002
1378 007460 005211
1379 007462 120227 000375
1380 007466 001366
1381 007470 105222
1382 007472 000142
1383
1384
1385
1386 007474 177560
1387 007476 177550
1388
1389

:
: PAPER TAPE BOOT
:
HSR=177550          :HIGH SPEED READER ADDRESS
LSR=177560          :LOW SPEED READER ADDRESS
:
:      .=ORIGIN+1400
:
PTBOOT: MOV      #160000,R1      :SET MEMORY CHECK LIMITS
        MOV      #4,R2          :TRAP ADDRESS IS LOC. 4
        MOV      #DEV+4!160000,R3 :POINTER TO DEVICE ADDRESSES
        MOV      PC,@R2         :PRESET TRAP ADDRESS IN LOC. 4
        MOV      #24,SP         :STACK SET UP AT SPECIAL ADDRESS
DEV1:   MOV      -(R3),R4       :GET DEVICE ADDRESS
        TST      @R4            :CHECK AVAILABILITY OF DEVICE
        BMI      DEV1          :CHECK DEVICE FOR ERRORS
        MOV      PC,@R2         :RESET TRAP ADDRESS AT LOC. 4
        MOV      #24,SP         :SPECIAL ADDRESS USED AS MASK LATER
        MOV      R4,-(R1)       :DO MEM CHK:READER STATUS ADDRESS
        :IS MOVED
        BIC      SP,R1          :SET R1=X7752,MASK IN SP=24
        MOV      R1,@R1         :STORE OWN ADDRESS IN POINTER
LOOP:   MOV      @R1,R2         :GET BYTE POINTER
        INC      @R4            :ENABLE READER
        TSTB     @R4            :TEST DONE BIT
        BPL      -2            :WAIT UNTIL READY
        MOVB     2(R4),@R2      :THEN PICK IT UP AND STORE IT
        INC      @R1            :BUMP POINTER
        CMPB     R2,#375        :STORED JUMP OFFSET?
        BNE      LOOP          :NOT YET
        INCB     (R2)+          :YES,ALL DONE
        JMP      -(R2)         :GO EXECUTE AS BRANCH
:
: DEVICE ADDRESSES FOLLOW - DO NOT CHANGE THE ORDER
:
DEV:    LSR                      :LOW SPEED READER
        HSR                      :HIGH SPEED READER
:
.SBTTL  CASSETTE BOOT

```

```

1392
1393
1394
1395          177500
1396
1397 007500 012700 177500
1398 007504 005010
1399 007506 010701
1400 007510 062701 000052
1401 007514 012702 000375
1402 007520 112103
1403
1404 007522 112110
1405 007524 100413
1406 007526 130310
1407 007530 001776
1408 007532 105202
1409 007534 100772
1410 007536 116012 000002
1411 007542 120337 000000
1412 007546 001767
1413 007550 000000
1414 007552 000755
1415
1416 007554 005710
1417 007556 100774
1418 007560 005007
1419
1420 007562 017640
1421
1422 007564 002415
1423
1424 007566 112024
1425
1426 007570 000000 000000
1427 007574 167500
1428 007576 000340
1429
1430
1431

: CASSETTE BOOT
TACS=177500 ;TA-11 CONTROL AND STATUS REGISTER
: =ORIGIN+1500
TABOOT: MOV #TACS,R0
        CLR (R0) ;SELECT UNIT #0
RES:    MOV PC,R1 ;USE FOR PIC
        ADD #TABLE--,R1 ;R1 HOLDS ADDR. OF COMMAND TABLE
        MOV #375,R2 ;MEMORY PTR. AND DATA FLAG
        MOVB (R1)+,R3 ;TEST BITS

:
LOOP1:  MOVB (R1)+,(R0) ;COMMAND FROM TABLE TO TACS
        BMI DONE ;WHEN COMMAND CODE NEG., QUIT
LOOP2:  BITB R3,(R0) ;TEST READY AND T-REQ BITS IN TACS
        BEQ LOOP2 ;LOOP 'TIL SOMETHING COMES UP
        INCB R2 ;ADVANCE MEMORY POINTER
        BMI LOOP1 ;IF MINUS, TRY NEXT COMMAND
        MOVB 2(R0),(R2) ;READ DATA INTO MEMORY
        CMPB R3,@#0 ;FIRST BYTE READ SHOULD BE '240'
        BEQ LOOP2 ;IF O.K., GO READ ANOTHER BYTE
STOP:   HALT ;HALT ON ERROR
        BR RES ;RESTART ON CONTINUE

:
DONE:   TST (R0) ;CHECK FOR ERROR
        BMI STOP ;HALT ON ERROR
        CLR PC ;= 'JMP @#0'

:
TABLE:  .WORD 17640 ;.BYTE 240: READY+T-REQ.
        ;.BYTE 37: ILBS+READY+GO
        .WORD 2415 ;.BYTE 15: SFB+GO
        ;.BYTE 5: READ+GO
        .WORD 112024 ;.BYTE 24: READ+ILBS
        ;.BYTE 224: READ+ILBS+E.O.TABLE
        .WORD 0,0 ;THESE ARE FILLER WORDS
        .WORD TABOOT!160000 ;POWER UP VECTOR AND PRIORITY
        .WORD 340 ;

.SBTTL MR11-DB BOOT

```



```
1433 ;MR11-DB BULK STORAGE PROGRAM LOADER LISTING
1434
1435 ;      .=ORIGIN+1600 ;KEEP TRACK OF ORIGIN
1436
1437 007600 010702 RF11: MOV PC,R2 ;FIXED HEAD DISK (256 KW)
1438 007602 000451 BR OTHER
1439 007604 177462 177462
1440 007606 000005 5
1441
1442 007610 010702 RK11: MOV PC,R2 ;MOVING HEAD DISK (CARTRIDGE)
1443 007612 000445 BR OTHER
1444 007614 177406 177406
1445 007616 000005 5
1446
1447
1448 007620 010702 TC11: MOV PC,R2
1449 007622 000417 BR TAPES ;ADDRESS OF WORD COUNT
1450 007624 177344 177344 ;LAST COMMAND
1451 007626 000005 5 ;FIRST COMMAND
1452 007630 004003 4003 ;DONE MASK
1453 007632 100000 100000 ;ERROR MASK
1454 007634 024000 24000
1455
1456
1457 007636 010702 TM11: MOV PC,R2
1458 007640 000410 BR TAPES ;ADDRESS OF BYTE COUNT
1459 007642 172524 172524 ;LAST COMMAND
1460 007644 060003 60003 ;FIRST COMMAND
1461 007646 060011 60011 ;DONE MASK
1462 007650 000200 200 ;ERROR MASK
1463 007652 100000 100000
1464
1465
1466 007654 010702 RP11: MOV PC,R2 ;MOVING HEAD DISK (PACK)
1467 007656 000423 BR OTHER
1468 007660 176716 176716
1469
1470
1471 007662 000005 TAPES: RESET
1472 007664 010200 MOV R2,R0 ;GET THE ADDRESS OF THE BRANCH
1473 007666 005720 TST (0)+ ;R0 TO POINT AT LAST COMMAND
1474 007670 012001 MOV (0)+,R1 ;GET THE WORD COUNT ADDRESS
1475 007672 005311 DEC (1) ;SET UP FOR ADVANCE 1 RECORD
1476 007674 005720 TST (0)+ ;MOVE R0 TO FIRST COMMAND
1477 007676 012041 MOV (0)+,-(1) ;COMMAND WORD TO COMMAND REG.
1478 007700 031011 BIT (0),(1) ;LOOK FOR DONE INDICATORS
1479 007702 001776 BEQ ,-2 ;NONE SET, TRY AGAIN
1480 007704 005720 TST (0)+ ;DONE FIRST COMMAND, CHECK FOR ERROR
1481 007706 031041 BIT (0),-(1) ;LOOK FOR SET ERROR BITS
1482 007710 001406 BEQ OTHER ;NO ERRORS - TRY THE READ
1483 007712 000112 AGAIN: JMP (2) ;RERUN FOR ERRORS
1484
1485
1486 007714 167600 RFVEC: RF11!160000 ;RF11 POWER UP VECTOR
1487 007716 000340 340
1488
```

1489 007720 010702
1490 007722 000401
1491 007724 177450
1492
1493
1494 007726 000005
1495 007730 010200
1496 007732 005720
1497 007734 012001
1498 007736 012711 177000
1499 007742 011041
1500 007744 032711 100200
1501 007750 001775
1502 007752 100757
1503 007754 005007
1504
1505 007756 000000
1506 007760 167610
1507 007762 000340
1508 007764 167720
1509 007766 000340
1510 007770 167654
1511 007772 000340
1512 007774 167620
1513 007776 000340
1514
1515
1516

RC11: MOV PC,R2 ;FIXED HEAD DISK (64KW)
BR OTHER ;ADRS OF WORD COUNT (COMMAND+2)
177450 ;COMMAND WORD (5) IS THE RESET

OTHER: RESET
MOV R2,R0 ;R0 TO POINT AT WORD COUNT ADRS
TST (0)+ ;POINT TO ADDRESS
MOV (0)+,R1 ;WORD COUNT ADDRESS TO R1
MOV #-1000,(1) ;LOAD WORD COUNT
MOV (0),-(1) ;COMMAND TO COMMAND REGISTER
BIT #100200,(1) ;CHECK FOR ERROR OR DONE
BEQ .-4 ;IF NEITHER, KEEP LOOKING
BMI AGAIN ;ERROR, TRY AGAIN
CLR PC

0 ;FILLER
RKVEC: RK11!160000 ;RK POWER UP VECTOR
340
RCVEC: RC11!160000 ;RC POWER UP VECTOR
340
RPVEC: RP11!160000 ;RP POWER UP VECTOR
340
TCVEC: TC11!160000 ;TC11 POWER UP VECTOR
340

.SBTTL ROM VERSION 1 VALUES
.PAGE

1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572

000000
000001
000002
000003
000004
000005
000006
000007

000006
000007

000000
000001
000002
000003
000004
000005

000003
000000
000005
000001

```
.DSABL AMA

;DATA PATTERN STORED IN THE GT40 BOOTSTRAP VERSION 1

; ***** THIS IS A IMAGE LISTING OF THE GT40 <VT40> BOOTSTRAP *****
;
; THE DATA IS A MIRROR IMAGE OF THAT IN THE BOOTSTRAP ROMS
; ONLY THE ADDRESS FIELD IS CHANGED

;BOOTVT.S09 5/2/72 <SPECIAL>

; VT-40 BOOTSTRAP LOADER, VERSION S09, RELEASE R01, 5/2/72

; COPYRIGHT 1972, DIGITAL EQUIPMENT CORPORATION.
; 146 MAIN STREET
; MAYNARD, MASSACHUSSETTS 01754

; WRITTEN BY JACK BURNES, SENIOR SYSTEMS ARCHITECT!

; THIS ROUTINE IS INTENDED TO BE LOADED IN THE ROM PORTION OF THE VT-40.

; REGISTER DEFINITIONS:
R0=%0
R1=%1
R2=%2
R3=%3
R4=%4
R5=%5
R6=%6
R7=%7

SP=R6
PC=R7

RET1=R0 ;RETURN OF VALUE REGISTER.
INP1=R1 ;ARGUMENT FOR CALLED FUNCTION
INP2=R2 ;SECOND ARGUMENT.
WORK1=R3 ;FIRST WORK REGISTER.
WORK2=R4 ;SECOND WORKING REGISTER.
SCR1=R5 ;SCRATCH REGISTER.

LCKSM=WORK1 ;OVERLAPPING DEFINITIONS FOR LOADER PORTION.
LBYT=RET1
LBC=SCR1
LADR=INP1
```

1573						
1574	036000			COREND=36000		;FIRST LOCATION OF NON-CORE.
1575	166000			ROMORG=166000		;WHERE THE ROM PROGRAM SHOULD GO.
1576						
1577	000000			STARTX=0		;WHERE TO START DISPLAYING THE X POSITIONS.
1578	001360			STARTY=1360		;WHERE TO START DISPLAYING THE Y.
1579						
1580						
1581	022000			VT40PC=172000-150000		;VT40 PROGRAM COUNTER.
1582	027560			KBDIS=27560		;TTY INPUT STATUS.
1583	025614			P100S=25614		;PDP-10 OUTPUT STATUS.
1584	025610			P10IS=25610		;PDP-10 INPUT STATUS.
1585						
1586	027562			KBDIB=KBDIS+2		;TTY INPUT BUFFER.
1587	025612			P10IB=P10IS+2		;PDP-10 INPUT CHARACTER.
1588	025616			P10OB=P100S+2		;PDP-10 OUTPUT BUFFER.
1589						
1590						
1591	045776			P100C=COREND-2+10000		;CHARACTER TO BE SENT TO THE PDP-10
1592	045772			P10IC=P100C-4		;INPUT CHARACTER FROM 10 PLUS ONE SAVE CHARACTER
1593	015770			STKSRT=P10IC-2-30000		;FIRST LOCATION OF STACK.
1594						
1595						
1596	160000			JMPDIS=160000		;THE VT-40 DISPLAY JUMP INSTRUCTION.
1597						
1598						
1599	000024			PWRFAL=24		;POWER FAIL RESTART LOCATION.
1600						
1601						
1602						
1603						
1604						
1605						
1606						
1607						
1608						
1609						
1610						
1611	016000			.=16000		
1612				.=ROMORG		;SET THE ORIGIN NOW!!!!
1613						
1614						
1615						
1616						
1617						
1618						
1619						
1620	016000	012705	000026	STARTA: MOV #PWRFAL+2,SCR1		;PICK UP POINTER TO P.F. STATUS.
1621	016004	005015		CLR @SCR1		;CLEAR IT OUT TO BE SURE.
1622	016006	010745		MOV PC,-(SCR1)		;SET UP THE RESTART LOCATION.
1623						
1624	016010	000005		RESET		;RESET THE BUS.
1625						
1626	016012	012767	000007 007570	MOV #7,P10IS		;INITIALIZE PDP-10 INPUT
1627	016020	012767	000001 011532	MOV #1,KBDIS		;INITIALIZE TTY INPUT.
1628	016026	012767	000201 007560	MOV #201,P100S		;INITIALIZE PDP-10 OUTPUT.


```

1629
1630
1631
1632 016034 012706 015770      RSTRT:  MOV    #STKSRT,SP      ;SET UP THE STACK NOW!
1633 016040 005001              CLR    LADR                  ;CLEAR ADDRESS POINTER.
1634 016042 012702 160000      MOV    #JMPDIS,INP2         ;PLACE A DISPLAY JUMP INSTRUCTION IN A REGISTER.
1635 016046 010221              MOV    INP2,(LADR)+         ;MOVE IT TO LOCATION 0.
1636 016050 012711 166756      MOV    #DISPRG+150000,(LADR) ;MOVE ADDRESS POINTER INTO 2.
1637 016054 012701 000030      MOV    #PWRFAL+4,LADR      ;SET UP WHERE WE WILL STORE CHARACTERS.
1638 016060 005000              CLR    RET1                 ;PREPARE TO INSERT A ZERO CHARACTER.
1639 016062 004767 000022      JSR    PC,DOCHAR           ;INSERT IT NOW.
1640 016066 005067 003706      CLR    VT40PC              ;CLEAR THE DISPLAY PROGRAM COUNTER AND START.
1641
1642 016072 004767 000210      MAJOR:  JSR    PC,GTCHR      ;GT A CHARACTER NOW.
1643 016076 000240              NOP
1644 016100 000240              NOP
1645 016102 000240              NOP
1646 016104 012746 166072      MOV    #MAJOR+150000,-(SP) ;INSERT IN DISPLAY BUFFER NOW.
1647
1648 016110 010105      DOCHAR: MOV    LADR,SCR1      ;GT CURRENT BUUFER POSITION NOW.
1649 016112 022525      CMP    (SCR1)+,(SCR1)+     ;BYPASS CURRENT DISPLAY JUMP.
1650 016114 005025      CLR    (SCR1)+            ;CLEAR FUTURE ADDRESS FOR JUMP.
1651 016116 010225      MOV    INP2,(SCR1)+       ;STICK IN TEMPORARY JUMP WHILE WE REPLACE CURREN
1652 016120 005015      CLR    (SCR1)             ;A DISPLAY JUMP TO ZERO.
1653 016122 005011      CLR    (LADR)            ;NOW REPLACE CURRENT DISPLAY JUMP BY THE CHARACT
1654 016124 050021      BIS    RET1,(LADR)+       ;IT'S DONE THIS WAY TO WASTE 2 CYCLES.
1655 016126 010211      MOV    INP2,(LADR)       ;TO AVOID TIMING PROBLEMS WITH THE VT40.
1656 016130 000207      RTS    PC                 ;AND FINALLY RETURN.
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673 016132 004767 000124      GT8:    JSR    PC,GTSIX     ;GT SIX BITS NOW.
1674 016136 010046      MOV    RET1,-(SP)        ;SAVE THE CHARACTER NOW.
1675 016140 000401      BR     GTP84             ;BYPASS THE 8'ER
1676 016142 005002      GT84:  CLR    INP2        ;RESET THE MAGIC REGISTER NOW.
1677 016144 005722      GTP84: TST    (INP2)+      ;INCREMENT WHERE TO GO.
1678 016146 066207 166250      ADD    GT8TB+150000(INP2),PC ;UPDATE PC NOW.
1679
1680          016152      GT8P=.
1681
1682 016152 004767 000104      GT81:  JSR    PC,GTSIX     ;GT A CHARACTER NOW.
1683 016156 010004      MOV    RET1,WORK2       ;SAVE FOR A SECOND.
1684 016160 006300      ASL    RET1

```

1685	016162	006300		ASL	RET1	;SHIFT TO LEFT OF BYTE
1686	016164	106300		ASLB	RET1	
1687	016166	106116		ROLB	@SP	;PACK THEM IN.
1688	016170	106300		ASLB	RET1	
1689	016172	106116		ROLB	@SP	;A GOOD 8 BIT THING.
1690	016174	012600		MOV	(SP)+,RET1	;POP AND RETURN NOW.
1691	016176	000207		RTS	PC	
1692						
1693	016200	006300	GT82:	ASL	RET1	;WORST CASE. SHIFT 4
1694	016202	006300		ASL	RET1	
1695	016204	106300		ASLB	RET1	
1696	016206	106104		ROLB	WORK2	
1697	016210	106300		ASLB	RET1	
1698	016212	106104		ROLB	WORK2	
1699	016214	106300		ASLB	RET1	
1700	016216	106104		ROLB	WORK2	
1701	016220	106300		ASLB	RET1	
1702	016222	106104		ROLB	WORK2	
1703	016224	010400		MOV	WORK2,RET1	
1704	016226	012604		MOV	(SP)+,WORK2	
1705	016230	000207		RTS	PC	
1706						
1707	016232	006100	GT83:	ROL	RET1	
1708	016234	006100		ROL	RET1	
1709	016236	006004		ROR	WORK2	
1710	016240	106000		RORB	RET1	
1711	016242	006004		ROR	WORK2	
1712	016244	106000		RORB	RET1	;FINAL CHARACTER ASSEMBLED.
1713	016246	005726		TST	(SP)+	;FUDGE STACK.
1714	016250	000207		RTS	PC	;AND RETURN NOW.
1715						
1716		016250	GT8TB =		.-2	;PUSH ZERO CONDITION BACK INTO NEVER-NEVER LAND.
1717						
1718	016252	000000		.WORD	GT81-GT8P	
1719	016254	000026		.WORD	GT82-GT8P	
1720	016256	000060		.WORD	GT83-GT8P	
1721	016260	177770		.WORD	GT84-GT8P	
1722						
1723						
1724	016262	004767	000020	JSR	PC,GTCHR	
1725	016266	020027	000040	CMP	RET1,#40	
1726	016272	002546		BLT	LBAD	
1727	016274	020027	000137	CMP	RET1,#137	
1728	016300	003143		BGT	LBAD	
1729	016302	000207		RTS	PC	
1730						
1731						
1732						
1733	016304	005726	GTCHP:	TST	(SP)+	;UPDATE THE STACK.
1734						
1735	016306	012700	015772	GTCHR:	MOV #P10IC-30000,RET1	;SET UP POINTER TO THE INPUT CHARACTER.
1736	016312	004767	000064	GTCHL:	JSR PC,CHECK	
1737	016316	005710		TST	@RET1	;ANY CHARACTERS THERE?
1738	016320	001774		BEQ	GTCHL	
1739	016322	011046		MOV	@RET1,-(SP)	;PUSH THE CHAR ON THE STACK.
1740	016324	005020		CLR	(RET1)+	;CLEAR THE CHAR GOT FLAG NOW.


```

1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809 016520 005002
1810 016522 012712 172000
1811 016526 012706 015770
1812
1813 016532 005003
1814 016534 004767 000070
1815 016540 105300
1816 016542 001373
1817 016544 004767 000060
1818
1819 016550 004767 000072
1820 016554 010005
1821 016556 162705 000004
1822 016562 022705 000002
1823 016566 001437
1824 016570 004767 000052
1825 016574 010001
1826
1827 016576 004767 000026
1828 016602 002010
1829 016604 105703
1830 016606 001751
1831
1832 016610 012700
1833
1834 016612 102 175
1835 016614 004767 000110
1836 016620 000167 177210
1837
1838 016624 110021
1839 016626 000763
1840
1841 016630 004767 177276
1842 016634 060003
1843 016636 042700 177400
1844 016642 005305
1845 016644 000207
1846
1847 016646 004767 177756
1848 016652 010046
1849 016654 004767 177750
1850 016660 000300
1851 016662 052600
1852 016664 000207

```

```

; THE LOADER
LOAD: CLR INP2 ;RESET TO FIRST 8 BIT CHARACTER.
      MOV #172000,(INP2) ;AND ALSO CLEVERLY STOP THE VT40.
      MOV #STKSRT,SP ;RESET STACK POINTER NOW.

LLD2: CLR LCKSM ;CLEAR THE CHECKSUM
      JSR PC,LPTR ;GT A BYTE NOW.
      DECB LBYT ;IS IT ONE?
      BNE LLD2 ;NOPE. WAIT AWHILE
      JSR PC,LPTR ;YEP. GT NEXT CHARACTER.

      JSR PC,LGWRD ;GT A WORD.
      MOV LBYT,LBC ;GT THE COUNTER NOW.
      SUB #4,LBC ;CHOP OFF EXTRA STUFF.
      CMP #2,LBC ;NULL?
      BEQ LJMP ;YEP. MUST BE END.
      JSR PC,LGWRD ;NOPE. GT THE ADDRESS.
      MOV LBYT,LADR ;AND REMEMBER FOR OLD TIMES SAKE.

LLD3: JSR PC,LPTR ;GT A BYTE (DATA)
      BGE LLD4 ;ALL DONE WITH THE COUNTER?
      TSTB LCKSM ;YEP. GOOD CHECK SUM?
      BEQ LLD2 ;NOPE. LOAD ERROR.

LBAD: MOV (PC)+,RET1 ;SEND OUT SOME CHARACTERS NOW.
; .BYTE 175,102 ;'CTRL BAD'
; .BYTE 102,175 ;'BAD CTRL'
      JSR PC,SENDIT
      JMP RSTRT

LLD4: MOVB LBYT,(LADR)+ ;PLACE THE BYTE IN CORE.
      BR LLD3 ;GT ANOTHER ONE.

LPTR: JSR PC,GT8 ;GT 8 BITS NOW.
      ADD LBYT,LCKSM ;UPDATE CHECKSUM
      BIC #177400,LBYT ;CLEAN UP THE BYTE NOW.
      DEC LBC ;UPDATE THE COUNTER.
      RTS PC ;RETURN NOW.

LGWRD: JSR PC,LPTR ;GT A CHARACTER.
      MOV LBYT,-(SP) ;SAVE FOR A SECOND.
      JSR PC,LPTR ;GT ANOTHER CHARACTER.
      SWAB LBYT ;NOW ASSEMBLE THE WORD.
      BIS (SP)+,LBYT ;AND RETURN WITH A 16 BITER.
      RTS PC

```


1853
 1854 016666 004767 177754
 1855 016672 010046
 1856 016674 004767 177730
 1857 016700 105703
 1858 016702 001342
 1859 016704 032716 000001
 1860 016710 001406
 1861 016712 012700
 1862
 1863 016714 107 175
 1864 016716 004767 000006
 1865 016722 000000
 1866 016724 000776
 1867
 1868 016726 000136
 1869
 1870
 1871
 1872
 1873
 1874
 1875
 1876
 1877
 1878
 1879
 1880
 1881
 1882
 1883
 1884
 1885
 1886 016730 004767 177446
 1887 016734 005767 027036
 1888 016740 001373
 1889 016742 010067 006650
 1890 016746 105000
 1891 016750 000300
 1892 016752 001366
 1893 016754 000207
 1894
 1895
 1896
 1897
 1898
 1899
 1900
 1901
 1902
 1903
 1904
 1905
 1906
 1907
 1908

```

LJMP: JSR PC, LGWRD ;GT A WORD
      MOV LBYT, -(SP) ;SAVE ON THE STACK.
      JSR PC, LPTR ;GT A CHARACTER.
      TSTB LCKSM ;IS IT ZERO?
      BNE LBAD ;YEP. WHAT CRAP.
      BIT #1, (SP) ;IS IT ODD?
      BEQ LJMP1 ;YEP. START PROGRAM GOING NOW.
      MOV (PC)+, RET1 ;TELL PDP-10 WE'VE LOADED OK.
      ; .BYTE 175, 107 ;'CTRL GOOD'
      ; .BYTE 107, 175 ;'GOOD CTRL'
      JSR PC, SENDIT
      HALT
      BR .-2

LJMP1: JMP @ (SP)+ ;AND AWAY WE GO.

SENDIT: JSR PC, CHECK ;POLL THE OUTPUT DEVICE NOW.
        TST P100C ;OUTPUT CLEAR?
        BNE SENDIT ;NOPE. LOOP AWHILE LONGER.
        MOV RET1, P100B ;SEND OUT THE CHARACTER.
        CLRB RET1 ;CLEAR THE BYTE.
        SWAB RET1 ;AND SWAP THEM NOW.
        BNE SENDIT ;IF NOT EQUAL, REPEAT.
        RTS PC
  
```

1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941

016756 170256
016760 115124
016762 000000
016764 001360
016766 100000
016770 160000
016772 000030
016774 000000
016776 000000

DISPRG: .WORD 170256
 .WORD 115124
 .WORD STARTX
 .WORD STARTY
 .WORD 100000
 .WORD JMPDIS
 .WORD PWRFAL+4
 .WORD 0
 .WORD 0

: THIS IS THE INITIALIZING VT40 PROGRAM WHICH WILL
: JUMP TO THE PROGRAM AFTER THE POWER FAIL LOCATIONS
: WHICH WILL JUMP TO ZERO WHICH WILL JUMP BACK TO HERE.

:LOAD STATUS REGISTER FOR NORMAL OPERATION.
:SET POINT MODE, 'NORMAL'.
:X COORDINATE
:Y COORDINATE
:SET CHARACTER MODE.
:THEN JUMP TO THE POWERFAIL LOCATION.
:TO DISPLAY USERS CHARACTERS.

000001

.END

AGAIN	007712	1483#	1502						
AL TMOD=	000175	682#	1003	1163	1196				
BELL	006250	823	855#						
BLIMIT=	007000	675#	678	768	789	874	893	940	944
BLKOFF=	000020	1244#	1280						
BLKON =	000030	1245#	1315						
BSTART=	001000	674#	876	896	913	943			
CHAR =	100000	625#							
CHARA	001034	208#	408*	409	415	431			
CHECK	016402	1736	1766#	1778	1886				
CHECK1	016430	1767	1769	1773#					
CHECK2	016450	1778#	1780						
CHECK3	016466	1774	1784#						
CHECK4	016516	1785	1790#						
COREND=	036000	1574#	1591						
CORSTR=	000004	677#	737						
CR	006206	832#							
CRLF =	005015	681#	771						
DEV	007474	1361	1386#						
DEV1	007422	1364#	1366						
DISJMP=	160000	684#	904	942	943	944	960		
DISPRG	016756	1636	1919#						
DISTOP=	173000	685#	1012	1135					
DJMP =	160000	1252#	1347						
DL11IB=	175612	663#	664	982					
DL11IS=	175610	662#	663	730*	980	983*			
DL11OB=	175616	665#	990*	1037*	1038*				
DL11OS=	175614	664#	665	733*	751	753*			
DNOP =	164000	1253#							
DOCHAR	016110	1639	1648#						
DONE	007554	1405	1416#						
DONE0	001536	344#							
DONE1	001610	351	358#						
DSR	001006	197#	343*						
DSTOP =	173400	1255#							
DSWR =	177570	182#	187	505					
DUMP	001032	207#	226*	229*	247*	249*	250	255*	
D2BTYP	002230	376*	380*	459#	464				
END	001524	342#							
ENDCOR	006036	740#	741						
ERROR1	001136	223	234#						
ERROR2	001210	252#							
ERRVEC=	000004	183#	503	504*	511*				
FF	006256	828	858#						
FFLOOP	006262	853	860#	863					
FILCNT	001026	205#	420*	425*					
FILE0	007214	1277	1280#	1348					
FILLER	001024	204#	420						
GETCHR	006564	811	1002#	1024					
GETDL	006516	980#	985	989	991	1002	1006		
GETDL1	006546	981	988#						
GETEXT	006650	1004	1010	1030#					
GETSIX	006630	1024#	1062	1077					
GET8	006664	1062#	1172						
GET8TB	006700	1065	1067#						
GET81	006712	1067	1077#						

GET82	006740	1068	1089#						
GET83	006772	1069	1104#						
GET84	006706	1073#							
CRAPHX=	120000	1229#	1329						
GRAPHY=	124000	1230#	1340						
GTBUSE	006424	913#	936						
GTCHL	016312	1736#	1738						
GTCHP	016304	1733#	1742	1744					
GTCHR	016306	1642	1724	1735#	1756				
GTNP	016366	1746	1753#						
GTP84	016144	1675	1677#						
GTSIX	016262	1673	1682	1724#					
GT40PC=	172000	670#	671	913*	947	1277*			
GT40SR=	172002	671#	855*						
GT8	016132	1673#	1841						
GT8P =	016152	1680#	1718	1719	1720	1721			
GT8TB =	016250	1678	1716#						
GT81	016152	1682#	1718						
GT82	016200	1693#	1719						
GT83	016232	1707#	1720						
GT84	016142	1676#	1721						
HEADER	006474	903	942	948	954#				
HSR =	177550	1355#	1387						
ICNT	001030	206#	215*	342*					
ICOUNT	002210	440	447#						
IMAGE	001004	196#	315	515*	518*				
INCR =	000100	1266#	1324						
INSERT	006350	836	844	882	890#				
INSRTL	006406	894	897	902#					
INSRTX	006422	892	907#						
INTX =	040000	1267#	1285	1289	1293	1297			
INT0 =	002000	1233#	1284						
INT1 =	002200	1234#	1303						
INT2 =	002400	1235#	1288						
INT3 =	002600	1236#	1305						
INT4 =	003000	1237#	1292	1315					
INT5 =	003200	1238#	1307						
INT6 =	003400	1239#	1296						
INT7 =	003600	1240#	1309						
ITAL0 =	000040	1259#							
ITAL1 =	000060	1260#							
JMPADD=	007012	678#	870	881*	960	1012*	1135*		
JMPDIS=	160000	1596#	1634	1924					
KBDIB =	027562	668#	990	1586#	1775				
KBDIS =	027560	667#	668	988	1582#	1586	1627*	1773	1776*
LBAD	016610	1726	1728	1832#	1858				
LF	006300	826	867#						
LFLOOP	006310	872#	875	877					
LFOUND	006330	873	879#						
LFSUB	006304	861	870#						
LGWRD	016646	1819	1824	1847#	1854				
LINE0 =	000004	1247#	1284						
LINE1 =	000005	1248#	1288						
LINE2 =	000006	1249#	1292						
LINE3 =	000007	1250#	1296						
LJMP	016666	1823	1854#						

LJMP1	016726	1860	1868#						
LLD2	016532	1813#	1816	1830					
LLD3	016576	1827#	1839						
LLD4	016624	1828	1838#						
LOAD	016520	1751	1809#						
LOADER	007012	1008	1135#						
LOGICA	001600	174	354#						
LONGV =	110000	1227#	1284	1288	1292	1296			
LOOP	007444	1373#	1380						
LOOP1	007522	1404#	1409						
LOOP2	007526	1406#	1407	1412					
LPDARK=	000200	1258#							
LPLITE=	000300	1257#							
LPOFF =	000100	1242#							
LPON =	000140	1243#							
LPTR	016630	1814	1817	1827	1841#	1847	1849	1856	
LSR =	177560	1356#	1386						
L.BAD	007100	1026	1028	1162#	1193				
L.GWRD	007126	1147	1152	1179#	1189				
L.HALT	007200	1201#							
L.JMP	007146	1151	1189#						
L.JMP1	007202	1199	1203#						
L.LD2	007022	1140#	1143	1159					
L.LD3	007066	1156#	1168						
L.LD4	007110	1157	1167#						
L.PTR	007114	1141	1145	1156	1172#	1179	1181	1191	
MAJOR	016072	1642#	1646						
MAXSX =	017600	1272#							
MAXSY =	000077	1273#							
MAXX =	001777	1268#	1285	1293					
MAXY =	001377	1269#	1282	1290	1298				
MINSUY=	000100	1274#							
MINUSX=	020000	1270#	1271	1290	1293				
MINUSY=	020000	1271#							
M7	002336	370	489#						
M8	002361	368	391	394	396	490#			
M9	002365	379	491#						
NORMAL	006212	815	824	836#					
NOTHER	006042	738	744#						
NUMLIN=	000040	679#	768	769	858	944			
NXTCHR	006132	811#	813	819	838	848	856	864	867
ORIGIN=	166000	660#							
OTHER	007726	1438	1443	1467	1482	1490	1494#		
OUTLIT	006652	734	1037#	1162	1195				
O2A	002232	377	381	460#					
O2AA	002262	469#	481						
POINT =	114000	1228#	1280	1300	1312	1321	1325	1336	
PRESTR	006624	1013#	1164						
PRGO	001050	215#	358						
PRGOR	001054	216#	217						
PRG1	001614	190	364#	398					
PRG1A	001652	372#							
PRG1B	001670	376#	392						
PRG1C	001706	380#	388						
PRG1D	001756	386	393#						
PRMTRS	001040	189	210#						

TAB	006222	825	843#	847															
TABLE	007562	1400	1420#																
TABLEX	002652	324	542#																
TABLEO	002536	321	523#																
TABOOT	007500	1397#	1427																
TACS =	177500	1395#	1397																
TAPES	007662	1449	1458	1471#															
TCVEC	007774	1512#																	
TC11	007620	1448#	1512																
TERM	001036	209#	407*	409															
TMPEND=	007776	676#	731	765															
TM11	007636	1457#																	
TPCSR =	177564	179#	345	348	374	382	412	417	423	429	460	470	485*						
TPDBR =	177566	180#	344*	347*	384*	419*	422*	427*	431*	472*									
TYPEM	001776	367	369	378	390	393	395	405#											
TYPEMA	002006	408#	428	432															
TYPEMB	002034	410	415#																
TYPEMC	002122	416	429#																
T1	001066	221#																	
T1A	001104	224#	232	237															
T1B	001150	233	238#																
T2	001152	244#																	
T2A	001170	247#	254	272															
T2B	001220	251	255#	259															
T2C	001236	256	260#	264															
T2D	001252	261	265#	269															
T2E	001266	266	270#																
T3	001276	280#																	
T3A	001320	284#	287																
T3AA	001312	283#	303																
T3B	001332	283	288#																
T3C	001342	290#	293																
T3D	001354	288	294#																
T3E	001364	296#	299																
T3F	001400	294	300#																
T4	001420	314#																	
T4A	001434	317#	334	339															
T4B	001450	322#	327																
T4C	001470	323	328#																
T4D	001500	325	332#																
T4E	001510	320	331	336#															
VT	006244	827	851#																
VT40PC=	022000	1581#	1640*																
WORDS	001002	195#	222	245	282	338	371	514*	517*										
.	= 017000	164#	168	170#	173#	175#	185#	188#	193#	346	375	383	413	418					
		424	430	471	492#	715#	1376	1400	1479	1501	1611#	1680	1716	1866					

. ABS. 017000 000 CON RW ABS LCL I

ERRORS DETECTED: 0

CDGTDD,CDGTDD/CRF=CDGTDD
 RUN-TIME: 24.6 SECONDS
 RUN-TIME RATIO: 18/7=2.5

SCROLLING ROM BOOTSTRAP FOR THE GT40
CDGTDD.P11 31-AUG-79 11:12

MACY11 30G(1063) 31-AUG-79^{H 4} 12:59 PAGE 17-5
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0046

CORE USED: 7K (13 PAGES)