

GT40,VT11

GT40 QUICK VERIFY
CDGTED0

AH-7930D-MC

COPYRIGHT 75-80
FICHE 1 OF 1

JAN 1980

digital

MADE IN USA

This microfiche card contains a grid of frames, each containing a small table of data. The data is organized into columns and rows, with some frames containing headers and footers. The text is too small to read accurately but appears to be a list of items or a data table.

Frame	Column 1	Column 2	Column 3	Column 4
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20

.REM -

IDENTIFICATION

PRODUCT CODE: AC-7928D-MC
PRODUCT NAME: CDGTED0 GT40 QUICK VERIFY
DATE: AUG. 1979
MAINTAINER: DIAGNOSTIC GROUP

COPYRIGHT (C) 1975, 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 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).

THIS PROGRAM IS A QUICK GO-NOGO TEST OF THE GT40 SYSTEM. THE PURPOSE OF THIS TEST IS TO QUICKLY IDENTIFY ANY PROBLEM IN THE SYSTEM. THE PROGRAM WILL START THE DISPLAY AND THEN INITIATE THE COMMUNICATION LINE. TWO BACKGROUND TASKS ARE EXECUTED, THE FIRST IS A GT-40 ROM VERIFY TEST. THE SECOND TASK IS A WORSE CASE NOISE TEST THRU MEMORY.

2. REQUIREMENTS

2.1 EQUIPMENT

GT40 SYSTEM (11/05, DISPLAY PROCESSOR AND VR14 SCOPE)
MODEM TEST CONNECTOR WHICH CONNECTS DATA OUT TO DATA IN.

2.2 STORAGE

THIS PROGRAM USED MEMORY LOCATIONS 0-7776 AND 16000-16776 <2K OF MEMORY>.

3. LOADING PROCEDURE

3.1 METHOD

PROCEDURE FOR NORMAL BINARY TAPES SHOULD BE FOLLOWED.

4. STARTING PROCEDURE

4.1 CONTROL SWITCH SETTINGS

CONSOLE SW 09 = 0	ROM PRESENT AS SPECIFIED BY SW 08
CONSOLE SW 09 = 1	NO ROM PRESENT
CONSOLE SW 08 = 0	TEST AS VERSION 2 OR 3 ROM (512. WORDS)
CONSOLE SW 08 = 1	TEST AS VERSION 1 ROM (256. WORDS)

4.2 STARTING ADDRESS OR ADDRESSES

200 IS THE ONLY STARTING ADDRESS OF THIS TEST

5. OPERATING PROCEDURE

ONCE STARTED THE TEST WILL RUN IN THEIR NORMAL MANNER WITHOUT OPERATOR INTERVENTION OR SWITCH SELECTION. THE OPERATOR MUST VERIFY THE DATA RETURNING FROM THE COMMUNICATION LINE BY COMPARING 'COM OUTPUT' TO 'COM INPUT' ON THE DISPLAY SCREEN. BY TYPING ON THE CONSOLE KEYBOARD, THE CHARACTER AND OCTAL VALUE WILL BE DISPLAYED.

6. ERRORS

THE PROGRAM WILL ONLY HALT ON AN ERROR. THE PROGRAM DOES NOT CONTAIN FACILITES FOR REPORTING MESSAGES OR ERROR CONDITIONS.

7. RESTRICTIONS

A COMMUNICATION TEST PLUG MUST BE INSTALLED ON THE DL-11.

8. MISCELANEOUS

8.1 EXECUTION TIME

THE TEST WILL TAKE APPROXIMATELY 10 SECONDS FOR COMPLETION AND WILL RING THE 'GT-40' BELL.

8.2 DEVICE ADDRESS PROGRAM LOCATIONS

LOCATION 1000 CONTAINS THE GT40 DEVICE ADDRESS
LOCATION 1002 CONTAINS THE GT40 INTERRUPT VECTOR.
LOCATION 1004 CONTAINS THE GT40 INTERRUPT LEVEL.
LOCATION 1006 CONTAINS THE DL-11 DEVICE ADDRESS.
LOCATION 1010 CONTAINS THE DL-11 INTERRUPT VECTOR.
LOCATION 1012 CONTAINS THE DL-11 INTERRUPT LEVEL.
LOCATION 1014 CONTAINS THE GT-40 ROM BOOTSTRAP ADDRESS.
LOCATION 1016 CONTAINS THE GT-40 ROM WORD LENGTH.

8.3 XXDP/ACT/APT

THE PROGRAM IS CHAINABLE UNDER XXDP/ACT BUT CANNOT BE SCRIPTED UNDER APT.

9. PROGRAM DESCRIPTION

9.1 DISPLAY FILE <FORGROUND TASK>

THE DISPLAY FILE IS A COMPACT VISUAL TEST OF ALL GT40 DISPLAY INSTRUCTIONS. A BOX OUTLINING THE SCREEN WITH DIFFERENT LINE TYPE VALUE IS DISPLAYED. THREE PAIRS OF ASCII STRINGS ARE ALSO DISPLAYED TO TEST THE CHARACTER LOGIC. THE FIRST LINE OF A STRING IS DISPLAYED IN 'NORMAL' FONT THE SECOND LINE OF A STRING IS DISPLAYED IN 'ITALICS'. ALSO INCLUDED IN THIS VISUAL TEST ARE THE 8 DIFFERENT INTENSITY LEVELS. THE DISPLAY PATTERN IS ENHANCED BY THE USE OF BLINKING OCTAGONS AND MOVING SINE WAVES. THE DISPLAY PATTERN ALSO SERVES AS FOR VISUAL INSPECTION OF THE COMMUNICATION LINE DATA. ALL LINES AND CHARACTERS ARE ENABLED FOR LIGHT-PEN INTERACTION EXCEPT FOR THE LARGEST OCTAGON. UPON LIGHT-PEN HIT, THE TEXT 'LIGHT-PEN HIT' WILL BE DISPLAYED NEAR CENTER SCREEN.

9.2 COMMUNICATION DATA <FORGROUND TASK>

THE DATA PRESENTED TO THE COMMUNICATION LINE APPEARS ON THE DISPLAY SCREEN AS FOUND AT 'COM OUTPUT'. (DECGRAPHIC-11 DISPLAY TERMINAL GT40 VR14) THE DATA ECHOED BACK BY THE TEST CONNECTOR IS DISPLAYED ON THE SCREEN AS FOUND AT 'COM INPUT'. A VISUAL TEST OF THE DATA MUST BE PERFORMED.

9.3 ROM VERIFY TEST <BACKGROUND TASK>

THIS TEST VERIFIES THE DATA CONTAINED IN THE GT-40 ROM BOOTSTRAP.

9.4 WORSE CASE NOISE TEST <BACKGROUND TASK>

THIS IS A BACKGROUND TEST OF ALL AVAILABLE MEMORY. A SMALL PROGRAM IS LOADED INTO ALL EXISTING MEMORY AND THEN EXECUTED THRU THE REMAINDER OF MEMORY.

9.5 KEYBOARD DATA <FORGROUND TASK>

UPON DEPRESSING A KEYBOARD KEY, THE OCTAL VALUE WILL BE DISPLAYED AND ECHO ONTO THE SCREEN.

-

```

.TITLE GT40 QUICK VERIFY MAINDEC-11-CDGTE-D
.ENABL ABS,AMA

199
200
457
458 000000 000000 .=0 ;0-776 IS LOADED WITH .+2, HALT
459 000000 000000 HALT
460 000002 000000 HALT
465 000024 000024 .=24
466 000024 001674 PWRFL ;POWER FAIL VECTOR
467 000026 000340 340
468 000046 000046 .=46
469 000046 002752 LOGICAL ;XXDP/ACT END PASS POINTER
470 000052 000052 .=52
471 000052 000000 0 ;NO MANUAL INTERVENTION REQUIRED
472 000004 000004 ERRVEC=4
473 177570 177570 DSWR=177570 ;11/45 LIGHT DISPLAY REGISTER
474 000170 000170 .=170
475 000170 000000 SWREG: .WORD 0
476 000200 000200 .=200
477 000200 000137 001022 JMP STARTB
478 001000 001000 .=1000
479 001000 172000 GTADD: 172000 ;GT-40 ADDRESS
480 001002 000320 GTVCT: 320 ;GT-40 VECTOR
481 001004 000200 GTBRL: 200 ;GT-40 BR LEVEL
482
483 001006 175610 DLADD: 175610 ;DL-11 ADDRESS
484 001010 000300 DLVCT: 300 ;DL-11 VECTOR
485 001012 000240 DLBRL: 240
486
487 001014 166000 ROMADD: 166000 ;ROM STARTING ADDRESS
488 001016 001000 WORDS: 512
489 001020 006000 IMAGE: 6000
490
491 001022 012706 000500 STARTB: MOV #500,SP ;LOAD THE STACK POINTER
492 001026 012777 000340 000176 MOV #340,@PSW ;RAISE PSW
493 001034 004737 001316 JSR PC,INITGT ;INIT DEVICE ADDRESSES
494 001040 005037 001020 CLR IMAGE ;PRESET FOR NO ROM SELECTED
495 001044 032777 001000 000156 BIT #1000,@SWR ;TEST FOR ROM SELECTED SWITCH
496 001052 001021 BNE 2$ ;NO ROM SELECTED
497 001054 032777 000400 000146 BIT #400,@SWR ;TEST ROM SWITCH
498 001062 001007 BNE 1$ ;BR IF SET
499 001064 012737 001000 001016 MOV #512.,WORDS ;ASSUME VER. 2 ROM
500 001072 012737 006000 001020 MOV #START,IMAGE ;LOAD IMAGE ADDRESS
501 001100 000406 BR 2$ ;START TEST
502 001102 012737 000400 001016 1$: MOV #256.,WORDS ;SELECT VER. 1 ROM
503 001110 012737 016000 001020 MOV #STARTA,IMAGE ;LOAD IMAGE ADDRESS
504 001116 005077 000042 2$: CLR @DLODBR ;CLEAR OUTPUT
505 001122 005077 000036 CLR @DLODBR
506 001126 004737 001720 JSR PC,DOCORE ;SET UP CORE SIZE
507 001132 004737 001234 JSR PC,PRIME ;INIT THE DEVICES
508 001136 005077 000070 CLR @PSW
509 001142 000137 002510 JMP OVER ;EXECUTE BACKGROUND TASK
    
```

```
511
512
513 001146 172000      GTPL: 172000      ;DISPLAY PC
514 001150 172002      GTSR: 172002      ;DISPLAY STATUS REG.
515 001152 172004      GTXPOS: 172004     ;DISPLAY X REGISTER
516 001154 172006      GTYPOS: 172006     ;DISSPLAY Y REGISTER
517
518 001156 175610      DLICSR: 175610     ;DL-11 STATUS
519 001160 175612      DLIDBR: 175612     ;DL-11 BUFFER
520 001162 175614      DLOCSR: 175614     ;DL-11 STATUS
521 001164 175616      DLODBR: 175616     ;DL-11 BUFFER
522
523 001166 000320      GTDONE: 320        ;DISPLAY DONE VECTOR
524 001170 000322      GTDNE1: 322
525
526 001172 000324      GTLPH: 324         ;DISPLAY LIGHT-PEN VECTOR
527 001174 000326      GTLPH1: 326
528
529 001176 000330      GTSOTM: 330        ;DISPLAY SHIFT-OUT/ TIME-OUT VECTOR
530 001200 000332      GTSOT1: 332
531
532 001202 000300      DLIVT: 300
533 001204 000302      DLIVT1: 302
534 001206 000304      DLOVT: 304
535 001210 000306      DLOVT1: 306
536
537 001212 177560      TKS: 177560
538 001214 177562      TKB: 177562
539 001216 177564      TPS: 177564
540 001220 177566      TPB: 177566
541
542 001222 000060      KRBT: 60
543 001224 000062      KRBT1: 62
544
545 001226 000200      KRBBRL: 200
546
547 001230 177570      SWR: 177570
548 001232 177776      PSW: 177776
549
550 001234 012777 002772 177704 PRIME: MOV #FILE00,@GTPC ;START THE DISPLAY
551 001242 012777 000100 177712      MOV #100,@DLOCSR ;ENABLE DL OUTPUT
552 001250 012777 000100 177700      MOV #100,@DLICSR ;ENABLE DL INPUT
553 001256 012777 000100 177726      MOV #100,@TKS ;ENABLE KEYBOARD
554 001264 113777 005402 177672      MOVB BUFF1,@DLODBR ;OUTPUT A CHAR
555 001272 012737 000001 002310      MOV #1,PPNT ;PRESET PRINT POINTER
556 001300 005037 002312      CLR RPNT ;CLEAR READ BUFFER
557 001304 005037 002504      CLR KPNT
558 001310 000207      RTS PC ;EXIT
559
560 001312 017476      SIZE: 17476
561 001314 000000      GTDLY0: 0
```

```

563 001316 012700 001146 INITGT: MOV #GTPC,R0 ;LOAD STARTING ADDRESS
564 001322 013701 001000 MOV GTADD,R1 ;SAVE VALUE
565 001326 004737 001400 JSR PC,LOADRO ;LOAD GT ADDR
566 001332 013701 001006 MOV DLADD,R1 ;LOAD STARTING ADDRESS <DL-11>
567 001336 004737 001400 JSR PC,LOADRO ;LOAD DL-11 ADDRESSES
568 001342 013701 001002 MOV GTVCT,R1 ;LOAD VECTOR VALUE
569 001346 004737 001400 JSR PC,LOADRO ;LOAD GT-40 VECTORS
570 001352 010110 MOV R1,(R0)
571 001354 062720 000010 ADD #10,(R0)+ ;LOAD GT TIME-OUT
572 001360 010110 MOV R1,(R0)
573 001362 062720 000012 ADD #12,(R0)+
574 001366 013701 001010 MOV DLVCT,R1 ;LOAD VECTOR VALUE
575 001372 004737 001400 JSR PC,LOADRO ;LOAD DL-11 VECTORS
576 001376 000436 BR INGT ;BR
577 001400 010120 LOADRO: MOV R1,(R0)+ ;LOAD DONE
578 001402 010110 MOV R1,(R0)
579 001404 062720 000002 ADD #2,(R0)+
580 001410 010110 MOV R1,(R0)
581 001412 062720 000004 ADD #4,(R0)+ ;LOAD DONE
582 001416 010110 MOV R1,(R0)
583 001420 062720 000006 ADD #6,(R0)+ ;LOAD PSW
584 001424 013746 000004 MOV @#ERRVEC,-(SP) ;SAVE VECTORS CONTENTS
585 001430 012737 001456 000004 MOV #1$,@#ERRVEC ;SET UP FOR TRAP
586 001436 012737 177570 001230 MOV #DSWR,@#SWR ;SET UP TO TEST FOR SWITCH REGISTER
587 001444 022777 177777 177556 CMP #-1,@#SWR ;TEST FOR SWITCH REGISTER
588 001452 001005 BNE 3$ ;SWITCH REGISTER IS PRESENT
589 001454 000401 BR 2$ ;NO SWITCH REGISTER
590 001456 022626 1$: CMP (SP)+,(SP)+ ;POP 2 WORDS OFF STACK
591 001460 012737 000170 001230 2$: MOV #SWREG,@#SWR ;SET UP FOR SOFTWARE SWITCH REGISTER
592 001466 012637 000004 3$: MOV (SP)+,@#ERRVEC ;RESTORE VECTORS CONTENTS
593 001472 000207 RTS PC ;EXIT
594
595 001474 012777 002010 177464 INGT: MOV #GTSTOP,@GTDONE ;LOAD DONE VECTOR
596 001502 013777 001004 177460 MOV GTBRL,@GTDNE1
597 001510 012777 002100 177454 MOV #GTLPHEN,@GTLPH ;LOAD LIGHT-PEN VECTOR
598 001516 013777 001004 177450 MOV GTBRL,@GTLPH1
599 001524 012777 002116 177444 MOV #GTSHIF,@GTSOTM ;LOAD SHIFT-OUT VECTOR
600 001532 013777 001004 177440 MOV GTBRL,@GTSOT1
601 001540 012737 000040 001314 MOV #40,GTDLY0
602 001546 012737 005732 005702 MOV #FILEOC,FILEOA
603 001554 012737 174104 003054 MOV #STATSB!INCR+4,GRPINC
604 001562 012700 005476 MOV #BUFF2,R0
605 001566 005020 INTD: CLR (R0)+
606 001570 022700 005546 CMP #BUFF2+50,R0
607 001574 001374 BNE INTD
608 001576 012700 005572 MOV #BUFF3,R0 ;SET UP KRB BUFFER

```



```

610 001602 005020          INTE: CLR      (R0)+
611 001604 022700 005642  CMP      #BUFF3+50,R0
612 001610 001374          BNE      INTE
613 001612 105037 005657  CLRB    OCTA
614 001616 105037 005660  CLRB    OCTA+1
615 001622 105037 005661  CLRB    OCTA+2
616 001626 012777 002206 177346  MOV     #DLIN,@DLIVT
617 001634 013777 001012 177342  MOV     DLBRL,@DLIVT1
618 001642 012777 002124 177336  MOV     #DLOUT,@DLOVT
619 001650 013777 001012 177332  MOV     DLBRL,@DLOVT1
620 001656 012777 002320 177336  MOV     #KBIN,@KRBVT ;LOAD KRB VECTOR
621 001664 013777 001226 177332  MOV     KRBBRL,@KRBVT1 ;LOAD PSW
622 001672 000207          RTS     PC
623
624 001674 012737 001704 000024 PWRFL: MOV     #PWRUP,@#24 ;LOAD VECTOR
625 001702 000000          HALT
626
627 001704 000005          PWRUP: RESET
628 001706 012737 001674 000024  MOV     #PWRFL,@#24
629 001714 000137 001022          JMP     STARTB ;RESTART AT BEGINING
630
631 ;SUBROUTINE TO DETERMINE THE SIZE OF CORE
632 ; AND SET UP LOCATION SIZE WITH THE VALUE
633
634 001720 012737 001754 000004 DOCORE: MOV     #2$,@#4 ;SET UP FOR NEM
635 001726 012701 017776          MOV     #17776,R1 ;SET UP ADDRESS
636 001732 005000          CLR     R0
637 001734 062701 020000 1$: ADD     #20000,R1 ;MOVE TO THE NEXT BANK
638 001740 005200          INC     R0 ;INC BANK COUNTER
639 001742 005711          TST    (1) ;TIMEOUT ?
640 001744 022701 157776          CMP     #157776,R1 ;END ?
641 001750 001371          BNE    1$
642 001752 000404          BR     3$
643 001754 022626          2$: CMP     (SP)+,(SP)+ ;POP THE STACK X2
644 001756 005300          DEC     RC ;DECREMENT BANK COUNT
645 001760 162701 020000          SUB     #20000,R1 ;RESTORE R1
646 001764 012737 000006 000004 3$: MOV     #6,@#4 ;RESET BUSS ERROR
647 001772 010137 001312          MOV     R1,SIZE
648 001776 162737 007776 001312  SUB     #7776,SIZE ;BACK PAST LOADER
649 002004 000137 010000          JMP     CORTST
  
```

```

651          .ENABLE AMA
652
653          ;INTERRUPT SERVICE FOR THE GT STOP INTERRUPT
654
655 002010 005777 177134  GTSTOP: TST    @GTSR          ;TEST STOP
656 002014 100403          BMI    1$
657 002016 000000          HALT
658 002020 000137 001022  JMP    STARTB          ;ERROR, STOP INTERRUPT BUT NO STOP FLAG
659                                     ;RESTART TEST
660 002024 005337 001314  1$:   DEC    GTDLY0          ;DECREMENT DELAY
661 002030 001014          BNE    GTST1           ;BRANCH IF NOT
662 002032 012737 000040 001314  MOV    #40,GTDLY0      ;RESET DELAY
663 002040 005237 003054          INC    GRPINC          ;UPDATE GRAPH INCREMENT
664 002044 022737 174110 003054  CMP    #STATSB!INCR+10,GRPINC ;TEST FOR INCREMENT
665 002052 001003          BNE    GTST1           ;BRANCH IF NOT
666 002054 012737 174100 003054  MOV    #STATSB!INCR,GRPINC ;RESET GRAPH INCREMENT
667 002062 012737 005732 005702  GTST1: MOV    #FILEOC,FILEOA
668 002070 012777 000001 177050  MOV    #1,@GTPC          ;RESUME THE DISPLAY
669 002076 000002          RTI
670                                     ;EXIT
671 002100 012737 005704 005702  GTLPEN: MOV   #FILEOB,FILEOA
672 002106 012777 000001 177032  MOV    #1,@GTPC          ;RESUME THE DISPLAY
673 002114 000002          RTI
674
675 002116 000000          GTSHIF: HALT
676 002120 000137 001022  JMP    STARTB          ;GT-40 SHIFT-OUT/TIME-OUT ERROR

```

```

678
679      ;INTERRUPT SERVICE FOR THE DL PRINTER
680
681 002124 105777 177032      DLOUT:  TSTB   @DLOCSR      ;TEST FOR DONE
682 002130 100403              BMI     .+10
683 002132 000000              HALT
684 002134 000137 001022      JMP     STARTB      ;ERROR, PRINTER INTERRUPT BUT NO PRINTER FLAG
685                                     ;RESTART TEST
686 002140 010446              MOV     R4,-(SP)
687
688 002142 013704 002310      DLOUTA: MOV     PPNT,R4      ;LOAD R4 WITH BYTE POINTER
689 002146 116437 005402 002314  MOVB   BUFF1(R4),PUNCHR    ;LOAD A CHARACTER TO BE OUTPUTTED
690 002154 005237 002310      INC     PPNT         ;UPDATE CHARACTER POINTER
691 002160 022737 000050 002310  CMP     #40.,PPNT      ;TEST FOR END
692 002166 001002              BNE    DLOUTB
693 002170 005037 002310      CLR     PPNT         ;CLEAR PUNCH POINTER
694
695 002174 113777 002314 176762  DLOUTB: MOVB  PUNCHR,@DLODBR ;OUTPUT A CHARACTER
696 002202 012604              MOV     (SP)+,R4      ;RESTORE R4
697 002204 000002              RTI     ;EXIT
698
699      ;INTERRUPT SERVICE FOR THE DL READER
700
701 002206 105777 176744      DLIN:   TSTB   @DLICSR      ;TEST FOR DONE
702 002212 100403              BMI     .+10
703 002214 000000              HALT
704 002216 000137 001022      JMP     STARTB      ;NOT DL INPUT FLAG
705                                     ;RESTART TEST
706                                     MOV     R4,-(SP)      ;SAVE R4
707 002222 010446              MOV     RPNT,R4
708 002224 013704 002312      MOVB   @DLIDBR,REDCHR    ;READ A CHARACTER
709 002230 117737 176724 002316  BIC     #177600,REDCHR    ;MASK CHARACTER
710 002236 042737 177600 002316  MOVB   REDCHR,BUFF2(R4)  ;PUT CHARACTER INTO THE BUFFER
711 002244 113764 002316 005476  INC     RPNT         ;UPDATE READ POINTER
712 002252 005237 002312      CMP     #40.,RPNT      ;TEST FOR END
713 002256 022737 000050 002312  BNE    DLINB
714 002264 001002              CLR     RPNT
715 002266 005037 002312      DLINB: MOV     RPNT,R4
716 002272 013704 002312      MOVB   #177,BUFF2(R4)   ;ADD CURSOR
717 002276 112764 000177 005476  MOV     (SP)+,R4      ;RESTORE R4
718 002304 012604              RTI     ;EXIT
719 002306 000002
720 002310 000000      PPNT:   0
721 002312 000000      RPNT:   0
722 002314 000240      PUNCHR: 240
723 002316 000240      REDCHR: 240
  
```

```

725
726
727
728      ; INTERRUPT SERVICE FOR THE KEYBOARD
729
730 002320 105777 176666      KBIN:  TSTB   @TKS           ; TEST FOR DONE
731 002324 100403             BMI     .+10
732 002326 000000             HALT
733 002330 000137 001022     JMP     STARTB       ; NOT KRB INPUT FLAG
734                                     ; RESTART
735 002334 010346             MOV     R3, -(SP)     ; SAVE R3
736 002336 010446             MOV     R4, -(SP)     ; SAVE R4
737 002340 013704 002504     MOV     KPNT, R4
738 002344 117737 176644 002506  MOVB   @TKB, KBCHR    ; READ CHARACTER
739 002352 042737 177600 002506  BIC     #177600, KBCHR ; MASK
740 002360 113764 002506 005572  MOVB   KBCHR, BUFF3(4) ; SAVE THE CHAR
741 002366 005237 002504             INC     KPNT         ; UPDATE POINTER
742 002372 022737 000050 002504  CMP     #40., KPNT    ; TEST FOR END
743 002400 001002             BNE     1$
744 002402 005037 002504             CLR     KPNT         ; CLEAR POINTER
745 002406 013704 002504      1$:  MOV     KPNT, R4
746 002412 112764 000177 005572  MOVB   #177, BUFF3(R4) ; ADD CURSOR
747
748      ; UPDATE OCTAL READOUT
749
750 002420 013703 002506     MOV     KBCHR, R3    ; GET CHAR
751 002424 004737 002470     JSR     PC, 10$     ; LOAD BITS
752 002430 110437 005661     MOVB   R4, OCTA+2   ; SAVE BITS
753 002434 004737 002462     JSR     PC, 11$     ; MOVE BITS
754 002440 110437 005660     MOVB   R4, OCTA+1   ; SAVE BITS
755 002444 004737 002462     JSR     PC, 11$     ; MOVE BITS
756 002450 110437 005657     MOVB   R4, OCTA     ; SAVE BITS
757 002454 012604             MOV     (SP)+, R4    ; RESTORE R4
758 002456 012603             MOV     (SP)+, R3    ; RESTORE R3
759 002460 000002             RTI                    ; EXIT
760
761 002462 006003      11$:  ROR     R3
762 002464 006003             ROR     R3
763 002466 006003             ROR     R3
764 002470 010304      10$:  MOV     R3, R4           ; LOAD R4
765 002472 042704 177770     BIC     #177770, R4  ; MASK BITS
766 002476 062704 000060     ADD     #60, R4      ; MAKE A NUMBER
767 002502 000207             RTS     PC
768
769 002504 000000      KPNT:  0
770 002506 000240      KBCHR: 240
  
```

```

772 ;PART 1 OF THE BACKGROUND TASK
773
774 002510 012737 001000 002770 OVER: MOV #1000,PCOUNT ;SET UP EXECUTION COUNT
775
776 : COMPARE THE ROM DATA TO THE IMAGE DATA
777 : R0=WORD NUMBER
778 : R1=GOOD DATA
779 : R2=GOOD DATA
780 : R3=BAD ADDRESS
781 : R4=BAD DATA
782
783 002516 012700 000000 BACK: MOV #0,%0 ;SETUP INITIAL WORD NUMBER
784 002522 013701 001020 MOV IMAGE,%1 ;SET UP BUFFER
785 002526 001455 BEQ TMEM ;NO ROM SELECTED
786 002530 013703 001014 MOV ROMADD,%3 ;SET UP ROM ADDRESS
787 002534 011102 BACK1: MOV (%1),%2 ;READ A IMAGE WORD
788 002536 011304 MOV (%3),%4 ;READ A ROM WORD
789 002540 020204 CMP %2,%4 ;TEST FOR EQUAL
790 002542 001442 BEQ BACK2 ;BRANCH IF OK
791 002544 012705 010124 MOV #TABLE0,R5 ;LOAD PATCH POINTER
792 002550 020015 1$: CMP R0,(R5) ;TEST IF A PATCHED LOCATION
793 002552 001406 BEQ 2$ ;BR IF YES
794 002554 020527 010240 CMP R5,#TABLEX ;TEST IF END OF PATCH TABLE
795 002560 001407 BEQ 3$ ;BR IF END * MUST BE AN ERROR
796 002562 062705 000004 ADD #4,R5 ;UPDATE PATCH POINTER
797 002566 000770 BR 1$
798 002570 005725 2$: TST (R5)+
799 002572 011502 MOV (R5),R2 ;GET PATCHED VALUE
800 002574 020204 CMP R2,R4 ;COMPARE EXPECTED TO VALUE READ
801 002576 001424 BEQ BACK2 ;BR IF OK
802 002600 010037 002634 3$: MOV R0,R0SAV ;SAVE REGISTERS
803 002604 010137 002636 MOV R1,R1SAV
804 002610 010237 002640 MOV R2,R2SAV
805 002614 010337 002642 MOV R3,R3SAV
806 002620 010437 002644 MOV R4,R4SAV
807 002624 010537 002646 MOV R5,R5SAV
808 002630 000000 HALT ;ERROR ROM VALUE FAILED TO EQUAL THE
809 002632 000740 BR BACK1 ; THE EXPECTED
810 002634 000000 R0SAV: 0 ;CONTENTS OR REGS. AT ROM COMPARE HALT
811 002636 000000 R1SAV: 0
812 002640 000000 R2SAV: 0
813 002642 000000 R3SAV: 0
814 002644 000000 R4SAV: 0
815 002646 000000 R5SAV: 0
816
817 002650 022123 BACK2: CMP (%1)+,(%3)+ ;BUMP BOTH REGISTERS
818 002652 005200 INC %0 ;UPDATE WORD COUNTER
819 002654 023700 001016 CMP WORDS,%0 ;TEST FOR LAST WORD
820 002660 001325 BNE BACK1 ;BRANCK IF NOT LAST
821

```

```

823                                     ;PART 2 OF THE BACKGROUND TASK
824                                     ; EXECUTE WORSE CASE NOISE TEST THRU MEMORY
825
826 002662 004737 017000          TMMEM: JSR      PC,BUFFER      ;EXECUTE NOISE TEST
827 002666 005337 002770          DEC      PCOUNT      ;DONE PASS ?
828 002672 001311                  BNE      BACK        ;NO
829 002674 012777 000001 176246  MOV      #1,@GTSR    ;YES RING THE BELL
830 002702 012777 000207 176310  MOV      #207,@TPB   ;RING THE BELL
831 002710 105777 176302          1$:  TSTB     @TPS
832 002714 100375                  BPL      1$
833 002716 012777 000207 176274  MOV      #207,@TPB
834 002724 105777 176266          2$:  TSTB     @TPS
835 002730 100375                  BPL      2$
836 002732 005737 000042          TST      @42         ;TEST LOC. 42
837 002736 001664                  BEQ      OVER        ;BR IF =0
838 002740 000005                  RESET
839 002742 000005                  RESET
840 002744 000005                  RESET
841 002746 013700 000042          MOV      @42,R0      ;READ VALUE
842 002752 004710          LOGICAL: JSR      PC,(0)
843 002754 000240                  NOP
844 002756 000240                  NOP
845 002760 000240                  NOP
846 002762 000240                  NOP
847 002764 000137 001022          JMP      STARTB
848
849
850 002770 000000          PCOUNT: 0
  
```

```
852
853 002772 114140 FILE00: POINT!LPON
854 002774 000000 0
855 002776 001377 MAXY
856 003000 174300 STATSB.LPLITE
857
858 ;LINE THE EDGES OF THE SCREEN
859
860 003002 113004 LONGV.INT4!LINE0 ;TOP LINE
861 003004 041777 INTX!MAXX
862 003006 000000 0
863 003010 110005 LONGV!LINE1 ;RIGHT LINE
864 003012 040000 INTX
865 003014 021377 MINUSX!MAXY
866 003016 110006 LONGV!LINE2 ;BOTTOM LINE
867 003020 061777 INTX!MINUSX!MAXX
868 003022 000000 0
869 003024 110007 LONGV!LINE3 ;LEFT LINE
870 003026 040000 INTX
871 003030 001377 MAXY
872
873 ;SETUP THE X SINEWAVE
874
875 003032 114004 POINT!LINE0
876 003034 000400 400
877 003036 000200 200
878 003040 110000 LONGV
879 003042 041200 INTX+1200 ;DRAW X AXIS
880 003044 000000 0
881 003046 114000 POINT
882 003050 000440 440
883 003052 000200 200
884 003054 174104 GRPINC: STATSB!INCR+4 ;GRAPHPLOT THE X SINEWAVE
885
886 003056 124000 GRAPHY
888
889 ;SETUP THE Y SINEWAVE
890
891 003536 114000 POINT
892 003540 000200 200
893 003542 000040 40
894 003544 110000 LONGV ;DRAW Y AXIS
895 003546 040000 INTX
896 003550 001200 1200
897 003552 114000 POINT
898 003554 000200 200
899 003556 000100 100
900 003560 120000 GRAPHX ;GRAPHPLOT THE Y SINEWAVE
901
903
```

```
905
906 ;SETUP TO DISPLAY THE OCTAGONS
907
908 004240 114000 POINT
909 004242 001434 1434
910 004244 000724 724
911 004246 130030 RELATV!BLKON
913 004270 114000 POINT
914 004272 001430 1430
915 004274 000710 710
916 004276 130020 RELATV.BLKOFF
918 004320 114000 POINT
919 004322 001420 1420
920 004324 000660 660
921 004326 104030 SHORTV.BLKON
923 004350 114000 POINT
924 004352 001400 1400
925 004354 000600 600
926 004356 104020 SHORTV!BLKOFF
928 004400 114030 POINT!BLKON
929 004402 001360 1360
930 004404 000520 520
932 004450 114120 POINT.BLKOFF!LPOFF
933 004452 001340 1340
934 004454 000440 440
936
937 004520 114140 POINT!LPON
938 004522 000100 100
939 004524 001277 MAXY-100
940 004526 164000 DNOP
941 004530 170040 STATSA.ITALO
942 004532 100000 CHAR
944 004630 164000 DNOP
945 004632 170060 STATSA!ITAL1
946 004634 114000 POINT
947 004636 000100 100
948 004640 001247 MAXY-130
949 004642 100000 CHAR
951 004740 170040 STATSA!ITALO
952 004742 114000 POINT
953 004744 000220 220
954 004746 001177 MAXY-200
955 004750 100000 CHAR
957 005012 170060 STATSA!ITAL1
958 005014 114000 POINT
959 005016 000220 220
960 005020 001147 MAXY-230
961 005022 100000 CHAR
963 005064 170040 STATSA.ITALO
964 005066 114000 POINT
965 005070 000220 220
966 005072 001077 MAXY-300
967 005074 100000 CHAR
969 005140 170060 STATSA!ITAL1
970 005142 114000 POINT
971 005144 000220 220
```


972	005146	001047	MAXY-330
973	005150	100000	CHAR
975	005214	170040	STATSA!ITALO
976			
977			;SETUP INTENSITY LEVEL TEST
978	005216	114000	POINT
979	005220	000340	340
980	005222	001000	1000
981	005224	113604	LONGV.INT7!LINE0
982	005226	040400	INTX+400
983	005230	000000	0
984	005232	114000	POINT
985	005234	000340	340
986	005236	000740	740
987	005240	113400	LONGV.INT6
988	005242	040400	INTX+400
989	005244	000000	0
990	005246	114000	POINT
991	005250	000340	340
992	005252	000700	700
993	005254	113200	LONGV.INT5
994	005256	040400	INTX+400
995	005260	000000	0
996	005262	114000	POINT
997	005264	000340	340
998	005266	000640	640
999	005270	113000	LONGV!INT4
1000	005272	040400	INTX+400
1001	005274	000000	0
1002	005276	114000	POINT
1003	005300	000340	340
1004	005302	000600	600
1005	005304	112600	LONGV.INT3
1006	005306	040400	INTX+400
1007	005310	000000	0
1008	005312	114000	POINT
1009	005314	000340	340
1010	005316	000540	540
1011	005320	112400	LONGV.INT2
1012	005322	040400	INTX+400
1013	005324	000000	0
1014	005326	114000	POINT
1015	005330	000340	340
1016	005332	000500	500
1017	005334	112200	LONGV.INT1
1018	005336	040400	INTX+400
1019	005340	000000	0
1020	005342	114000	POINT
1021	005344	000340	340
1022	005346	000440	440
1023	005350	112000	LONGV.INT0
1024	005352	040400	INTX+400
1025	005354	000000	0
1026			
1027			;SETUP THE MESSAGA BUFFERS
1028			

1052	005646	000400				400
1053	005650	100000				CHAR
1054	005652	041517	020124	000		.ASCIZ /OCT /
1055	005657	000	000	000	OCTA:	.BYTE 0,0,0
1056	005662	164000				DNOP
1057	005664	.64000				DNOP
1058	005666	164000				DNOP
1059	005670	164000				DNOP
1060	005672	164000				DNOP
1061	005674	164000				DNOP
1062	005676	164000				DNOP
1063	005700	160000				DJMP
1064	005702	005732			FILEOA:	FILEOC
1065	005704	114000			FILEOB:	POINT
1066	005706	001000				1000
1067	005710	000440				440
1068	005712	100000				CHAR
1069	005714	044514	044107	026524		.ASCIZ /LIGHT-PEN HIT/
	005722	042520	020116	044510		
	005730	000124				
1070						.EVEN
1071	005732	173400			FILEOC:	DSTOP
1072	005734	160000				DJMP
1073	005736	002772				FILEOO
1074						

1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128

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


```
1184
1185
1186          :
1187          :
1188
1189
1190
1191          166000          ORIGIN=166000          ;ORIGIN OF THE BOOTSTRAP.
1192
1193          175610          DL11IS=175610          ;INPUT STATUS REGISTER OF DL11
1194          175612          DL11IB=DL11IS+2          ;INPUT CHARACTER FROM DL11
1195          175614          DL11OS=DL11IB+2          ;OUTPUT STATUS OF THE DL11
1196          175616          DL11OB=DL11OS+2          ;OUTPUT CHARACTER TO THE DL11
1197
1198          177560          KBDIS=177560          ;KEYBOARD INPUT STATUS
1199          177562          KBDIB=KBDIS+2          ;CURRENT CHARACTER FROM KEYBOARD.
1200
1201          172000          GT40PC=172000          ;GT40 PROGRAM COUNTER.
1202          172002          GT40SR=GT40PC+2          ;GT40 STATUS REGISTER ADDRESS.
1203
1204
1205          001000          BSTART=1000          ;START OF THE DISPLAY BUFFER
1206          007000          BLIMIT=7000          ;APPROXIMATE END OF THE DISPLAY BUFFER.
1207          007776          TMPEND=7776          ;LOCATION OF INITIALIZATION STACK.
1208          000004          CORSTR=4          ;LOCATION OF PDP-11 TRAP VECTOR.
1209          007012          JMPADD=BLIMIT+10.          ;WHERE THE POINTER IS TO FIRST CHAR ON SCREEN
1210          000040          NUMLIN=32.          ;NUMBER OF LINES ON TEXT TO SHOW ON THE SCREEN
1211
1212          005015          (RLF-5015          ;CARRIAGE RETURN - LINE FEED
1213          000175          ALTMOD=175          ;THE 'KEY' CHARACTER [I.E. ALTMODE].
1214
1215          160000          DISJMP=160000          ;THE GT40 JMP INSTRUCTION
1216          173000          DISTOP=173000          ;THE GT40 STOP DISPLAY INSTRUCTION.
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235          .SBTTL  INITIALIZATION AND RESTART CODE
```

```

1237
1238
1239
1240          :          GT40 BOOTSTRAP CODE
1241          :          -----
1242
1243
1244
1245
1246          006000          .=6000
1247
1248          :          .-ORIGIN          ; REFINE ORIGIN OF THE BOOTSTRAP.
1249
1250
1251
1252
1253
1254          :          COLD INITIALIZATION CODE
1255          :          -----
1256
1257
1258
1259
1260          006000 000005          START: RESET          ;RESET ALL HARDWARE NOW.
1261          006002 012737 000007 175610          MOV          #7,DL11IS          ;INITIALIZE DL-11 INPUT NOW.
1262          006010 012706 007776          MOV          #TMPEND,SP          ;ESTABLISH A GOOD TEMPORARY STACK
1263          ;POINTER FOR CORE SEARCH.
1264          006014 005237 175614          INC          DL110S          ;SET BREAK BIT
1265          006020 004337 166652          JSR          SCAN,OUTLIT!160000          ;FOR 2 CHARACTER TIMES
1266          006024 000000          .WORD 0          ;SEND TWO ZERO'S
1267
1268          006026 012703 000004          MOV          #CORSTR,SCAN          ;GET ADDRESS OF BAD CORE TRAP VECTOR.
1269          006032 012723 166042          MOV          #NOTHERE!160000,(SCAN)+ ;AND INSERT A POINTER TO US THERE.
1270
1271          006036 005023          ENDCOR: CLR          (SCAN)+          ;NOW CLEAR ALL OF MEMORY BEYOND THE POINTER,
1272          006040 000776          BR          ENDCOR          ;UNTIL WE RUN OUT OF MEMORY AND TRAP.
1273
1274
1275          006042 005743          NOTHER: TST          -(SCAN)          ;WHEN WE TRAP OUT, WE COME HERE.
1276          ;WE BACK UP POINTER TO GOOD CORE.
1277          ;NOTE THAT IF WE TRAP OUT AGAIN, IT
1278          ;IS STILL OK, BECAUSE WE WILL LOOP
1279          ;UNTIL WE GET A GOOD CORE ADDRESS.
1280          006044 010306          MOV          SCAN,SP          ;WHEN WE GET ONE, THAT IS LAST LOCATION
1281          ;IN THE MACHINE, AND HENCE OUR SP.
1282          006046 105737 175614          1$: TSTB          DL110S          ;SEE IF BREAK IS DONE
1283          006052 100375          BPL          1$          ;NO GO BACK
1284          006054 005037 175614          CLR          DL110S          ;CLEAR BREAK BIT
1285
1286
1287
1288
1289
1290
1291          :          RESTART INITIALIZATION CODE WHEN COMMUNICATIONS IS WORKING.
1292          :          -----
  
```

```
1293
1294
1295
1296 006060 052706 007776      RESTR: BIS      #TMPEND,SP      ;FORCE THE SP TO LIMIT OF EXISTING CORE.
1297
1298
1299 006064 012703 006700      MOV      #BLIMIT-NUMLIN-NUMLIN,SCAN      ;NOW WE WILL FILL THE KEY AREAS OF THE
1300 006070 012702 000040      MOV      #NUMLIN,TABCNT      ;DISPLAY BUFFER WITH INITIAL CR-LF'S.
1301
1302 006074 012723 005015      SETLP1: MOV     #CRLF,(SCAN)+      ;INSERT A CRLF NOW.
1303 006100 005302      DEC     TABCNT      ;AND LOOP UNTIL DONE.
1304 006102 003374      BGT     SETLP1      ;THUS DISPLAY CORE IS ALMOST CORRECT.
1305
1306
1307 006104 012703 166432      MOV     #SETUP!160000,SCAN      ;NOW WE WILL INITIALIZE CORE FOR THE
1308      ;DISPLAY. PICK UP POINTER TO LIST.
1309
1310 006110 012302      SETLP2: MOV     (SCAN)+,TABCNT      ;GET NUMBER OF ITEMS TO INSERT.
1311 006112 001405      BEQ     SETDUN      ;IF ZERO, WE ARE DONE.
1312 006114 012301      MOV     (SCAN)+,POINTR      ;PICK UP FIRST CORE ADDRESS POINTER.
1313
1314 006116 012321      SETLP3: MOV     (SCAN)+,(POINTR)+      ;MOVE OVER A DATA ITEM NOW.
1315 006120 005302      DEC     TABCNT      ;ALL DONE?
1316 006122 003375      BGT     SETLP3      ;NOPE. MOVE OVER THE NEXT.
1317 006124 000771      BR      SETLP2      ;YES. GET NEXT MAJOR LIST TO INSERT.
1318
1319
1320 006126 012701 006776      SETDUN: MOV     #BLIMIT-2,POINTR      ;ESTABLISH THE BUFFER POINTER NOW.
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
```

.SBTTL VT05 SIMULATOR

1334
 1335
 1336
 1337
 1338
 1339
 1340
 1341
 1342
 1343
 1344
 1345
 1346
 1347
 1348
 1349
 1350
 1351
 1352
 1353
 1354
 1355
 1356
 1357
 1358
 1359
 1360
 1361
 1362
 1363
 1364
 1365
 1366
 1367
 1368
 1369
 1370
 1371
 1372
 1373
 1374
 1375
 1376
 1377
 1378
 1379
 1380
 1381
 1382
 1383
 1384
 1385
 1386
 1387
 1388
 1389

006132 004737 166564
 006136 020027 000177
 006142 002373
 006144 020027 000040
 006150 002020
 006152 010003
 006154 162703 000007
 006160 020327 000007
 006164 103362
 006166 006303
 006170 060307

 006172 000426
 006174 000406
 006176 000411
 006200 000437
 006202 000420
 006204 000424

 006206 012702 177777

 006212 004737 166350
 006216 005202
 006220 000744

 006222 012700 000040
 006226 004737 166350
 006232 005202
 006234 032702 000007
 006240 001370
 006242 000733

 006244 111705

 006246 000405

 006250 005037 172002
 006254 000726

 006256 012705 000040

NXTCHR: JSR PC,GETCHR!160000
 CMP CHAR,#177
 BGE NXTCHR
 CMP CHAR,#40
 BGE NORMAL
 MOV CHAR,SCAN
 SUB #7,SCAN
 CMP SCAN,#7
 BHS NXTCHR
 ASL SCAN
 ADD SCAN,PC

 BR BELL
 BR NORMAL
 BR TAB
 BR LF
 BR VT
 BR FF

 CR: MOV #-1,TABCNT

 NORMAL: JSR PC,INSERT!160000
 INC TABCNT
 BR NXTCHR

 TAB: MOV #40,CHAR
 JSR PC,INSERT!160000
 INC TABCNT
 BIT #7,TABCNT
 BNE TAB
 BR NXTCHR

 VT: MOVB (PC),COUNTR
 BR FFLOOP

 BELL: CLR GT40SR
 BR NXTCHR

 FF: MOV #NUMLIN,COUNTR

VTOS (SCROLLING) PORTION OF THE BOOTSTRAP

;GET A CHARACTER NOW.
 ;IS IT OUT OF RANGE?
 ;YEP. GET ANOTHER ONE.
 ;IS IT A PRINTING CHARACTER?
 ;YES. IT'S A NORMAL PRINTING CHARACTER.
 ;MOVE IT OVER SO WE CAN PLAY WITH IT.
 ;BIAS SO THAT BELL [7] IS ZERO.
 ;IF CHARACTER IS LESS THEN BELL OR
 ;GREATER THEN CR, THEN IGNORE.
 ;IF GOOD, MAKE IT WORD INDEX.
 ;AND GO TO THE CORRECT ROUTINE.

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

 ;RESET TAB POSITION ON A CR, AND
 ;FALL THROUGH TO INSERT THE CHARACTER.

 ;INSERT THE CHARACTER IN THE BUFFER.
 ;UPDATE TAB POSITION NOW.
 ;AND GET NEXT CHARACTER.

 ;ON A TAB, INSERT BLANKS UNTIL THE
 ;NEXT CHARACTER POSITION IS A MULTIPLE
 ;OF 8.
 ;ARE WE DONE YET?
 ;NOPE.
 ;YES.

 ;THIS PUTS THE LOW BYTE OF THE
 ;BRANCH CODE IN COUNTR-SAVE A WORD

 ;RING BELL -WRITE IN GT40SR
 ;AND LOOP BACK

 ;FORM FEED IS DONE BY INSERTING LF'S.

```

1390
1391 006262 012700 000012      FFLOOP: MOV      #12,CHAR      ;MAKE THE CHARACTER A LINEFEED.
1392 006266 004737 166304      JSR      PC,LFSUB!160000      ;DO A LINEFEED.
1393 006272 005305              DEC      COUNTR              ;DONE?
1394 006274 003372              BGT      FFLOOP              ;NOPE. KEEP SENDING THEM.
1395 006276 000715              BR       NXTCHR              ;YES. NOW RETURN. DO NOT FALL THROUGH.
1396
1397
1398 006300 012746 166132      LF:      MOV      #NXTCHR!160000,-(SP)      ;RETURN TO NXTCHR AFTER PROCESSING
1399                                         ;THE LF BY FAKING A JSR.
1400
1401 006304 013703 007012      LFSUB:  MOV      JMPADD,SCAN      ;GET POINTER TO FIRST CHAR ON SCREEN
1402
1403 006310 122300              LFFLOOP: CMPB     (SCAN)+,CHAR      ;AND LOOK FOR A LINEFEED.
1404 006312 001406              BEQ      LFOUND              ;GOT IT. SEARCH HAS ENDED.
1405 006314 020327 007000      CMP      SCAN,#BLIMIT        ;ARE WE AT END OF BUFFER?
1406 006320 103773              BLO      LFFLOOP              ;NOPE. KEEP ON LOOKING.
1407 006322 012703 001000      MOV      #BSTART,SCAN        ;IF AT TOP, RESET TO BOTTOM OF BUFFER
1408 006326 000770              BR       LFFLOOP              ;AND KEEP ON LOOKING.
1409
1410 006330 005203              LFOUND: INC      SCAN          ;WE'VE GOT THE LINE FEED. STOP SHOWING
1411 006332 042703 000001      BIC      #1,SCAN            ;FIRST LINE BY CHANGING THE 'DISJMP'
1412 006336 010337 007012      MOV      SCAN,JMPADD         ;INSTRUCTION TO FIRST CHAR BEYOND LF.
1413 006342 004737 166350      JSR      PC,INSERT!160000    ;INSERT THE LF IN THE BUFFER.
1414 006346 005000              CLR      CHAR                ;AND THEN INSERT ONE NULL CHARACTER BECAUSE
1415                                         ;THE 'DISJMP' ADDRESS MUST BE EVEN, AND
1416                                         ;THIS GUARANTEES WE WILL NOT LOSE A
1417                                         ;A GOOD DATA CHARACTER. WE FALL THROUGH
1418                                         ;TO INSERT THE NULL IN THE BUFFER.
1419
1420
1421 006350 110021              INSERT: MOVB     CHAR,(POINTR)+      ;STICK IN THE CHARACTER NOW.
1422 006352 032701 000001      BIT      #1,POINTR          ;IS NEXT POSITION EVEN OR ODD?
1423 006356 001021              BNE      INSRTX              ;ODD. NO PROBLEMS. SPACE IS ALLOCATED.
1424 006360 020127 007000      CMP      POINTR,#BLIMIT      ;EVEN. ARE WE AT THE END OF THE BUFFER?
1425 006364 103410              BLO      INSRTL              ;NO. JUST MAKE ROOM FOR ANOTHER WORD.
1426 006366 010103              MOV      POINTR,SCAN         ;AT THE END. MOVE THE STUFF TO THE
1427 006370 012701 001000      MOV      #BSTART,POINTR      ;BEGINNING OF THE BUFFER.
1428 006374 004737 166406      JSR      PC,INSRTL!160000    ;CALL THE ROUTINE TO SAVE SPACE.
1429 006400 005023              CLR      (SCAN)+            ;AND CLEAR UP THE INSTRUCTIONS AT THE
1430 006402 005013              CLR      (SCAN)             ;END OF THE BUFFER.
1431 006404 000207              RTS      PC                  ;AND THEN RETURN.
1432
1433 006406 022121              INSRTL: CMP      (POINTR)+,(POINTR)+ ;BYPASS THE 'DISJMP' BY ADDING 4 TO POINTR.
1434 006410 012711 166474      MOV      #HEADER!160000,(POINTR) ;NOW INSERT THE DISJMP INSTRUCTION TO OUR HEADER
1435 006414 012741 160000      MOV      #DISJMP,-(POINTR)    ;AND IT'S ADDRESS (PUT THEM IN BACKWARDS).
1436 006420 005041              CLR      -(POINTR)          ;MAKE AVAILABLE A NEW CHARACTER SPOT.
1437
1438 006422 000207              INSRTX: RTS      PC          ;FINALLY RETURN TO THE CALLER.
1439
1440
1441
1442
1443
1444 006424 012737 001000 172000 GTBUSE: MOV      #BSTART,GT40PC      ;ON A BUS ERROR, WE MERELY RESTART THE GT40 AT
1445

```


COMMUNICATIONS HANDLING ROUTINES

THE DL-11 HANDLER

```

1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511 006516 105737 175610 GETDL: TSTB DL11IS ;CHECK THE HOST INPUT STATUS.
1512 006522 100011 BPL GETDL1 ;HOST DID NOT SEND ANYTHING, YET.
1513 006524 113700 175612 MOVB DL11IB,CHAR ;HOST SENT US A CHARACTER. PROCESS IT.
1514 006530 012737 000007 175610 MOV #7,DL11IS ;REENABLE THE HOST TELECOMMUNICATIONS.
1515 006536 042700 177600 BIC #-200,CHAR ;MAKE CHARACTER JUST SEVEN BITS.
1516 006542 001765 BEQ GETDL ;IF NULL, IGNORE IT.
1517 006544 000207 RTS PC ;ELSE RETURN NOW.
1518
1519 006546 105737 177560 GETDL1: TSTB KBDIS ;DID USER TYPE A CHARACTER?
1520 006552 100361 BPL GETDL ;NO. GO BACK AND CHECK HOST MACHINE.
1521 006554 113737 177562 175616 MOVB KBDIB,DL110B ;MOVE THE CHARACTER TO THE HOST.
1522 006562 000755 BR GETDL ;AND CHECK AGAIN FOR INPUT.
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533 006564 004737 166516 GETCHR: JSR PC,GETDL!160000 ;GET A CHARACTER FROM THE HOST NOW.
1534 006570 020027 000175 CMP CHAR,#ALTMOD ;IS IT AN 'ALTMODE'?
1535 006574 001025 BNE GETEXT ;NO. EXIT NOW.
1536
1537 006576 004737 166516 JSR PC,GETDL!160000 ;YES. GET ANOTHER ONE NOW.
1538 006602 020027 000114 CMP CHAR,#'L' ;IS IT AN 'L'?
1539 006606 001501 BEQ LOADER ;YES. START LOADING NOW.
1540 006610 020027 000122 CMP CHAR,#'R' ;IS IT AN 'R'?
1541 006614 001015 BNE GETEXT ;NO. IGNORE THE ALTMODE AND JUST RETURN THE CHAR
1542
1543 006616 012737 173000 007010 MOV #DSTOP,JMPADD-2 ;YES. RESET. STOP DISPLAY BY INSERTING A 'DSTOP
1544 006624 000137 166060 PRESTR: JMP RESTR!160000 ;INSTRUCTION IN THE BUFFER, AND RESTART.
1545
1546
1547
1548
1549
1550

```

THE 'GET A SIX BIT CHARACTER' ROUTINE

```

1551 : -----
1552 :
1553 :
1554 :
1555 006630 004737 166564 GETSIX: JSR PC,GETCHR!160000 ;GET A CHARACTER NOW.
1556 006634 020027 000040 CMP CHAR,#40 ;IS IT A LEGAL PRINTING CHARACTER?
1557 006640 002517 BLT L.BAD ;NOPE. ABORT
1558 006642 020027 000137 CMP CHAR,#137 ;IT'S BIG ENOUGH. IS IT TOO BIG?
1559 006646 003114 BGT L.BAD ;YEP. ABORT.
1560 :
1561 006650 000207 GETEXT: RTS PC ;RETURN TO THE CALLER.
1562 :
1563 :
1564 : THIS OUTPUTS TWO CHARACTERS VIA A
1565 : JSR SCAN,OUTLIT
1566 : 'TWO CHARACTERS'
1567 :
1568 006652 112337 175616 OUTLIT: MOVB (SCAN)+,DL110B
1569 006656 112337 175616 MOVB (SCAN)+,DL110B ;DOUBLE BUFFERED
1570 006662 000203 RTS SCAN ;RETURN
1571 :
1572 :
1573 :
1574 :
1575 :
1576 :
1577 :
1578 : THE 'GET AN EIGHT BIT CHARACTER' ROUTINE
1579 : -----
1580 :
1581 :
1582 :
1583 : THIS ROUTINE DIFFERS FROM THE PREVIOUS ROUTINES
1584 : IN THAT IT WILL TAKE SIX BIT CHARACTERS AND ASSEMBLE
1585 : THEM FOR THE LOADER TO USE. NOTE THAT FROM THIS POINT
1586 : ON WE WILL SWITCH TO THE LOADER DEFINITIONS OF THE
1587 : REGISTERS. THUS THE CHARACTER IS RETURNED IN
1588 : REGISTER 'L.BYT' RATHER THAN CHAR (THOUGH THEY ARE
1589 : PHYSICALLY THE SAME).
1590 :
1591 :
1592 :
1593 006664 004737 166630 GET8: JSR PC,GETSIX!160000 ;GET A SIXBIT CHARACTER.
1594 006670 010046 MOV L.BYT,-(SP) ;SAVE IT ON THE STACK.
1595 006672 005723 TST (INDEX)+ ;UPDATE INDEX TO NEXT ITEM (ALL ARE *2)
1596 006674 000163 166676 JMP GET8TB-2!160000(INDEX) ;AND DISPATCH ACCORDING TO THE INDEX.
1597 :
1598 006700 000404 GET8TB: BR GET81 ;INDEX=2: ASSEMBLE FIRST CHAR
1599 006702 000416 BR GET82 ;INDEX=4: ASSEMBLE SECOND CHAR
1600 006704 000432 BR GET83 ;INDEX=6: ASSEMBLE THIRD AND LAST CHAR
1601 : ;INDEX=8: RESET INDEX TO 0 [2] AND RETRY.
1602 :
1603 :
1604 006706 012703 000002 GET84: MOV #2,INDEX ;THE FOURTH INDEX IS THE SAME AS THE FIRST
1605 : ;INDEX. JUST RESET IT AND FALL THROUGH.
1606 :

```

```

1607
1608 006712 004737 166630 GET81: JSR PC,GETSIX!160000 ;GET ANOTHER CHARACTER NOW.
1609 006716 010004 MOV L.BYT,HOLD ;AND PRESERVE IT FOR NEXT TIME THROUGH.
1610 006720 006300 ASL L.BYT ;NOW THROW AWAY LEFT MOST BITS OF
1611 006722 006300 ASL L.BYT ;THE 8 BIT CHARACTER. NOW MERGE IN
1612 006724 106300 ASLB L.BYT ;THE LEFT TWO BITS OF THE
1613 006726 106116 ROLB (SP) ;NEW SIX BIT CHARACTER WITH THE SIX
1614 006730 106300 ASLB L.BYT ;BITS FROM THE CHARACTER ON THE
1615 006732 106116 ROLB (SP) ;STACK. 1ST CHARACTER IS NOW ASSEMBLED,
1616 006734 012600 MOV (SP)+,L.BYT ;SO WE'LL RETURN IT TO THE USER.
1617 006736 000207 RTS PC ;AND THEN WE SHALL RETURN TO HIM.
1618
1619
1620 006740 006300 GET82: ASI L.BYT ;THE SECOND CHARACTER IS CREATED FROM
1621 006742 006300 ASL L.BYT ;THE 4 RIGHT BITS OF THE PREVIOUS CHARACTER
1622 006744 106300 ASLB L.BYT ;AND THE FOUR MIDDLE BITS OF THE PRESENT
1623 006746 106104 ROLB HOLD ;8 BIT CHARACTER.
1624 006750 106300 ASLB L.BYT ;WE WILL CREATE THE NEW 8 BIT
1625 006752 106104 ROLB HOLD ;IN THIS REGISTER, SINCE IT
1626 006754 106300 ASLB L.BYT ;MORE CONVIENT. WE WILL MOVE OVER THE
1627 006756 106104 ROLB HOLD ;ANSWER AT THE END.
1628 006760 106300 ASLB L.BYT ;ONE MORE TO GO
1629 006762 106104 ROLB HOLD ;DONE.
1630 006764 010400 MOV HOLD,L.BYT ;BRING OVER THE VALUE.
1631 006766 012604 MOV (SP)+,HOLD ;AND REMEMBER THE LAST CHARACTER WE RECEIVED.
1632 006770 000207 RTS PC ;AND RETURN TO THE CALLER.
1633
1634
1635 006772 006100 GET83: ROL L.BYT ;FINAL CHARACTER IS EASY. JUST A
1636 006774 106100 ROLB L.BYT ;SIMPLE MERGER OF LEFT TWO BITS OF
1637 006776 006004 ROR HOLD ;PREVIOUS VALUE WITH RIGHT SIX BITS
1638 007000 106000 RORB L.BYT ;OF LAST (4TH) CHARACTER RECEIVED.
1639 007002 006004 ROR HOLD
1640 007004 106000 RORB L.BYT ;AND WE ARE DONE.
1641 007006 005726 TST (SP)+ ;FINALLY THROW AWAY STACK.
1642 007010 000207 RTS PC ;AND RETURN TO THE CALLER.
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655 .SBITL THE LOADER

```

```

1657
1658
1659
1660          :          THE LOADER
1661          :          -----
1662
1663
1664
1665
1666 007012 012737 173000 007010 LOADER: MOV  #DISTOP,JMPADD-2      ;STOP THE GT40 BY INSERTING A 'DISTOP' IN THE LI
1667
1668 007020 005003          CLR      INDEX          ;RESET THE 8 BIT ASSEMBLER TO THE FIRST CHAR
1669
1670
1671 007022 005005          L.LD2: CLR      L.CKSM          ;CLEAR THE CHECKSUM
1672 007024 004737 167114      JSR      PC,L.PTR!160000      ;GET A BYTE NOW.
1673 007030 105300          DECB    L.BYT          ;IS IT A ONE (HEADER)?
1674 007032 001373          BNE     L.LD2          ;NO. WAIT FOR THE ONE.
1675
1676 007034 004737 167114      JSR      PC,L.PTR!160000      ;YES. SKIP OVER THE NEXT CHARACTER NOW.
1677
1678 007040 004737 167126      JSR      PC,L.GWRD!160000      ;ASSEMBLE A WORD NOW.
1679 007044 010002          MOV     L.BYT,L.BC          ;MOVE OVER TO THE COUNTER.
1680 007046 162702 000004      SUB     #4,L.BC          ;REDUCE TO ACTUAL DATA COUNT.
1681 007052 022702 000002      CMP     #2,L.BC          ;ANY DATA AT ALL?
1682 007056 001433          BEQ     L.JMP          ;NO. MUST BE END
1683 007060 004737 167126      JSR      PC,L.GWRD!160000      ;YES. ASSEMBLE A DATA WORD NOW.
1684 007064 010001          MOV     L.BYT,L.ADR          ;AND THIS MUST BE THE FIRST ADDRESS.
1685
1686
1687 007066 004737 167114      L.LD3: JSR      PC,L.PTR!160000      ;GET A BYTE OF DATA NOW.
1688 007072 002006          BGE     L.LD4          ;ALL DONE?
1689 007074 105705          TSTB   L.CKSM          ;YEP. COUNTER IS MINUS. CHECK CHECKSUM.
1690 007076 001751          BEQ     L.LD2          ;CHECKSUM GOOD. GET NEXT COMMAND.
1691
1692
1693 007100 004337 166652      L.BAD: JSR      SCAN,OUTLIT.160000      ;BAD LOAD INFORM HOST
1694 007104 175 102          .BYTE  ALTMOD,'B          ;SEND ALTMODE B
1695 007106 000646          BR      PRESTR          ;AND RESTART THE DISPLAY.
1696
1697
1698 007110 110021          L.LD4: MOVB   L.BYT,(L.ADR)+      ;INSERT BYTE INTO MEMORY.
1699 007112 000765          BR      L.LD3          ;AND GET THE NEXT BYTE.
1700
1701
1702
1703 007114 004737 166664      L.PTR: JSR      PC,GET8!160000      ;ASSEMBLE AN 8 BIT CHARACTER NOW.
1704 007120 060005          ADD     L.BYT,L.CKSM      ;UPDATE THE CHECKSUM NOW.
1705 007122 005302          DEC     L.BC          ;DECREMENT THE CHARACTER COUNTER.
1706 007124 000207          RTS     PC          ;AND RETURN TO THE CALLER NOW.
1707
1708
1709
1710 007126 004737 167114      L.GWRD: JSR      PC,L.PTR.160000      ;ASSEMBLE A WORD. FIRST GET A CHARACTER
1711 007132 010046          MOV     L.BYT,-(SP)        ;AND SAVE IT.
1712 007134 004737 167114      JSR      PC,L.PTR!160000      ;AND THEN GET ANOTHER ONE.

```

```

1713 007140 000300          SWAB  L.BYT          :AND THEN REASSEMBLE THE MESS.
1714 007142 052600          BIS   (SP)+,L.BYT      :WITH THE FEARSOME POWER OF THE 11.
1715 007144 000207          RTS   PC              :AND RETURN TO THE CALLER.
1716
1717
1718
1719
1720 007146 004737 167126    L.JMP: JSR   PC,L.GWRD!160000      ;ALL DONE WITH THE LOAD. ASSEMBLE
1721 007152 010046          MOV   L.BYT,-(SP)          :THE STARTING ADDRESS NOW.
1722 007154 004737 167114    JSR   PC,L.PTR!160000      :AND DON'T FORGET TO CHECKSUM IT.
1723 007160 105705          TSTB L.CKSM
1724 007162 001346          BNE  L.BAD              :A BAD CHECKSUM. ALL IS EVIL.
1725
1726 007164 004337 166652    JSR   SCAN,OUTLIT.160000    ;GOOD CHKSUM,INFORM HOST
1727 007170      175      107    .BYTE ALTMOD,'G          :WITH ALTMOD G
1728
1729 007172 032716 000001    BIT   #1,(SP)            :DO WE WANT TO START EXECUTION?
1730 007176 001401          BEQ  L.JMP1             :YES. AWAY WE GO.
1731
1732 007200 000000    L.HALT: HALT              :IF NOT, HALT.
1733
1734 007202 000136    L.JMP: JMP   @(SP)+        :IF GO, THEN GO ALREADY. WHEEEE:
1735
1736
1737
1738          .SBITL  THE SELF TEST
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750

```


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

1752			
1753			
1754			
1755			
1756	100000	CHAR=100000	
1757	104000	SHORTV=104000	
1758	110000	LONGV=110000	
1759	114000	POINT=114000	
1760	120000	GRAPHX=120000	
1761	124000	GRAPHY=124000	
1762	130000	RELATV=130000	
1763			
1764	002000	INT0=2000	
1765	002200	INT1=2200	
1766	002400	INT2=2400	
1767	002600	INT3=2600	
1768	003000	INT4=3000	
1769	003200	INT5=3200	
1770	003400	INT6=3400	
1771	003600	INT7=3600	:BRIGHTEST
1772			
1773	000100	LPOFF=100	
1774	000140	LPON=140	
1775	000020	BLKOFF=20	
1776	000030	BLKON=30	
1777			
1778	000004	LINE0=4	
1779	000005	LINE1=5	
1780	000006	LINE2=6	
1781	000007	LINE3=7	
1782			
1783	160000	DJMP=160000	
1784	164000	DNOP=164000	
1785	170000	STATSA=170000	
1786	173400	DSTOP=173400	:STOP INTERRUPT
1787			
1788	000300	LPLITE=300	
1789	000200	LPDARK=200	
1790	000040	ITAL0=40	:ITALICS OFF
1791	000060	ITAL1=60	:ON
1792	000004	SYNON=4	:SYNC ON
1793			
1794			
1795	174000	STATSB=174000	
1796			
1797	000100	INCR=100	:LOAD GRAPH INCR
1798	040000	INTX=40000	:INTENSIFY BIT
1799	001777	MAXX=1777	:BIGGEST X VECTOR
1800	001377	MAXY=1377	:BIGGEST Y VECTOR
1801	020000	MINUSX=20000	:THE MINUS BIT
1802	020000	MINUSY=MINUSX	
1803	017600	MAXSX=17600	:BIGGEST X IN SHORTVEC
1804	000077	MAXSY=77	:Y IN
1805	000100	MINSUY=100	:MINUS BIT FOR Y IN SHORTVEC
1806			
1807			

1808	007204	012737	167214	172000	MOV #FILE0:160000,GT40PC	:START THE GT40
1809	007212	000001			WAIT	:AND WAIT
1810						
1811	007214	114020			FILE0: POINT BLKOFF	:POINT--INVISIBLE
1812	007216	000000			0	
1813	007220	001377			MAXY	
1814						
1815	007222	112004			LONGV.INT0.LINE0	:DRAW TOP LINE
1816	007224	041777			INTX.MAXX	
1817	007226	000000			0	
1818						
1819	007230	112405			LONGV INT2.LINE1	
1820	007232	040000			INTX	:DRAW LINE TO RIGHT
1821	007234	021377			MINUSX.MAXY	
1822						
1823	007236	113006			LONGV.INT4.LINE2	
1824	007240	061777			INTX MINUSX!MAXX	:DRAW BOTTOM LINE
1825	007242	000000			0	
1826						
1827	007244	113407			LONGV.INT6.LINE3	
1828	007246	040000			INTX	
1829	007250	001377			MAXY	:DRAW LINE TO LEFT
1830						
1831	007252	114000			POINT	
1832	007254	000400			400	
1833	007256	000500			500	
1834	007260	106200			SHORTV.INT1	
1835	007262	057677			57677	:+X+Y
1836	007264	106600			SHORTV!INT3	
1837	007266	077677			77677	:+X-Y
1838	007270	107200			SHORTV!INT5	
1839	007272	077777			77777	:-X-Y
1840	007274	107600			SHORTV!INT7	
1841	007276	057777			57777	:-X+Y
1842						
1843	007300	114000			POINT	
1844	007302	001400			1400	
1845	007304	000500			500	
1846	007306	133030			RELATV!INT4!BLKON	
1847	007310	057677			57677	:+X+Y
1848	007312	077677			77677	:+X-Y
1849	007314	077777			77777	:-X-Y
1850	007316	057777			57777	:-X+Y
1851						
1852	007320	114000			POINT	
1853	007322	000400			400	
1854	007324	000100			100	
1855	007326	174120			STATSB.INCR+20	:TRY GRAPH MODES
1856	007330	114000			POINT	
1857	007332	001000			1000	
1858	007334	000200			200	
1859						
1860	007336	120000			GRAPHX	
1861	007340	001010			1010	
1862	007342	001020			1020	
1863	007344	001030			1030	

SCROLLING ROM BOOTSTRAP FOR THE GT40
CDGTED.P11 31-AUG-79 11:13

MACV11 30G(1063) 31-AUG-79 13:00 PAGE 16-15
THE SELF TEST

SEQ 0034

1864	007346	001040	1040
1865	007350	001050	1050
1866			
1867	007352	114000	POINT
1868	007354	001000	1000
1869	007356	001200	1200
1870			
1871	007360	124000	GRAPHY
1872	007362	001020	1020
1873	007364	001030	1030
1874	007366	001040	1040
1875	007370	001050	1050
1876	007372	001060	1060
1877			
1878	007374	160000	DJMP
1879	007376	167214	FILE0:160000
1880			
1881			.SBTTL PAPER TAPE BOOT

```

1883
1884
1885
1886          177550
1887          177560
1888
1889
1890 007400 012701 160000
1891 007404 012702 000004
1892 007410 012703 167500
1893 007414 010712
1894 007416 012706 000024
1895 007422 014304
1896 007424 005714
1897 007426 100775
1898 007430 010712
1899 007432 012706 000024
1900 007436 010441
1901
1902 007440 040601
1903 007442 C10111
1904 007444 011102
1905 007446 005214
1906 007450 105714
1907 007452 100376
1908 007454 116412 000002
1909 007460 005211
1910 007462 120227 000375
1911 007466 001366
1912 007470 105222
1913 007472 000142
1914
1915
1916
1917 007474 177560
1918 007476 177550
1919
1920

: 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

```

```

1923      ;
1924      ; CASSETTE BOOT
1925      ;
1926      177500      TACS=177500      ;TA-11 CONTROL AND STATUS REGISTER
1927      ;          .=ORIGIN+1500
1928      007500  012700  177500      TABOOT: MOV      #TACS,R0
1929      007504  005010      CLR      (R0)      ;SELECT UNIT #0
1930      007506  010701      RES:     MOV      PC,R1      ;USE FOR PIC
1931      007510  062701  000052      ADD      #TABLE-.,R1      ;R1 HOLDS ADDR. OF COMMAND TABLE
1932      007514  012702  000375      MOV      #375,R2      ;MEMORY PTR. AND DATA FLAG
1933      007520  112103      MOVB     (R1)+,R3      ;TEST BITS
1934      ;
1935      007522  112110      LOOP1:  MOVB     (R1)+,(R0)      ;COMMAND FROM TABLE TO TACS
1936      007524  100413      BMI     DONE      ;WHEN COMMAND CODE NEG., QUIT
1937      007526  130310      LOOP2:  BITB     R3,(R0)      ;TEST READY AND T-REQ BITS IN TACS
1938      007530  001776      BEQ     LOOP2      ;LOOP 'TIL SOMETHING COMES UP
1939      007532  105202      INCB     R2      ;ADVANCE MEMORY POINTER
1940      007534  100772      BMI     LOOP1      ;IF MINUS, TRY NEXT COMMAND
1941      007536  116012  000002      MOVB     2(R0),(R2)      ;READ DATA INTO MEMORY
1942      007542  120337  000000      CMPB     R3,@#0      ;FIRST BYTE READ SHOULD BE '240'
1943      007546  001767      BEQ     LOOP2      ;IF O.K., GO READ ANOTHER BYTE
1944      007550  000000      STOP:   HALT      ;HALT ON ERROR
1945      007552  000755      BR      RES      ;RESTART ON CONTINUE
1946      ;
1947      007554  005710      DONE:   TST      (R0)      ;CHECK FOR ERROR
1948      007556  100774      BMI     STOP      ;HALT ON ERROR
1949      007560  005007      CLR     PC      ;= 'JMP @#0'
1950      ;
1951      007562  017640      TABLE: .WORD    17640      ;.BYTE 240: READY+T-REQ.
1952      ;          ;.BYTE 37: ILBS+READY+GO
1953      007564  002415      ;.WORD    2415      ;.BYTE 15: SFB+GO
1954      ;          ;.BYTE 5: READ+GO
1955      007566  112024      ;.WORD    112024      ;.BYTE 24: READ+ILBS
1956      ;          ;.BYTE 224: READ+ILBS+E.O.TABLE
1957      007570  000000  000000      ;.WORD    0,0      ;THESE ARE FILLER WORDS
1958      007574  167500      ;.WORD    TABOOT.160000      ;POWER UP VECTOR AND PRIORITY
1959      007576  000340      ;.WORD    340      ;
1960      ;
1961      ;
1962      ;.SBTTL  MR11-DB BOOT

```

```

1964 ;MR11-DB BULK STORAGE PROGRAM LOADER LISTING
1965
1966 ; .=ORIGIN+1600 ;KEEP TRACK OF ORIGIN
1967
1968 RF11: MOV PC,R2 ;FIXED HEAD DISK (256 KW)
1969 BR OTHER
1970 177462
1971 007606 000005
1972
1973 RK11: MOV PC,R2 ;MOVING HEAD DISK (CARTRIDGE)
1974 BR OTHER
1975 007614 177406
1976 007616 000005
1977
1978
1979 TC11: MOV PC,R2
1980 007622 000417 BR TAPES
1981 007624 177344 177344 ;ADDRESS OF WORD COUNT
1982 007626 000005 5 ;LAST COMMAND
1983 007630 004003 4003 ;FIRST COMMAND
1984 007632 100000 100000 ;DONE MASK
1985 007634 024000 24000 ;ERROR MASK
1986
1987
1988 TM11: MOV PC,R2
1989 007640 000410 BR TAPES
1990 007642 172524 172524 ;ADDRESS OF BYTE COUNT
1991 007644 060003 60003 ;LAST COMMAND
1992 007646 060011 60011 ;FIRST COMMAND
1993 007650 000200 200 ;DONE MASK
1994 007652 100000 100000 ;ERROR MASK
1995
1996
1997 RP11: MOV PC,R2 ;MOVING HEAD DISK (PACK)
1998 007656 000423 BR OTHER
1999 007660 176716 176716
2000
2001
2002 TAPES: RESET
2003 007664 010200 MOV R2,R0 ;GET THE ADDRESS OF THE BRANCH
2004 007666 005720 TST (0)+ ;R0 TO POINT AT LAST COMMAND
2005 007670 012001 MOV (0)+,R1 ;GET THE WORD COUNT ADDRESS
2006 007672 005311 DEC (1) ;SET UP FOR ADVANCE 1 RECORD
2007 007674 005720 TST (0)+ ;MOVE R0 TO FIRST COMMAND
2008 007676 012041 MOV (0)+,-(1) ;COMMAND WORD TO COMMAND REG.
2009 007700 031011 BIT (0),(1) ;LOOK FOR DONE INDICATORS
2010 007702 001776 BEQ .-2 ;NONE SET, TRY AGAIN
2011 007704 005720 TST (0)+ ;DONE FIRST COMMAND, CHECK FOR ERROR
2012 007706 031041 BIT (0),-(1) ;LOOK FOR SET ERROR BITS
2013 007710 001406 BEQ OTHER ;NO ERRORS - TRY THE READ
2014 007712 000112 AGAIN: JMP (2) ;RERUN FOR ERRORS
2015
2016
2017 RFVEC: RF11.160000 ;RF11 POWER JP VECTOR
2018 007716 000340 340
2019

```

```
2020 007720 010702          RC11:  MOV PC,R2          ;FIXED HEAD DISK (64KW)
2021 007722 000401          BR OTHER
2022 007724 177450          177450          ;ADRS OF WORD COUNT (COMMAND+2)
2023                                     ;COMMAND WORD (5) IS THE RESET
2024
2025 007726 000005          OTHER:  RESET
2026 007730 010200          MOV R2,R0          ;R0 TO POINT AT WORD COUNT ADRS
2027 007732 005720          TST (0)+          ;POINT TO ADDRESS
2028 007734 012001          MOV (0)+,R1       ;WORD COUNT ADDRESS TO R1
2029 007736 012711 177000  MOV #-1000,(1)    ;LOAD WORD COUNT
2030 007742 011041          MOV (0),-(1)     ;COMMAND TO COMMAND REGISTER
2031 007744 032711 100200  BIT #100200,(1)  ;CHECK FOR ERROR OR DONE
2032 007750 001775          BEQ .-4          ;IF NEITHER, KEEP LOOKING
2033 007752 100757          BMI AGAIN        ;ERROR, TRY AGAIN
2034 007754 005007          CLR PC
2035
2036 007756 000000          0                ;FILLER
2037 007760 167610          RKVEC:  RK11!160000 ;RK POWER UP VECTOR
2038 007762 000340          340
2039 007764 167720          RCVEC:  RC11!160000 ;RC POWER UP VECTOR
2040 007766 000340          340
2041 007770 167654          RPVEC:  RP11.160000 ;RP POWER UP VECTOR
2042 007772 000340          340
2043 007774 167620          TCVEC:  TC11!160000 ;TC11 POWER UP VECTOR
2044 007776 000340          340
```

```

2046 ;ROUTINE TO LOAD EXCESS CORE WITH WORSE CASE MEMORY TEST
2047
2048 010000 013700 001312 CORTST: MOV SIZE,R0 ;GET LAST FREE CORE ADDRESS
2049 010004 012701 017000 MOV #BUFFER,R1 ;GET END OF PROGRAM
2050 010010 020001 CMP R0,R1 ;TEST FOR EQUAL
2051 010012 103410 BLO XMRTS ;BRANCH IF NO ROOM
2052 010014 012702 010050 XMLOP1: MOV #MEMTST,R2 ;MOVE CODE BETWEEN
2053 010020 012221 XMLOP2: MOV (R2)+,(R1)+ ;MEMTST AND MEMEND TILL
2054 010022 022702 010120 CMP #MEMEND,R2 ;CORE IS FULL
2055 010026 001374 BNE XMLOP2 ;TEST FOR MORE ROOM
2056 010030 020100 CMP R1,R0
2057 010032 101770 BLOS XMLOP1 ;SETUP RTS PC
2058 010034 012721 000207 XMRTS: MOV #207,(R1)+
2059 010040 005021 CLR (R1)+
2060 010042 005021 CLR (R1)+
2061 010044 000207 RTS PC
2062
2063 010046 151456 ROTVAL: 151456
2064 .DSABL AMA
2065
2066 ;THIS IS THE BACKGROUND TASK WHICH WILL BE LOADED THRU
2067 ; THE REMAINDER OF MEMORY
2068
2069 010050 000277 MEMTST: SCC ;SET CARRY BIT
2070 010052 012727 123456 MOV #123456,(PC)+ ;MEMDAT CONTAINS
2071 010056 123456 MEMDAT: 123456
2072 010060 106067 177773 RORB MEMDAT+1 ;ROTATE LEFT BYTE OF MEMDAT
2073 010064 103401 BCS .+4 ;C BIT WAS NOT SET
2074 010066 000000 HALT ;C BIT WAS NOT SET
2075 010070 102001 BVC .+4 ;V BIT WAS SET
2076 010072 000000 HALT ;V BIT WAS SET
2077 010074 022767 151456 177754 CMP #151456,MEMDAT ;CHECK HERE FOR CORRECT ROTATE
2078 010102 001401 BEQ .+4 ;CHECK HERE FOR CORRECT ROTATE
2079 010104 000000 HALT ;ROTATE FAILED
2080 010106 026737 177744 010046 CMP MEMDAT,#ROTVAL ;CHECK AGAIN REFERENCE LOW MEMORY
2081 010114 001401 BEQ .+4 ;CHECK AGAIN REFERENCE LOW MEMORY
2082 010116 000000 HALT ;REF. TO LOW MEMORY FAILED
2083 010120 000000 MEMEND: 0
2084 010122 000000 0
  
```


2086
2087
2088
2089 010124 000001 000137
2090 010130 000002 167636
2091 010134 000003 000000
2092 010140 000327 000137
2093 010144 000330 167620
2094 010150 000710 105737
2095 010154 000711 175614
2096 010160 000712 100375
2097 01016 000713 112337
2098 010170 000714 175616
2099 010174 000715 000203
2100 010200 000716 000000
2101 010204 000717 005037
2102 010210 000720 177776
2103 010214 000721 012737
2104 010220 000722 010007
2105 010224 000723 175610
2106 010230 000724 000137
2107 010234 000725 166010
2108 010240 000000

.SBTTL TABLE OF PATCHES TO VERSION 2

TABLEC: .WORD 1,137
.WORD 2,167636
.WORD 3,0
.WORD 215,137
.WORD 216,167620
.WORD 456,105737
.WORD 457,175614
.WORD 458,100375
.WORD 459,112337
.WORD 460,175616
.WORD 461,203
.WORD 462,0
.WORD 463,5037
.WORD 464,177776
.WORD 465,12737
.WORD 466,10007
.WORD 467,175610
.WORD 468,137
.WORD 469,166010
TABLEX: 0

2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165

```
.SBTTL ROM VERSION 1 VALUES
.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
```

```

2166
2167
2168
2169      036000      COREND=36000      ;FIRST LOCATION OF NON-CORE.
2170      166000      ROMORG=166000     ;WHERE THE ROM PROGRAM SHOULD GO.
2171
2172      000000      STARTX=0      ;WHERE TO START DISPLAYING THE X POSITIONS.
2173      001360      STARTY=1360     ;WHERE TO START DISPLAYING THE Y.
2174
2175
2176      022000      VT40PC=172000-150000 ;VT40 PROGRAM COUNTER.
2177      027560      KBDIS=27560      ;TTY INPUT STATUS.
2178      025614      P10OS=25614     ;PDP-10 OUTPUT STATUS.
2179      025610      P10IS=25610     ;PDP-10 INPUT STATUS.
2180
2181      027562      KBDIB=KBDIS+2    ;TTY INPUT BUFFER.
2182      025612      P10IB=P10IS+2   ;PDP-10 INPUT CHARACTER.
2183      025616      P10OB=P10OS+2   ;PDP-10 OUTPUT BUFFER.
2184
2185
2186      045776      P10OC=COREND-2+10000 ;CHARACTER TO BE SENT TO THE PDP-10
2187      045772      P10IC=P10OC-4    ;INPUT CHARACTER FROM 10 PLUS ONE SAVE CHARACTER
2188      015770      STKSRT=P10IC-2-30000 ;FIRST LOCATION OF STACK.
2189
2190
2191      160000      JMPDIS=160000     ;THE VT-40 DISPLAY JUMP INSTRUCTION.
2192
2193
2194      000024      PWRFAL=24        ;POWER FAIL RESTART LOCATION.
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206      016000      ;      .=16000
2207      ;      ;      .=ROMORG      ;SET THE ORIGIN NOW...!.
2208
2209
2210
2211
2212
2213
2214
2215      016000      012705      000026      STARTA: MOV      #PWRFAL+2,SCR1 ;PICK UP POINTER TO P.F. STATUS.
2216      016004      005015      CLR      @SCR1      ;CLEAR IT OUT TO BE SURE.
2217      016006      010745      MOV      PC,-(SCR1) ;SET UP THE RESTART LOCATION.
2218
2219      016010      000005      RESET      ;RESET THE BUS.
2220
2221      016012      012767      000007      007570      MOV      #7,P10IS ;INITIALIZE PDP-10 INPUT
  
```

```

2222 016020 012767 000001 011532      MOV      #1,KBDIS      ;INITIALIZE TTY INPUT.
2223 016026 012767 000201 007560      MOV      #201,P100S   ;INITIALIZE PDP-10 OUTPUT.
2224
2225
2226
2227 016034 012706 015770      RSTRT:  MOV      #STKSRT,SP      ;SET UP THE STACK NOW.
2228 016040 005001              CLR      LADR              ;CLEAR ADDRESS POINTER.
2229 016042 012702 160000      MOV      #JMPDIS,INP2     ;PLACE A DISPLAY JUMP INSTRUCTION IN A REGISTER.
2230 016046 010221              MOV      INP2,(LADR)+     ;MOVE IT TO LOCATION 0.
2231 016050 012711 166756      MOV      #DISPRG+150000,(LADR) ;MOVE ADDRESS POINTER INTO 2.
2232 016054 012701 000030      MOV      #PWRFAL+4,LADR   ;SET UP WHERE WE WILL STORE CHARACTERS.
2233 016060 005000              CLR      RET1             ;PREPARE TO INSERT A ZERO CHARACTER.
2234 016062 004767 000022      JSR      PC,DOCHAR        ;INSERT IT NOW.
2235 016066 005067 003706      CLR      VT40PC          ;CLEAR THE DISPLAY PROGRAM COUNTER AND START.
2236
2237 016072 004767 000210      MAJOR:  JSR      PC,GTCHR     ;GT A CHARACTER NOW.
2238 016076 000240              NOP
2239 016100 000240              NOP
2240 016102 000240              NOP
2241 016104 012746 166072      MOV      #MAJOR+150000,-(SP) ;INSERT IN DISPLAY BUFFER NOW.
2242
2243 016110 010105      DOCHAR: MOV      LADR,SCR1      ;GT CURRENT BUUFER POSITION NOW.
2244 016112 022525      CMP      (SCR1)+,(SCR1)+  ;BYPASS CURRENT DISPLAY JUMP.
2245 016114 005025      CLR      (SCR1)+         ;CLEAR FUTURE ADDRESS FOR JUMP.
2246 016116 010225      MOV      INP2,(SCR1)+    ;STICK IN TEMPORARY JUMP WHILE WE REPLACE CURREN
2247 016120 005015      CLR      (SCR1)          ;A DISPLAY JUMP TO ZERO.
2248 016122 005011      CLR      (LADR)          ;NOW REPLACE CURRENT DISPLAY JUMP BY THE CHARACT
2249 016124 050021      BIS      RET1,(LADR)+    ;IT'S DONE THIS WAY TO WASTE 2 CYCLES.
2250 016126 010211      MOV      INP2,(LADR)     ;TO AVOID TIMING PROBLEMS WITH THE VT40.
2251 016130 000207      RTS      PC              ;AND FINALLY RETURN.
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268 016132 004767 000124      GT8:    JSR      PC,GTSIX     ;GT SIX BITS NOW.
2269 016136 010046      MOV      RET1,-(SP)      ;SAVE THE CHARACTER NOW.
2270 016140 000401      BR      GTP84            ;BYPASS THE 8'ER
2271 016142 005002      GT84:   CLR      INP2     ;RESET THE MAGIC REGISTER NOW.
2272 016144 005722      GTP84:  TST      (INP2)+    ;INCREMENT WHERE TO GO.
2273 016146 066207 166250      ADD      GT8TB+150000(INP2),PC ;UPDATE PC NOW.
2274
2275              016152      GT8P=.
2276
2277 016152 004767 000104      GT81:   JSR      PC,GTSIX     ;GT A CHARACTER NOW.

```

```

2278 016156 010004      MOV      RET1,WORK2      ;SAVE FOR A SECOND.
2279 016160 006300      ASL      RET1
2280 016162 006300      ASL      RET1      ;SHIFT TO LEFT OF BYTE
2281 016164 106300      ASLB     RET1
2282 016166 106116      ROLB     @SP      ;PACK THEM IN.
2283 016170 106300      ASLB     RET1
2284 016172 106116      ROLB     @SP      ;A GOOD 8 BIT THING.
2285 016174 012600      MOV      (SP)+,RET1   ;POP AND RETURN NOW.
2286 016176 000207      RTS      PC
2287
2288 016200 006300      GT82:   ASL      RET1      ;WORST CASE. SHIFT 4
2289 016202 006300      ASL      RET1
2290 016204 106300      ASLB     RET1
2291 016206 106104      ROLB     WORK2
2292 016210 106300      ASLB     RET1
2293 016212 106104      ROLB     WORK2
2294 016214 106300      ASLB     RET1
2295 016216 106104      ROLB     WORK2
2296 016220 106300      ASLB     RET1
2297 016222 106104      ROLB     WORK2
2298 016224 010400      MOV      WORK2,RET1
2299 016226 012604      MOV      (SP)+,WORK2
2300 016230 000207      RTS      PC
2301
2302 016232 006100      GT83:   ROL      RET1
2303 016234 006100      ROL      RET1
2304 016236 006004      ROR      WORK2
2305 016240 106000      RORB     RET1
2306 016242 006004      ROR      WORK2
2307 016244 106000      RORB     RET1
2308 016246 005726      TST      (SP)+
2309 016250 000207      RTS      PC
2310
2311          016250      GT8TB  -      .-2      ;PUSH ZERO CONDITION BACK INTO NEVER-NEVER LAND.
2312
2313 016252 000000      .WORD   GT81-GT8P
2314 016254 000026      .WORD   GT82-GT8P
2315 016256 000060      .WORD   GT83-GT8P
2316 016260 177770      .WORD   GT84-GT8P
2317
2318
2319 016262 004767 000020      GTSIX:  JSR      PC,GTCHR
2320 016266 020027 000040      CMP      RET1,#40
2321 016272 002546      BLT      LBAD
2322 016274 020027 000137      CMP      RET1,#137
2323 016300 003143      BGT      LBAD
2324 016302 000207      RTS      PC
2325
2326
2327
2328 016304 005726      GTCHP:  TST      (SP)+      ;UPDATE THE STACK.
2329
2330 016306 012700 015772      GTCHR:  MOV      #P10IC-30000,RET1      ;SET UP POINTER TO THE INPUT CHARACTER.
2331 016312 004767 000064      GTCHL:  JSR      PC,CHECK
2332 016316 005710      TST      @RET1
2333 016320 001774      BEQ      GTCHL
  
```



```

2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404 016520 005002
2405 016522 012712 172000
2406 016526 012706 015770
2407
2408 016532 005003
2409 016534 004767 000070
2410 016540 105300
2411 016542 001373
2412 016544 004767 000060
2413
2414 016550 004767 000072
2415 016554 010005
2416 016556 162705 000004
2417 016562 022705 000002
2418 016566 001437
2419 016570 004767 000052
2420 016574 010001
2421
2422 016576 004767 000026
2423 016602 002010
2424 016604 105703
2425 016606 001751
2426
2427 016610 012700
2428
2429 016612 102 175
2430 016614 004767 000110
2431 016620 000167 177210
2432
2433 016624 110021
2434 016626 000763
2435
2436 016630 004767 177276
2437 016634 060003
2438 016636 042700 177400
2439 016642 005305
2440 016644 000207
2441
2442 016646 004767 177756
2443 016652 010046
2444 016654 004767 177750
2445 016660 000300

```

; THE L O A D E R

```

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.

```

```

2446 016662 052600          BIS      (SP)+,LBYT          ;AND RETURN WITH A 16 BITER.
2447 016664 000207          RTS      PC
2448
2449 016666 004767 177754    LJMP:   JSR      PC,LGWRD          ;GT A WORD
2450 016672 010046          MOV      LBYT,-(SP)          ;SAVE ON THE STACK.
2451 016674 004767 177730    JSR      PC,LPTR          ;GT A CHARACTER.
2452 016700 105703          TSTB    LCKSM          ;IS IT ZERO?
2453 016702 001342          BNE     LBAD          ;YEP. WHAT CRAP.
2454 016704 032716 000001    BIT     #1,(SP)          ;IS IT ODD?
2455 016710 001406          BEQ     LJMP1          ;YEP. START PROGRAM GOING NOW.
2456 016712 012700          MOV     (PC)+,RET1        ;TELL PDP-10 WE'VE LOADED OK.
2457
2458 016714      107      175    :      .BYTE    175,107        ;'CTRL GOOD'
2459 016716 004767 000006    JSR     PC,SENDIT        ;'GOOD CTRL'
2460 016722 000000          HALT
2461 016724 000776          BR      .-2
2462
2463 016726 000136    LJMP1:  JMP     @ (SP)+          ;AND AWAY WE GO.
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481 016730 004767 177446    SENDIT: JSR     PC,CHECK          ;POLL THE OUTPUT DEVICE NOW.
2482 016734 005767 027036    TST     P100C          ;OUTPUT CLEAR?
2483 016740 001373          BNE     SENDIT          ;NOPE. LOOP AWHILE LONGER.
2484 016742 010067 006650    MOV     RET1,P100B      ;SEND OUT THE CHARACTER.
2485 016746 105000          CLRB   RET1            ;CLEAR THE BYTE.
2486 016750 000300          SWAB   RET1            ;AND SWAP THEM NOW.
2487 016752 001366          BNE     SENDIT          ;IF NOT EQUAL, REPEAT.
2488 016754 000207          RTS      PC
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501

```



```
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514 016756 170256      DISPRG: .WORD 170256      ;LOAD STATUS REGISTER FOR NORMAL OPERATION.
2515 016760 115124      .WORD 115124      ;SET POINT MODE, 'NORMAL'.
2516 016762 000000      .WORD STARTX      ;X COORDINATE
2517 016764 001360      .WORD STARTY      ;Y COORDINATE
2518 016766 100000      .WORD 100000      ;SET CHARACTER MODE.
2519 016770 160000      .WORD JMPDIS      ;THEN JUMP TO THE POWERFAIL LOCATION.
2520 016772 000030      .WORD PWRFAIL+4   ;TO DISPLAY USERS CHARACTERS.
2521 016774 000000      .WORD 0
2522 016776 000000      .WORD 0
2523
2524
2525
2526      ;STARTING FROM HERE TO THE TOP OF MEMORY
2527      ; A BACKGROUND WORSE CASE NOISE TASK WILL BE EXECUTED
2528
2529 017000 000000      BUFFER: 0
2530
2531      .END
```


DSWR =	177570	473#	586				
ENDCOR	006036	1271#	1272				
ERRVEC=	000004	472#	584	585*	592*		
FF	006256	1359	1389#				
FFLOOP	006262	1384	1391#	1394			
FILEO	007214	1808	1811#	1879			
FILEOA	005702	602*	667*	671*	1064#		
FILEOB	005704	671	1065#				
FILEOC	005732	602	667	1064	1071#		
FILEOO	002772	550	853#	1073			
GETCHR	006564	1342	1533#	1555			
GETDL	006516	1511#	1516	1520	1522	1533	1537
GETDL1	006546	1512	1519#				
GETEXT	006650	1535	1541	1561#			
GETSIX	006630	1555#	1593	1608			
GET8	006664	1593#	1703				
GET8TB	006700	1596	1598#				
GET81	006712	1598	1608#				
GET82	006740	1599	1620#				
GET83	006772	1600	1635#				
GET84	006706	1604#					
GRAPHX=	120000	206#	900	1760#	1860		
GRAPHY=	124000	207#	886	1761#	1871		
GRPINC	003054	603*	663*	664	666*	884#	
GTADD	001000	479#	564				
GTBRL	001004	481#	596	598	600		
GTBUSE	006424	1444#	1467				
GTCHL	016312	2331#	2333				
GTCHP	016304	2328#	2337	2339			
GTCHR	016306	2237	2319	2330#	2351		
GTDLYO	001314	561#	601*	660*	662*		
GTDNE1	001170	524#	596*				
GTDONE	001166	523#	595*				
GTLPEN	002100	597	671#				
GTLPH	001172	526#	597*				
GTLPH1	001174	527#	598*				
GTNP	016366	2341	2348#				
GTPC	001146	513#	550*	563	668*	672*	
GTP84	016144	2270	2272#				
GTSHIF	002116	599	675#				
GTSIX	016262	2268	2277	2319#			
GTSOTM	001176	529#	599*				
GTSOT1	001200	530#	600*				
GTSR	001150	514#	655	829*			
GTSTOP	002010	595	655#				
GTST1	002062	661	665	667#			
GTVCT	001002	480#	568				
GTXPOS	001152	515#					
GTYP0S	001154	516#					
GT40PC=	172000	1201#	1202	1444*	1478	1808*	
GT40SR=	172002	1202#	1386*				
GT8	016132	2268#	2436				
GT8P =	016152	2275#	2313	2314	2315	2316	
GT8TB =	016250	2273	2311#				
GT81	016152	2277#	2313				
GT82	016200	2288#	2314				

P100C = 045776	2186#	2187	2361	2365	2366*	2374	2482		
P100S = 025614	2178#	2183	2223*	2363					
RCVEC 007764	2039#								
RC11 007720	2020#	2039							
REDCHR 002316	708*	709*	710	723#					
RELATV= 130000	208#	911	916	1762#	1846				
RES 007506	1930#	1945							
RESTR 006060	1296#	1544							
RFVEC 007714	2017#								
RF11 007600	1968#	2017							
RKVEC 007760	2037#								
RK11 007610	1973#	2037							
ROMADD 001014	487#	786							
ROMORG= 166000	2170#								
ROTVL 010046	2063#	2080							
RPNT 002312	556*	707	711*	712	714*	715	721#		
RPVEC 007770	2041#								
RP11 007654	1997#	2041							
RSTR 016034	2227#	2344	2431						
ROSAV 002634	802*	810#							
R1SAV 002636	803*	811#							
R2SAV 002640	804*	812#							
R3SAV 002642	805*	813#							
R4SAV 002644	806*	814#							
R5SAV 002646	807*	815#							
SENDIT 016730	2430	2459	2481#	2483	2487				
SETDUN 006126	1311	1320#							
SETLP1 006074	1302#	1304							
SETLP2 006110	1310#	1317							
SETLP3 006116	1314#	1316							
SETUP 006432	1307	1465#							
SHORTV= 104000	203#	921	926	1757#	1834	1836	1838	1840	
SIZE 001312	560#	647*	648*	2048					
START 006000	500	1260#							
STARTA 016000	503	2215#							
STARTB 001022	477	491#	629	658	676	684	704	733	847
STARTX= 000000	2172#	2516							
STARTY= 001360	2173#	2517							
STATSA= 170000	227#	941	945	951	957	963	969	975	1785#
STATSB= 174000	235#	603	664	666	856	884	1795#	1855	
STKSRT= 015770	2188#	2227	2406						
STOP 007550	1944#	1948							
SWR 001230	495	497	547#	586*	587	591*			
SWREG 000170	475#	591							
SYNOFF= 000010	233#								
SYNON = 000004	234#	1792#							
TAB 006222	1356	1374#	1378						
TABLE 007562	1931	1951#							
TABLEX 010240	794	2108#							
TABLEO 010124	791	2089#							
TABOOT 007500	1928#	1958							
TACS = 177500	1926#	1928							
TAPES 007662	1980	1989	2002#						
TCVEC 007774	2043#								
TC11 007620	1979#	2043							
TKB 001214	538#	738							

SCROLLING ROM BOOTSTRAP FOR THE GT40
CDGTED.P11 31-AUG-79 11:13

MACY11 30G(1063) 31-AUG-79 13:00 PAGE 25
CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0055

GRAPH	294#	887	902		
OCTGN	248#	912	917	922	927
OCTGON	258#	931	935		
PAT1	277#	943	950		
PAT2	282#	968	974		
PAT3	287#	956	962		

. ABS. 017002 000 CON RW ABS LCL I

ERRORS DETECTED: 0

CDGTED,CDGTED/CRF=CDGTED
RUN-TIME: 25.8 SECONDS
RUN-TIME RATIO: 22/9=2.4
CORE USED: 7k (13 PAGES)