

VT61/VT71T

MULTI LINE LOOPBACK TEST
MD-11-DZVTL-A

EP-DZVTL-A-DL-A
COPYRIGHT © 1977
FICHE 1 OF 1

APR 1977

MADE IN USA



The left side of the page contains a grid of 30 small, illegible tables or diagrams arranged in 10 rows and 3 columns. Each cell in the grid appears to contain a small table with multiple columns and rows of text, but the text is too small and faded to be legible. The tables are separated by thin white lines.

B01

IDENTIFICATION

PRODUCT CODE:	MAINDEC-11-DZVTL-A-D
PRODUCT NAME:	VT61/T & VT71/T LOOPBACK TEST
PROGRAM DATE:	APRIL 1977
MAINTAINER:	DIAGNOSTICS

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION. DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT MAY APPEAR IN THIS DOCUMENT.

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

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

COPYRIGHT (C) 1977, BY DIGITAL EQUIPMENT CORPORATION

1.0 ABSTRACT.

THIS PROGRAM IS DESIGNED AS A MAINTENANCE AID FOR FIELD SERVICE PERSONEL. IT PLACES THE VT61/T IN LOOPBACK MODE AND TRANSMITS A 256 RANDOM CHARACTER PATTERN VIA DH-11, AND COMPARES LOOPBACKED DATA FOR ERRORS.

2.0 REQUIREMENTS.

2.1 EQUIPMENT

- A. PDP-11 SYSTEM WITH AT LEAST 4K OF CORE.
- B. TTY
- C. DH-11
- D. VT61/T OR LOOPBACK CONNECTOR FOR DH-11
- E. VT71/T WITH MRV11-VA CARD INSTALLED
- F. KW11/L OR KW11/P.

2.2 STORAGE.

4K OF CORE

3.0 LOADING PROCEDURE

THE ABS LOADER IS USED TO LOAD THE PROGRAM AND OVERLAYS.

4.0 OPERATING PROCEDURES.

- A.
 1. LOAD ADDRESS 200 AND START TO ENTER PARAMS FROM CONSOLE TTY, PROCEED TO SECTION B.
*THE PROGRAM MAY BE RESTARTED AT LOC 204 (ONCE PARAMETERS HAVE ALREADY BEEN SELECTED)
- B. CONSOLE DIALOGUE PARAMETER INPUT
 1. THE PROGRAM WILL TYPE THE DEFAULT BUS ADDRESS OF THE INTERFACE UNDER TEST.
 - A. TYPE A CAR. RETURN TO USE DEFAULT BUS ADDRESS
 - B. TYPEIN ACTUAL BUS ADDRESS
 2. THE PROGRAM WILL TYPE OUT THE DEFAULT VECTOR ADDRESS
 - A. TYPE A CAR. RETURN TO USE DEFAULT ADDRESS
 - B. TYPEIN ACTUAL VECTOR ADDRESS
 3. THE PROGRAM WILL TYPE OUT THE DEFAULT INTERFACE PRIORITY
 - NOTE: 200=PRIO 4, 240=PRIO 5, 300=PRIO 6, ETC.
 - A. TYPE A CAR. RETURN TO USE DEFAULT VALUE
 - B. TYPEIN ACTUAL VALUE
 4. THE PROGRAM WILL ASK FOR THE LINE SELECTION
 - A. TYPE IN VALUES (IN OCTAL) OF LINES DESIRED SEPARATED BY COMMAS

(U) TO SELECT LINES 17,16,8,2,0
ENTER 17,16,8,2,0 <CR> THE ORDER

E01

VT61/T LOOPBACK TEST MACY11 27(1006) 02-MAR-77 16:44 PAGE 3
DZVTLA.P11 02-MAR-77 16:43

DOES NOT MATTER

5. THE PROGRAM WILL TYPEOUT THE DEFAULT VALUE OF PARAM#1
 - A. TYPE A CAR. RETURN TO USE DEFAULT VALUE
 - B. ENTER ACTUAL VALUE
6. THE PROGRAM WILL REQUEST THAT SWITCH REGISTER BE SET.
 - A. SETUP SWITCH REGISTER AS SPECIFIED IN STEP D.
AND TYPE A CAR. RETURN.

NOTE: IF ANY OF THE ABOVE ITEMS 2 THRU 7 WERE CHANGED BY ENTERING NEW VALUES, THE NEW VALUE BECOMES THE DEFAULT VALUE FOR SUBSEQUENT RESTARTS OF THE PROGRAM.

THIS PROGRAM HAS BEEN MODIFIED TO RUN ON A PROCESSOR WITH OR WITHOUT A HARDWARE SWITCH REGISTER. WHEN FIRST EXECUTED THE PROGRAM TESTS THE EXISTENCE OF A HARDWARE SWITCH REGISTER. IF NOT FOUND A SOFTWARE SWITCH REGISTER LOCATION (SWREG=LOC. 176) IS DEFAULTED TO. IF THIS IS THE CASE, UPON EXECUTION THE CONTENTS OF THE SWREG ARE DUMPED IN OCTAL ON THE CONSOLE TTY AND ANY CHANGES ARE REQUESTED

(IE) SWR=XXXXXX NEW=

POSSIBLE RESPONSES ARE:

1. <CR> IF NO CHANGES ARE TO BE MADE
2. 6 DIGITS 0-7 TO REPRESENT IN OCTAL THE NEW SWITCH REGISTER VALUE ;LAST DIGIT FOLLOWED BY <CR>.
3. ^U TO ALLOW REENTERING VALUE IF ERROR IS COMMITTED KEYING IN SWREG VALUE.

BUILT INTO THE PROGRAM IS THE ABILITY TO DYNAMICALLY CHANGE THE CONTENTS OF SWREG DURING PROGRAM EXECUTION. BY STRIKING ^G (CNTL G) ON CONSOLE TTY THE OPERATOR SETS A REQUEST FLAG TO CHANGE THE CONTENTS OF SWREG, WHICH IS PROCESSED IN KEY AREAS OF THE PROGRAM CODE (IE) ERROR ROUTINES, AFTER HALTS END OF PASS, AND OTHER APPLICABLE AREAS.

D. OPERATIONAL SWITCH SETTINGS.

SW15=1 HALT ON ERROR
 SW14=1 SINGLE PASS AND REQUEST NEW PARAMS
 SW13=1 INHIBIT ERROR TYPEOUTS
 SW12=1 INHIBIT ALL TYPEOUTS EXCEPT ERRORS
 SW09=1 DATA SELECT
 0=USE 256 RANDOM CHARACTER PATTERN
 1=USE DATA INPUT FROM OPERATOR
 SW07=1 DO NOT TEST RECEIVED DATA
 SW04=1 TYPE OUT LINE BEING TESTED
 SW03=1 REPORT FIRST DATA COMPARE ERROR ONLY
 =0 REPORT ALL DATA COMPARE ERRORS
 SW02=1 TYPE TRANSMITTED AND RECEIVED DATA

IF OPERATOR SPECIFIED DATA WAS INDICATED, THE PROGRAM WILL TYPE A REQUEST FOR THE DATA. DATA MAY BE ENTERED AS ASCII CHARACTERS OR OCTAL CODE. TYPE IN THE DATA TERMINATED WITH A CR. OCTAL CODE MAY BE ENTERED BY TYPING AN ↑(UP ARROW) FOLLOWED BY THE OCTAL CODE (IN THE RANGE 40 TO 177) SEPERATED BY SPACES AND TERMINATED BY ↑(UP ARROW).

F. TEST EXECUTION MAY BE INTERRUPTED BY TYPING THE FOLLOWING CHARACTERS ON THE CONSOLE TTY.

QUESTION MARK = PRINTOUT FIRST 8 WORDS OF INPUT BUFFER.(ASCII)
 THEN TYPE EITHER:

*IXXXXXX TO PRINTOUT THE 8 WORDS
 AT LOC XXXXXX.

 *BXXXXXX TO PRINTOUT THE 16 BYTES
 AFTER LOC XXXXXX.

 *C TO CONTINUE

PROGRAM MUST BE RESTARTED AT 200 AFTER PRINTING.

↑C = EXIT LOOPBACK MODE (E.E.SEND ↑C TO VT61/T)
 AND RESTART PROGRAM AT LOC 200.

9.0

PARAMETERS FOR THE DH11
 PARAM#1 IS LOADED INTO THE LINE PARAMETER REGISTER, (LPR)
 BITS 0,1 CHARACTER LENGTH, DEFAULT= 8 BITS (11)
 BIT 2 STOP BITS, DEFAULT= 1 STOP BITS (0)
 BIT 4 PARITY ENABLED (1), DEFAULT= (1)
 BIT 5 ODD PARITY (1), DEFAULT= (0)
 BITS 6-9 RECEIVER SPEED, DEFAULT= 9600 BAUD (1101)
 BITS 10-13 TRANSMIT SPEED, DEFAULT= 9600 BAUD (1101)
 BIT 14 HALF DUPLEX (1), DEFAULT= FULL DUPLEX (0)

****DEFAULT PARAM#1=33523*****

GO1

VT61/T LOOPBACK TEST MACY11 27(1006) 02-MAR-77 16:44 PAGE 5
DZVTLA.P11 02-MAR-77 16:43

255	000300	PRTY6= 300
256	000340	PRTY7= 340
257		
258		: MISCELLANEOUS BIT ASSIGNMENT
259	100000	BIT15= 100000
260	040000	BIT14= 40000
261	020000	BIT13= 20000
262	010000	BIT12= 10000
263	004000	BIT11= 4000
264	002000	BIT10= 2000
265	001000	BIT09= 1000
266	000400	BIT08= 400
267	000200	BIT07= 200
268	000100	BIT06= 100
269	000040	BIT05= 40
270	000020	BIT04= 20
271	000010	BIT03= 10
272	000004	BIT02= 4
273	000002	BIT01= 2
274	000001	BIT00= 1
275		.EQUIV BIT09, BIT9
276		.EQUIV BIT08, BIT8
277		.EQUIV BIT07, BIT7
278		.EQUIV BIT06, BIT6
279		.EQUIV BIT05, BIT5
280		.EQUIV BIT04, BIT4
281		.EQUIV BIT03, BIT3
282		.EQUIV BIT02, BIT2
283		.EQUIV BIT01, BIT1
284		.EQUIV BIT00, BIT0
285		
286	000004	RCSR=R4
287	000020	LOOP=20
288	000200	NODAT=200
289	000000	XCSR=0
290	000002	NRCR=2
291	000004	LPR=4
292	000006	CAR=6
293	000010	BCR=10
294	000012	BAR=12
295	000016	SSR=16
296	000100	RIE=BIT6
297	040000	HALF.DUPLEX=BIT14
298	040000	ESC=BIT14
299	004000	MC=BIT11
300	010000	SIE=BIT12
301	040000	SI=BIT14
302	020000	TIE=BIT13
303	100000	TI=BIT15
304	002000	NEM=BIT10
305	040000	DO=BIT14
306	020000	FE=BIT13
307	010000	PE=BIT12
308	000000	STAT=RO
309	000010	FSTERR=BIT3
310	100000	XFLG=100000

;XMIT COMPLETE FLAG

```

311          040000          .          RFLG=40000          ;RCV COMPLETE FLAG
312          020000          .          DSFLG=20000          ;DATA SET STATUS CHANGE FLAG
313          020000          .          BIT13=20000          ;INHIBIT PRINTOUTS
314          .VECTOR ADDRESSES
315          000004          ERRVEC= 4
316          000010          RESVEC= 10
317          000014          TBITVEC=14
318          000014          TRTVEC= 14
319          000014          BPTVEC= 14
320          000020          IOTVEC= 20
321          000024          PWRVEC= 24
322          000030          EMTVEC= 30
323          000034          TRAPVEC=34
324          .EQUIV R4,CSR
325          .EQUIV R4,RCSR
326
327          000000          .=0
328          ;TRAP CATCHER IN UNUSED LOCATIONS FROM 0 - 776
329          ;LOCATION 0 WILL CATCH IMPROPERLY LOADED VECTORS
330          000024          .=24
331          000024          006750          $PWROD
332          000026          000340          340
333          000030          005600          $HLT
334          000032          000340          340
335          000034          006710          $TRAP
336          000036          000340          340
337
338
339          000100          .=100
340          000100          006456          TIMER
341          000102          000340          340
342
343          000174          .=174
344          000174          000000          DISREG:0
345          000176          000000          SWREG: 0
346
347          000200          .=200
348
349          000200          000137          004466          JMP      @#BEGIN          ;JUMP TO STARTING ADDRESS OF PROGRAM
350          000204          000137          004632          JMP      @#RSTART          ;RESTART AT 204. DO THE RESTART.

```

```

351                                     ;*****
352                                     ;.=1100
353                                     ;ROUTINE TO TYPE ASCII MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
354                                     ;THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
355                                     ;NOTE1: NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
356                                     ;NOTE2: $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
357
358                                     ;CALL:
359                                     ;1) USING A TRAP INSTRUCTION
360                                     ;      TYPE      ,MESADR      ;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
361                                     ;OR
362                                     ;      TYPE
363                                     ;      MESADR
364
365 001100 010046 STYPE: MOV      RO,-(SP)      ;SAVE RO
366 001102 010146      MOV      R1,-(SP)      ;SAVE R1
367 001104 017600      MOV      @4(SP),RO      ;GET ADDRESS OF ASCIZ STRING
368 001110 062766 000004 000004      ADD      #2,4(SP)      ;ADJUST RETURN PC
369 001116 012701 000101 75:      MOV      #101,R1      ;SET CHARACTER COUNT
370 001122 005301 15:      DEC      R1
371 001124 001432      BEQ      6$
372 001126 112046      MOVB     (RO)+,-(SP)      ;PUSH CHARACTER TO BE TYPED ONTO STACK
373 001130 001004      BNE      2$
374 001132 005726      TST      (SP)+
375 001134 012601      MOV      (SP)+,R1      ;RESTORE R1
376 001136 012600      MOV      (SP)+,RO      ;RESTORE RO
377 001140 000002      RTI
378 001142 004737 001174 25:      JSR      PC,5$
379 001146 122726 000012 35:      CMPB   #12,(SP)+
380 001152 001363      BNE      1$
381 001154 013746 001574      MOV      NULL,-(SP)
382                                     ;AND THE NULL CHAR.
383                                     ;DOES A NULL NEED TO BE TYPED?
384 001160 105366 000001 45:      DECB   1(SP)
385 001164 002770      BLT      3$
386 001166 004737 001174      JSR      PC,5$
387 001172 000772      BR       4$
388 001174 105777 000614 55:      TSTB   @TPS
389 001200 100375      BPL      5$
390 001202 116677 000002 000606      MOVB   2(SP),@TPB
391 001210 000207      RTS      PC
392 001212 104400 010266 65:      TYPE   ,CRLF
393 001216 000737      BR       7$
394                                     ;TYPE <CR><LF>
395                                     ;NEXT LINE

```

```

395
396
397
398 001220 013746 001760
399 001224 042737 000070 001760
400 001232 104400 010262
401 001236 104402
402 001240 012637 001342
403 001244 104400 001342
404 001250 042737 000040 001544
405 001256 122737 000127 001544
406 001264 001004
407 001266 052737 000010 001760
408 001274 000430
409 001276 122737 000102 001544 1S:
410 001304 001004
411 001306 052737 000020 001760
412 001314 000420
413 001316 122737 000103 001544 2S:
414 001324 001014
415 001326 012637 001760
416 001332 052737 000040 001760
417 001340 000413
418 001342 000000
419 001344 104400 010256 6S:
420 001350 104400 010266 4S:
421 001354 000721
422 001356 005037 001546 3S:
423 001362 005726
424 001364 104406 001546
425 001370 012700 001524
426 001374 062710 000020
427 001400 032737 000040 001760
428 001406 001005
429 001410 013737 001546 001524
430 001416 001001
431 001420 022020
432
433 001422 012001
434 001424 001675
435 001426 104400 010266
436 001432 010146
437 001434 004037 006514
438
439 001440 032737 000010 001760
440 001446 001014
441 001450 012702 000020
442 001454 005046
443 001456 112116
444 001460 104400 007500
445 001464 004037 006502
446 001470 003
447 001471 001
448 001472 005302
449 001474 001367
450 001476 000751

```

```

*****
;
;          DEBUG DUMP ROUTINE
;
*****
DMPHLT: MOV     FLAG, -(SP)
        BIC     #70, FLAG      ;INIT FLAGS
        TYPE   ,ASTRK        ;TYPE #
        GETCHR
        MOV     (SP)+, 6S
        TYPE   , 6S
        BIC     #BIT5, $CHAR   ;STRIP LOWER CASE OFF
        CMPB   #127, $CHAR    ;W? FOR WORD
        BNE    1S
        BIS     #BIT3, FLAG    ;SET FLAG BIT
        BR     3S
        CMPB   #102, $CHAR    ;B? FOR BYTE
        BNE    2S
        BIS     #BIT4, FLAG
        BR     3S
        CMPB   #103, $CHAR    ;C? FOR CONTINUE
        BNE    3S
        MOV     (SP)+, FLAG
        BIS     #BIT5, FLAG
        BR     DUMP
        6S:   000000
        4S:   TYPE     , QUES
        TYPE   , CRLF
        BR     DMPHLT
        3S:   CLR     WORK
        TST    (SP)+
        ACCEPT , WORK
DUMP:  MOV     #DMPHLT, R0      ;INIT DUMP LIST
        ADD    #20, (R0)      ;BUMP ADDRESS
        BIT    #BIT5, FLAG
        BNE    L1
        MOV    WORK, DMPHLT
        BNE    L1
        CMP    (R0)+, (R0)+   ;SKIP 1ST TWO ENTRIES
        L1:   MOV     (R0)+, R1 ;GET ADDR OF DATA FROM LIST
        BEQ   DMPHLT         ;BR IF END OF LIST
        TYPE   , CRLF
        MOV    R1, -(SP)     ;PUSH ADDR ON STACK
        JSR   R0, $B2016    ;PRINT OUT ADDRESS
        BIT    #BIT3, FLAG
        BNE    L3
        MOV    #20, R2      ;SET WORD COUNTER = 8
        L2:   CLR     -(SP)
        MOVB  (R1)+, (SP)
        TYPE   , MSG00
        JSR   R0, $B20CT
        .BYTE 3
        .BYTE 1
        DEC   R2           ;DECREMENT WORD COUNTER
        BNE   L2          ;BR IF NOT = 0
        BR    L1          ;GET NEXT ENTRY

```

MO1

451			
452	001500	012702	000010
453	001504	012146	
454	001506	104400	007500
455	001512	004037	006514
456	001516	005302	
457	001520	001371	
458	001522	000737	
459	001524	000000	
460	001526	000000	
461	001530	000001	
462	001532	000001	
463	001534	000000	
464	001536	000000	
465	001540	000000	

```

L3:  MOV  #10,R2
15:  MOV  (R1)+, -(SP)
      TYPE MSG00
      JSR  R0, $B2016
      DEC  R2
      BNE  15
      BR   L1
DMPST: 0
      0
.RX:   .BLKW 1
.TX:   .BLKW 1
      0
      0
      0

```

```

;DECREMENT THE WORD COUNT
;GET NEXT ENTRY
;RESERVED FOR SW. REG
;END OF TABLE FOR SW. REG

```

```

466
467
468
469 001546
470 001542 000000
471 001544 000000
472 001546 000000
473 001550 000000
474 001552 000000
475 001554 000000
476 001556 000000
477 001560 000000
478 001562 000000
479 001564 000000
480 001566 000000
481 001570 000000
482 001572 000010
483 001574 000006
484 001576 000000
485 001600 000
486 001601 012
487 001602 000000
488 001604 000000
489 001606 000000
490 001610 000120
491 001612 000067
492 001614 000047
493 001616 000037
494 001620 000034
495 001622 000027
496 001624 000021
497 001626 000011
498 001630 000006
499 001632 000005
500 001634 000003
501 001636 000003
502 001640 000003
503 001642 022333
504 001644 123456
505 001646 000000 000000
506 001652 000000
507 001654 000000
508 001656 000000
509 001660 000000
510 001662 000000
511 001664 000000
512 001666 000000
513 001670 000000
514 001672 000000
515 001674 000000
516 001676 000000
517 001700 000000
518 001702 000000
519 001704 000000
520 001706 000000
521 001710 000000

```

```

*****
CONSTANTS AND WORKING STORAGE
*****

```

```

$WORK=WORK
SLPADR: 0
SCHAR: 0
WORK: 0
WORK1: 0
WORK2: 0
WORK3: 0
WORK4: 0
WORK5: 0
WORK6: 0
LINE: .WORD 0
LINMSK: .WORD 0
FLAGS: 0
STIME: .WORD 10
NULL: .WORD 6
SPASS: 0
SICNT: .BYTE 0
STIME: .BYTE 10.
SHTAD: 0
SERTTL: 0
.BAUD: .WORD 0
        .WORD 80.
        .WORD 55.
        .WORD 39.
        .WORD 31.
        .WORD 28.
        .WORD 23.
        .WORD 17.
        .WORD 9.
        .WORD 6.
        .WORD 5.
        .WORD 3.
        .WORD 3.
L: 22333
H: 123456
COUNT: .WORD 0,0
QTPIE: 000000
SAVR0: 0
SAVR1: 0
SAVR2: 0
SAVR3: 0
SAVR4: 0
DELAY: 0
BACK: 0
STOP: 0
SERRR: 0
TEMP1: 0
TEMP2: 0
BARTMP: 0
SNCFLG: 0
TRNFLG: 0

```

;CONTAINS NULL CHARACTER FOR FILLS

```

:TIME DELAY FOR 0 BAUD
:TIME DELAY FOR 50 BAUD
:TIME DELAY FOR 75 BAUD
:TIME DELAY FOR 110 BAUD
:TIME DELAY FOR 134.5 BAUD
:TIME DELAY FOR 150 BAUD
:TIME DELAY FOR 200 BAUD
:TIME DELAY FOR 300 BAUD
:TIME DELAY FOR 600 BAUD
:TIME DELAY FOR 1200 BAUD
:TIME DELAY FOR 1800 BAUD
:TIME DELAY FOR 2400 BAUD
:TIME DELAY FOR 4800 BAUD
:TIME DELAY FOR 9600 BAUD

```

```

522 001712 177
523 001714 001714
524 001714 044104 000040
525 001714 160020
526 001720 000300
527 001722 000240
528 001724 000000
529 001726 000000
530 001730 033523
531 001732 177777
532 001734 000000
533 001736 000000
534 001740 000000
535 001742 000000
536 001744 000000
537 001746 000000
538 001750 000000
539 001752 011042
540 001754 000
541 001754 000
542 001755 001
543 001755 001
544 001756 115
545 001756 000000
546 001760 177570
547 001762 177570
548 001764 177570
549 001766 000000
550 001770 000000
551 001772 000000
552 001774 000000
553 001776 000000
554 002000 000000
555 002002 000000
556 002004 000000
557 002006 000000
558 002010 177560
559 002012 177562
560 002014 177564
561 002016 177566
562 002020 003224
563 002022 003024
564 002024 000400
565 003024 000100
566 003224 000400
567 004224 002 033 117
568 004232 004232

```

```

FILL: .BYTE 177
.EVEN
VISR:
ISR: .ASCIZ /DH /
BA: 160020
VA: 300
PRIOR: 240
PARAM1: 0
PARAM2: 033523
PARAM3: 177777
IRDA: .WORD 0
IXDA: .WORD 0
SETTLE: .WORD 0
IRCC: .WORD 0
B2016: .WORD 0
TIME: .WORD 0
MODEA: .WORD 0
TX. TERM: .WORD START
RX. TERM: .BYTE 000
ESC. TERM: .BYTE 001
FLAG: .WORD 115
SWR: 177570
DISPLAY: 177570
SXCSR: 0
SRCSR: 0
ERCSR: 0
ERDBR: 0
DSSTAT: 0
XCC: 0
RCC: 0
RDA: 0
XDA: 0
TKS: 177560
TKB: 177562
TPS: 177564
TPB: 177566
; DATA PATTERN ADDRESS TABLE AND PATTERNS
DAT: .WORD DP1
; ADDRESS OF VARIABLE DATA BUFFER
; ADDRESS OF MESSAGE 1
; RECEIVER DATA BUFFER STARTS HERE..
IBUF: .BLKW 400
VDB: .BLKW 100
; VARIABLE DATA BUFFER
DP1: .BLKW 400
LPB: .BYTE 002,033,117,115,000
.EVEN
; ISR NAME
; BUS ADDRESS
; VECTOR ADDRESS
; PRIORITY
; PARAM #1
; PARAM #2
; PARAM #3
; INITIAL READ DATA ADDRESS
; INITIAL XMIT DATA ADDRESS
; LINE SETTLE DELAY FLAG
; ADDR OF BIN TO OCT TYPE ROUTINE
; TIMER
; ADDR OF START OF PROGRAM
; TRANSMITTER TERMINATING CHAR.
; RECEIVER TERMINATING CHAR.
; ESCAPE SEQ TERMINATOR
; SAVED XMIT CSR
; SAVED RCV CSR
; RCV CSR SAVED ON ERROR
; RCV DATA REG SAVED ON ERROR
; RCV CSR SAVED ON DS CHANGE
; XMIT CHAR COUNT
; RCV CHAR COUNT
; RCV DATA ADDR.
; XMIT DATA ADDR.

```



```

572
573
574
575 004466
576 004466 012706 001070
577 004472 104410 000340
578 004476 104414
579 004500 005037 001564
580 004504 005037 001576
581 004510 012737 004466 001542
582 004516 005037 001570
583 004522 042737 177773 001760
584 004530 000005
585
586
587
588
589 004532 012700 001714
590 004536 032737 000004 001760
591 004544 001005
592 004546 104400 007502
593 004552 052737 000004 001760
594 004560 010004
595 004562 022020
596 004564 104400 007571
597 004570 004737 005550
598
599 004574 104400 007610
600 004600 004737 005550
601
602 004604 104400 007632
603 004610 004737 005550
604
605 004614 004737 011704
606
607 004620 104400 007670
608 004624 004737 005550
609 004630 000410
610
611
612
613
614
615 004632 104400 010253
616 004636 000005
617 004640 105227 000000
618 004644 001375
619 004646 005037 001564
620 004652 022737 000176 001762
621 004660 001007
622 004662 052737 000002 001760
623 004670 104400 007724
624 004674 104416
625 004676 000417
626 004700 104400 007724

```

```

*****
: START OF PROGRAM
*****
BEGIN:
MOV #STACK, SP ; SETUP THE STACK POINTER
STPS, PRTY7 ; SET PSW TO LEVEL 7
SUSWR
CLR LINE ; ALWAYS START AT LINE 0
CLR SPASS ; CLEAR THE PASS COUNT
MOV #BEGIN, SLPADR ; INITILIZE THE LOOP ADDRESS FOR SCOPE
CLR FLAGS ; RESET FLAGS
BIC #1<BIT2>, FLAG ; CLEAR ALL BUT TITLE BITS
RESET

*****
: ISR PARAM INPUT ROUTINE
*****
GETIT: MOV #VISR, R0 ; PRESET ISR ADDR IN R0
BIT #BIT2, FLAG ; FIRST TIME THRU
BNE GETBA ; NO
TYPE MSG01 ; TYPE ID
BIS #BIT2, FLAG ; SET TITLE FLAG
GETBA: MOV R0, R4 ; SAVE POINTER
CMP (R0)+, (R0)+ ; INCREMENT ISR POINTER
TYPE MSG03 ; <15><12> BUS ADDRESS=
JSR PC, GETANY ; GET THE BUS ADDR

GETVA: TYPE MSG04 ; <15><12> VECTOR ADDRESS=
JSR PC, GETANY ; GET THE VECTOR ADDR.

GETPRI: TYPE MSG05 ; <15><12> PRIORITY=
JSR PC, GETANY ; GET THE PRIORITY

JSR PC, GETPRM ; GET LINE #'S TO TEST

TYPE MSG07 ; <15><12> PARAMS #2=
JSR PC, GETANY ; GET PARAM
BR SWPRNT ; CONTINUE

RSTART: TYPE ,MFILL
RESET
INCB #0 ; DELAY HERE FOR AWILE
BNE .-4
CLR LINE
SWPRNT: CMP #SWREG, SWR
BNE XSWPNT
BIS #BIT1, FLAG
TYPE ,MSG09
SETSWI
BR REST
XSWPNT: TYPE ,MSG09 ; <15><12> SET SWITCHES

```

```

627 004704 105777 175100      TSTB   @TKS           ;WAIT FOR TTY INPUT
628 004710 100375              BPL     -4            ;LOOP
629 004712 017702 175074      MOV     @TKB,R2       ;RESET DONE FLAG
630 004716 017746 175040      MOV     @SWR, -(SP)   ;
631 004722 004037 006514      JSR     R0,$@2016     ;PRINTOUT SWITCHES
632 004726 104400 010266      TYPE   ,CRLF         ;
633 004732 000401              BR      +4            ;SKIP OVER HALT
634                                     ;*****
635                                     ;      SET SWITCH OPTIONS
636                                     ;*****
637
638 004734 000000      SWRSET: HALT           ;HALT FOR SWITCH SETUP
639
640 004736 012737 004744 001542  REST:  MOV     @RESTRT,SLPADR ;SETUP LOOP
641 004744 017701 175012      RESTRT: MOV     @SWR,R1
642 004750 000301              SWAB   R1
643 004752 042701 177775      BIC     @177775, R1   ;STRIP JUNK
644 004756 016137 002020 001736  MOV     DAT(1), IXDA  ;SETUP INIT DATA ADDR FROM TABLE
645 004764 005701              TST    R1             ;VARIABLE DATA SPECIFIED?
646 004766 001404              BEQ    SUXCC         ;BR IF NO
647                                     ;*****
648                                     ;      GET VARIABLE DATA
649                                     ;*****
650 004770 104400 007746      TYPE   ,MSG10        ;<15><12> ENTER DATA <15><12>
651 004774 104404              GETSTR
652 004776 003024              VDB
653 005000 012737 002024 001734  SUXCC: MOV     @IBUF, IRDA ;SETUP READ BUFFER ADDR
654 005006 004737 007230      JSR    PC,SETBUF    ;GET BYTE COUNTS
655 005012 004737 006064      JSR    PC,$BAUD     ;CALCULATE TIME DELAY
656
657 005016              SWRNXT:
658 005016 012737 004232 000060  MOV     @TTYINT,@#60 ;SETUP TTY VECTOR
659 005024 012737 000340 000062  MOV     @340,@#62   ;
660 005032 012777 000100 174750  MOV     @100,@TKS   ;AND ENABLE INTERRUPTS
661 005040 012702 002024      MOV     @IBUF, R2
662 005044 005022      CLRIB: CLR     (R2)+ ;CLEAR INPUT BUFFER
663 005046 022702 003024      CMP    @VDB,R2
664 005052 001374      BNE   CLRIB

```

```

665                                     ;*****
666                                     ;      SETUP TIMER      *
667                                     ;*****
668
669 005054 012737 000060 006500 SUTIME: MOV    #60,    MSECS ;PRESET COUNTER
670 005062 012737 006456 000100      MOV    #TIMER, 100 ;SETUP LINE CLOCK VECTOR
671 005070 012737 000340 000102      MOV    #340,   102 ;AND PRIORITY
672 005076 012737 005122 000004      MOV    #NOLC,  4   ;SETUP BUS ERROR VECTOR
673 005104 012737 000340 000006      MOV    #340,   6   ;SET PRIORITY
674 005112 052737 000100 177546      BIS    #100,   177546 ;ENABLE LINE CLOCK
675 005120 000441
676
677                                     ;BUS ERROR RETURNS HERE IF NO LINE CLOCK
678
679 005122 012737 006456 000104 NOLC:  MOV    #TIMER, 104 ;SETUP RTC VECTOR
680 005130 012737 000340 000106      MOV    #340,   106 ;AND PRIORITY
681 005136 012737 005172 000004      MOV    #15,    4   ;SETUP BUS ERROR VECTOR
682 005144 012737 000340 000006      MOV    #340,   6   ;SET PRIORITY
683 005152 012737 003100 172542      MOV    #1600., 172542 ;SET COUNTER BUFFER.
684 005160 012737 000111 172540      MOV    #111,   172540 ;ENABLE REAL TIME TIME CLOCK
685 005166 000240
686 005170 000415
687 005172 012737 005216 000100 1$:   MOV    #25, 100 ;SET UP FOR LSI
688 005200 104410 000000
689 005204 005227 000000 3$:   STPS, PRTY0 ;LOWER PSM TO 0
690 005210 001375
691 005212 104400 010055
692 005216 012737 006456 000100 2$:   INC    #0
693                                     BNE    3$
694 005224
695 005224 012737 000006 000004 NORTC: MOV    #6, 2#4 ;SET TRAP VECTOR
696 005232 005037 000006
697 005236 012706 001070
698 005242 104410 000000
699 005246 012737 006514 001744      MOV    #STACK, SP ;SETUP STACK
700                                     MOV    #5B2016, B2016 ;SETUP BIN TO OCT ADDR
701
702
703                                     ;*****
704                                     ;      GOTO THE MODULE AND RUN      *
705                                     ;*****
706
707 005254 004737 011042 VIGO:  JSR    PC, START ;GO TO ISR

```

```

708 ;*****
709 ;      END OF PASS ROUTINE
710 ;*****
711 EOPPY: TST      -(SP)      ;SAVE RETURN
712        TST      $PASS    ;FIRST PASS THRU
713        BEQ      1$       ;YESS NO ITERATIONS
714        INCB     $ICNT     ;INCREMENT ITERATION COUNT
715        CMPB     $TIMES,$ICNT ;DONE ENOUGH YET
716        BGE     8$       ;NO DO SOME MORE
717        MOVB     #1,$ICNT  ;INIT ITERATION COUNTER
718        JSR     PC,NEEXT   ;GO CALCULATE BAR REG
719        BIT      #BIT6,FLAG ;THRU ALL LINES YET
720        BEQ      7$       ;NO
721        BIC     #BIT6+ESC,FLAG ;INIT FLAG AND TYPE END PASS
722        KBDIN
723        TST      $PASS    ;FIRST PASS
724        BEQ      5$       ;
725        BIT      #SW12,$SWR ;INHIBIT TYPEOUTS?
726        BNE     5$       ;BR IF YES
727        TYPE     MSG11    ;<15><12> END OF PASS
728        MOV      $PASS, -(SP)
729        JSR     R0,$B2016 ;PRINTOUT PASS COUNT
730        TYPE     TUCRLF   ;COUPLE FORMS
731        BIT      #SM02,$SWR ;TYPE OUT XMIT +RCV DATA
732        BEQ      6$       ;NO BRANCH
733        TYPE     MSG26    ;TRANSMITTED DATA=
734        MOV      IXDA,2$  ;SET POINTER TO TXBUF
735        TYPE
736        0
737        TYPE     MSG27    ;RECEIVED DATA=
738        MOV      IRDA,4$  ;SET POINTER TO RXBUF
739        TYPE
740        0
741        BIT      #BIT14,$SWR ;ONE PASS ONLY
742        BNE     9$       ;
743        INC      $PASS    ;UP PASS COUNT
744        TST      $PASS
745        BEQ      8$       ;
746        BIT      #LOOP,$SWR ;TYPE UNIT UNDER TEST
747        BEQ      8$       ;NOP
748        TYPE     TESTXX
749        MOVB     LINE, -(SP)
750        JSR     R0,$B20CT ;TYPE UNIT #
751        .BYTE   2
752        .BYTE   1
753        MOV      2(SP),R0  ;GET RETURN ADDRESS
754        STPS,PRTY0
755        JMP     (R0)      ;GO BACK TO MODULE.
756        MOV     #STACK,SP ;RESET THE STACK POINTER.
757        MOV     @#42,R0   ;GET MONITOR ADDRESS
758        BEQ     $DOAGN    ;BR IF NONE
759        JSR     PC,(R0)   ;GO TO MONITOR
760        NOP
761        NOP
762        NOP
763        SDOAGN: JMP     .CTLC ;RESTART TEST

```

```

764
765
766
767
768
769 005550 011046
770 005552 004037 006514
771 005556 104400 010256
772 005562 011037 001546
773 005566 104406 001546
774 005572 013720 001546
775 005576 000207
776
777
778
779
780 005600
781 005600 000240
782 005602 000240
783 005604 005237 001604
784 005610 001775
785 005612 011637 001602
786 005616 162737 000002 001602
787 005624 010146
788
789 005626 032777 020000 174126
790 005634 001075
791
792 005636 104400 010266
793 005642 117701 173734
794 005646 006301
795 005650 016137 006050 005660
796 005656 104400 010275
797 005662 113746 001564
798 005666 004037 006502
799 005672 002
800 005673 001
801 005674 104400 010004
802 005700 013746 001602
803 005704 004037 006514
804 005710 005701
805 005712 001446
806
807
808 005714 022701 000006
809 005720 001023
810 005722 005702
811 005724 001406
812 005726 104400 010124
813 005732 110246
814 005734 004037 006502
815 005740 003
816 005741 001
817 005742 005703
818 005744 001410
819 005746 104400 010140

```

```

;*****
; SUBROUTINE TO INPUT OCTAL WORD FROM OPERATOR
;*****
GETANY: MOV (R0) -(SP) ;PUT WORD ON STACK
        JSR R0,$B2016 ;AND TYPE IT
        TYPE QUES
ANYMOR: MOV (R0),WORK ;PRESET FOR DEFAULT (CR)
        ACCEPT ,WORK ;OCTAL READIN
        MOV WORK, (R0)+ ;MOVE IT TO ISR
        RTS PC ;SUB/ROUTINE EXIT

;*****
; ERROR HLT HANDLER
;*****
$HLT:
        NOP
        NOP
$HLOT: INC $ERTTL ;INCREMENT ERROR COUNTER
        BEQ $HLOT ;MAKE SURE ITS NOT ZERO
        MOV (SP) $HLTAD ;SAVE ADDRESS OF HLT
        SUB #2, $HLTAD ;AND BACK IT UP
        MOV R1, -(SP) ;SAVE R1
        BIT #BIT13, $SWR ;INHIBIT ERR TYPEOUTS?
        BNE TRX ;BR IF YES
        TYPE CRLF
        MOVB $HLTAD,R1 ;EXTRACT HLT CODE
        ASL R1 ;AND ALIGN IT
        MOV ENTAB(R1),.+10 ;GET HEADER ADDRESS
        TYPE EMO ;AND PRINT HEADER
        MOVB LINE, -(SP) ;PUT LINE # ON STACK
        JSR R0,$B20CT ;TYPE IT
        .BYTE 2
        .BYTE 1
        TYPE MSG12 ; < AT LOC >
        MOV $HLTAD, -(SP) ;GET HLT ADDRESS
        JSR R0,$B2016 ;AND PRINT IT
        TST R1 ;HLT CODE = 0?
        BEQ TRX ;BR IF YES
        CMP #6,R1 ;IS IT HLT+3?
        BNE 1$ ;BR IF NO
        TST R2 ;PRINTOUT BAD DATA?
        BEQ 2$ ;BR IF NO
        TYPE MSG23 ;< BAD DATA= >
        MOVB R2, -(SP) ;GET DATA
        JSR R0,$B20CT ;AND PRINT IT
        .BYTE 3
        .BYTE 1
2$: TST R3 ;PRINT OUT GOOD DATA?
        BEQ 3$ ;BR IF NO
        TYPE ,MSG24 ;< GOOD DATA= >

```

```

820 005752 110346          MOVB   R3,-(SP)          ;GET DATA
821 005754 004037 006502   JSR    RO,$B20CT        ;AND PRINT IT
822 005760      003          .BYTE  3
823 005761      001          .BYTE  1
824 005762 104400 010266   TYPE  CRLF
825 005766 000420          BR     TRX
826
827
828 005770 005702          3S:   TST   R2          ;PRINTOUT RCV CSR?
829 005772 001405          BEQ   TR3          ;BR IF NO
830 005774 104400 010015   TYPE  MSG13        ; < RCV CSR=>
831 006000 010246          MOV   R2,-(SP)      ;GET DATA
832 006002 004037 006514   JSR    RO,$B2016    ;AND PRINT IT
833
834 006006 005703          TR3:  TST   R3          ;PRINTOUT XMIT CSR?
835 006010 001407          BEQ   TRX          ;BR IF NO
836 006012 104400 010027   TYPE  MSG14        ; < XMIT CSR=>
837 006016 010346          MOV   R3,-(SP)      ;GET DATA
838 006020 004037 006514   JSR    RO,$B2016    ;AND PRINT IT
839 006024 104400 010266   TYPE  CRLF
840
841 006030 032777 100000 173724 TRX:  BIT   #BIT15, R3WR  ;HALT ON ERROR?
842 006036 001401          BEQ   HLTX         ;BR IF NO
843 006040 000000          HALT
844
845 006042 104412          HLTX: KBDIN
846 006044 012601          MOV   (SP)+, R1    ;RESTORE R1
847 006046 000002          RTI                ;AND RETURN TO PROGRAM
848
849 006050 010275          EMTAB: EMO
850 006052 010322          EM1
851 006054 010353          EM2
852 006056 010425          EM7
853 006060 010405          EM6
854 006062 010462          EM10
855
856
857
858
859
860
861 006064 013702 001730          SBAUD: MOV   PARAM2,R2          ;GET PARAM2
862 006070 042702 176077          BIC   #176077,R2    ;STRIP OFF ALL BUT SPEED
863 006074 006102          ROL   R2
864 006076 006102          ROL   R2
865 006100 006102          ROL   R2
866 006102 000302          SWAB  R2
867 006104 016237 001606 001572   MOV   BAUD(R2), $TYME ;SET TIMER DELAY
868 006112 000207          RTS   PC            ;EXIT
869
870

```

```

;*****
;CALCULATE TIMER DELAY
;*****

```

```

871      ;*****
872      ;      READ A CHAR. ROUTINE
873      ;*****
874
875      ;      CALL=  GETCHR  ;INPUT A CHAR FROM TTY
876      ;      ;RETURNS HERE WITH CHAR ON STACK
877      006114  011646  $READC:  MOV      (SP),-(SP)  ;PUSH THE PC
878      006116  016666  000004  000002  MOV      4(SP),2(SP)  ;SAVE THE PS
879      006124  105777  173660  TSTB    @TKS          ;IS RECEIVE DONE
880      006130  100375  BPL     -4            ;LOOP IF NO
881      006132  017737  173654  001544  MOV      @TKB, $CHAR  ;SAVE THE CHAR.
882      006140  042737  177600  001544  BIC     @177600,$CHAR ;STRIP JUNK
883      006146  013766  001544  000004  MOV      $CHAR,4(SP)  ;PUT CHAR ON STACK
884      006154  000002  RTI
885      ;*****
886      ;      READ A STRING ROUTINE
887      ;*****
888
889      ;      CALL=  GETSTR  ;INPUT A STRING OF CHARS FROM TTY
890      ;      ADDR  ;TO THIS ADDRESS
891      ;      ;TERMINATE INPUT WITH LINE FEED
892      006156  011602  $READS:  MOV      (SP), R2  ;SETUP ADDRESS OF INPUT BUFFER
893      006160  012201  MOV      (R2)+, R1      ;INCREMENT RETURN ADDRESS
894      006162  010216  MOV      R2, (SP)      ;AND PUT BACK ON STACK
895
896      ;      GETIC:  GETCHR  ;GET A CHAR
897      006164  104402  TYPE    $CHAR
898      006166  104400  001544  CMPB    @136, (SP)+  ;IS IT BINARY DELIMITER
899      006172  122726  000136  BNE     GOTIC        ;BR IF NO
900      006176  001011
901      ;      OCT:  ACCEPT  $WORK  ;GET OCT. CHAR
902      006200  104406  001546  MOVB   $WORK,(R1)+  ;STORE OCT. CHAR.
903      006204  113721  001546  CMPB   @136,$CHAR  ;TERMINATOR=BIN. DELIMITER?
904      006210  122737  000136  001544  BNE    OCT          ;BR IF NO
905      006216  001370
906      006220  000761
907      ;      GOTIC:  MOVB   $CHAR, (R1)+  ;STORE CHAR. IN BUFFER
908      006222  113721  001544  CMP    @15, $CHAR  ;IS IT END OF INPUT (CAR. RETURN)
909      006226  022737  000015  001544  BNE    GETIC       ;BR IF NO
910      006234  001353
911      006236  005737  001570  TST    FLAGS
912      006242  100401  BMI    IS
913      006244  105041  CLRB  -(R1)
914      006246  113711  001544  IS:    MOVB   $CHAR,(R1)
915      006252  104400  010266  TYPE  ,CRLF
916      006256  000002  RTI  ;TYPE A LINE FEED
    
```

```

917
918
919
920
921
922 006260
923 006260 010046
924 006262 010146
925 006264 010246
926 006266 010346
927 006270 016600 000010
928 006274 005001
929 006276 012702 000006
930 006302 104402
931 006304 112603
932 006306 110337 006454
933 006312 104400 006454
934 006316 022703 000025
935 006322 001451
936 006324 022703 000015
937 006330 001427
938 006332 022703 000040
939 006336 001433
940 006340 022703 000136
941 006344 001430
942 006346 032703 000110
943 006352 001011
944
945 006354 005302
946 006356 002407
947 006360 006301
948 006362 006301
949 006364 006301
950 006366 042703 177770
951 006372 050301
952 006374 000742
953 006376 104400 010256
954 006402 104400 010266
955 006406 000732
956 006410 104400 010267
957 006414 022702 000006
958 006420 001002
959 006422 013001
960 006424 005740
961 006426 010130
962 006430 010066 000010
963 006434 012603
964 006436 012602
965 006440 012601
966 006442 012600
967 006444 000002
968 006446 104400 010246
969 006452 000710
970 006454 000 000

```

```

*****
ROUTINE TO ACCEPT AN OCTAL NUMBER FROM THE TTY
CALL:
ACCEPT .ADDR ;PUT OCTAL NUMBER IN ADDR
$ACCEPT:
MOV R0,-(SP) ;SAVE R0
MOV R1,-(SP) ;SAVE R1
MOV R2,-(SP) ;SAVE R2
MOV R3,-(SP) ;SAVE R3
MOV 10(SP),R0 ;GET ADDRESS OF WHERE TO PUT NUMBER
1$: CLR R1 ;CLEAR PARTIAL NUMBER
MOV #6,R2 ;MAX. # OF DIGITS ALLOWED
2$: GETCHR ;GET ONE CHARACTER
MOV# (SP)+,R3 ;AND PUT IT IN R3
MOV# R3,6$ ;ECHO THE CHARACTER
CMP #25,R3
BEQ 8$
CMP #15,R3 ;WAS THIS CHARACTER A "CR"?
BEQ 5$ ;BR IF YES
CMP #40,R3 ;WAS "SPACE" HIT?
BEQ 7$ ;BR IF YES
CMP #136,R3 ;WAS "↑" HIT?
BEQ 7$ ;BR IF YES
BIT #110,R3 ;INSURE THE CHARACTER IS
BNE 4$ ;A DIGIT BETWEEN 0 AND 7.
DEC R2 ;CHECK NUMBER OF CHARACTERS
BLT 4$ ;BR IF TOO MANY
ASL R1 ;POSITION PARTIAL NUMBER
ASL R1 ;FOR THIS DIGIT
BIC #1C<7>,R3 ;GET RID OF THE ASCII JUNK
BIS R3,R1 ;COMBINE THIS DIGIT WITH PARTIAL
BR 2$ ;GO GET ANOTHER DIGIT
4$: TYPE ,QUES ;TYPE "?"
TYPE ,CRLF ;TYPE CARRAGE RETURN AND LINE FEED.
BR 1$ ;GO START OVER
5$: TYPE ,LF ;FOLLOW "CR" WITH A "LF"
CMP #6,R2 ;WERE ANY DIGITS INPUT
BNE .+6 ;BR IF YES
MOV @ (R0)+,R1 ;USE OLD DATA
TST -(R0) ;BACKUP R0--
7$: MOV R1,@ (R0)+ ;PASS THE NUMBER TO THE USER
MOV R0,10(SP) ;SET FOR RETURN
MOV (SP)+,R3 ;RESTORE R3
MOV (SP)+,R2 ;RESTORE R2
MOV (SP)+,R1 ;RESTORE R1
MOV (SP)+,R0 ;RESTORE R0
RTI
8$: TYPE, CTLU
BR 1$
6$: .BYTE 0,0 ;STORAGE FOR ASCII CHAR, AND TERMINATOR

```

```

971
972
973
974 006456 005337 006500
975 006462 001005
976 006464 012737 000060 006500
977 006472 005237 001746
978 006476
979 006476 000002
980 006500 000000
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002 006502 112037 006707
1003 006506 112037 006705
1004 006512 000406
1005 006514 112737 000001 006705
1006 006522 112737 000006 006707
1007 006530 112737 000005 006704
1008 006536 010346
1009 006540 010446
1010 006542 010546
1011 006544 113704 006707
1012 006550 005404
1013 006552 062704 000006
1014 006556 110437 006706
1015 006562 113704 006705
1016 006566 016605 000010
1017 006572 005003
1018 006574 006105
1019 006576 000404
1020 006600 006105
1021 006602 006105
1022 006604 006105
1023 006606 010503
1024 006610 006103
1025 006612 105337 006706
1026 006616 100016

;*****
;CLOCK INTERRUPT ROUTINE
;*****
TIMER: DEC MSECS ;COUNT 60 CYCLES
;BNE TIMEX ;BR IF NOT 60
MOV #60, MSECS ;RESTORE COUNT
INC TIME ;INCREMENT SECONDS
TIMEX:
RTI ;RETURN FROM INTERRUPT
MSECS: 0
;*****
;BINARY TO OCTAL (ASCII) AND TYPE
;SB20CT---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
CALL:
MOV NUM, -(SP) ;NUMBER TO BE TYPED
JSR RO, $B20CT ;CALL FOR TYPEOUT
;BYTE N ;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
;BYTE M ;M=1 OR 0
;1=TYPE LEADING ZEROS
;0=SUPPRESS LEADING ZEROS
;SB201----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST $BSOCT OR $B2016
CALL:
MOV NUM, -(SP)
JSR RO, $B201
;SB2016---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
CALL:
MOV NUM, -(SP)
JSR RO, $B2016
;SB20CT: MOVB (RO)+, $OMODE+1 ;PICKUP THE NUMBER OF DIGITS TO TYPE
MOVB (RO)+, $OFILL ;GET THE ZERO FILL SWITCH
BR $B201
;SB2016: MOVB #1, $OFILL ;SET THE ZERO FILL SWITCH
MOVB #6, $OMODE+1 ;SET FOR SIX(6) DIGITS
;SB201: MOVB #5, $OCNT ;SET THE ITERATION COUNT
MOV R3, -(SP) ;SAVE R3
MOV R4, -(SP) ;SAVE R4
MOV R5, -(SP) ;SAVE R5
MOVB $OMODE+1, R4 ;GET THE NUMBER OF DIGITS TO TYPE
NEG R4
;SUBTRACT IT FOR MAX. ALLOWED
ADD #6, R4 ;SAVE IT FOR USE
MOVB R4, $OMODE
MOVB $OFILL, R4 ;GET THE ZERO FILL SWITCH
MOV 10(SP), R5 ;PICKUP THE INPUT NUMBER
CLR R3 ;CLEAR THE OUTPUT WORD
ROL R5 ;ROTATE MSB INTO "C"
BR 3$ ;GO DO MSB
;FORM THIS DIGIT
2$: ROL R5
ROL R5
MOV R5, R3
3$: ROL R3 ;GET LSB OF THIS DIGIT
DECB $OMODE ;TYPE THIS DIGIT?
BPL 7$ ;BR IF NO

```

```

1027 006620 042703 177770
1028 006624 001002
1029 006626 005704
1030 006630 001403
1031 006632 005204
1032 006634 052703 000060
1033 006640 052703 000040
1034 006644 110337 006702
1035 006650 104400 006702
1036 006654 105337 006704
1037 006660 003347
1038 006662 002402
1039 006664 005204
1040 006666 000744
1041 006670 012605
1042 006672 012604
1043 006674 012603
1044 006676 012616
1045 006700 000200
1046 006702 000
1047 006703 000
1048 006704 000
1049 006705 000
1050 006706 000000
1051
1052
1053
1054
1055 006710 010046
1056 006712 016600 000002
1057 006716 005740
1058 006720 111000
1059 006722 016000 006730
1060 006726 000200
1061
1062
1063
1064
1065
1066 006730 001100
1067 006732 006114
1068 006734 006156
1069 006736 006260
1070 006740 007120
1071 006742 007010
1072 006744 007142
1073 006746 007026
1074 006748 007142
1075 006750 007026
1076 006752 007142
1077 006754 007026
1078 006756 007142
1079 006758 007026
1080 006760 007142
1081 006762 007026
1082 006764 007142
    
```

```

BIC #177770,R3
BNE 4S
TST R4
BEQ 5S
4S: INC R4
BIS #'0,R3
5S: BIS #' ,R3
MOV R3,8S
TYPE 8S
7S: DECB $OCNT
BGT 2S
BLT 6S
INC R4
BR 2S
6S: MOV (SP)+,R5
MOV (SP)+,R4
MOV (SP)+,R3
MOV (SP)+,(SP)
RTS R0
8S: .BYTE 0
.BYTE 0
$OCNT: .BYTE 0
$OFILL: .BYTE 0
$OMODE: 0
    
```

```

;GET RID OF JUNK
;TEST FOR 0
;SUPPRESS THIS 0?
;BR IF YES
;DON'T SUPPRESS ANYMORE 0'S
;MAKE THIS DIGIT ASCII
;MAKE ASCII IF NOT ALREADY
;SAVE FOR TYPING
;GO TYPE THIS DIGIT
;COUNT BY 1
;BR IF MORE TO DO
;BR IF DONE
;INSURE LAST DIGIT ISN'T A BLANK
;GO DO THE LAST DIGIT
;RESTORE R5
;RESTORE R4
;RESTORE R3
;SET THE STACK FOR RETURNING
;RETURN
;STORAGE FOR ASCII DIGIT
;TERMINATOR FOR TYPE ROUTINE
;OCTAL DIGIT COUNTER
;ZERO FILL SWITCH
;NUMBER OF DIGITS TO TYPE
    
```

```

;TRAP HANDLER
$TRAP: MOV R0,-(SP) ;SAVE R0
MOV 2(SP),R0 ;GET TRAP ADDRESS
TST -(R0) ;BACKUP BY 2
MOVB (R0),R0 ;GET RIGHT BYTE OF TRAP
MOV $TRPAD(R0),R0 ;INDEX TO TABLE
RTS R0 ;GO TO ROUTINE
    
```

```

;TRAP TABLE
; ROUTINE
; -----
;
    
```

```

$TRPAD: $TYPE
        TYPE=TRAP+0
        $READC GETCHR=TRAP+2
        $READS GETSTR=TRAP+4
        $ACCEPT ACCEPT=TRAP+6
        .STPS STPS=TRAP+10
        .KBDIN KBDIN=TRAP+12
        .SUSWR SUSWR=TRAP+14
        .SETSWI SETSWI=TRAP+16
    
```

```

1083
1084
1085
1086
1087
1088
1089
1090
1091 006750 012737 006762 000024 SPWRDN: MOV    #SPWRUP,2024
1092 006756 000000          HALT
1093 006760 000776          BR     .-2
1094
1095
1096
1097
1098
1099
1100
1101
1102 006762 012737 006750 000024 SPWRUP: MOV    #SPWRDN,2024
1103 006770 012706 001070          MOV    #STACK,SP
1104 006774 104400 010103          TYPE  MSG20 ;<15><12> POWER FAILED..
1105 007000 104410 000000          STPS,PRTY0
1106 007004 000137 000200          JMP    20200
1107
1108 007010 032737 000001 001760 .KBDIN: BIT    #BIT0,FLAG      ;TEST 1G FLAG
1109 007016 001437          BEQ    TOOUT                ;NO, EXIT
1110 007020 042777 000100 172762 .SETSWI: BIC    #100,2TKS     ;CLEAR TTY IE
1111 007026 104400 010227          TYPE  SWR=                 ;TYPE SWR=
1112 007032 017746 172724          MOV    #SWR,-(SP)         ;SET UP OCTAL TYPEOUT
1113 007036 004037 006514          JSR    RD,#B2016         ;DO IT
1114 007042 104400 010237          TYPE  NEG                 ;TYPE NEW=
1115 007046 017737 172710 001546          MOV    #SWR,WORK         ;SET UP FOR CR DEFAULT
1116 007054 104406 001546          ACCEPT WORK              ;GET VALUE
1117 007060 013777 001546 172674          MOV    WORK,#SWR        ;REPLACE IT
1118 007066 032737 000002 001760          BIT    #BIT1,FLAG       ;SEE HOW WE GOT HERE
1119 007074 001005          BNE    IS                 ;WRONG WAY?
1120 007076 005777 172710          TST   2TKB              ;CLEAR BUFFER
1121 007102 052777 000100 172700          BIS    #100,2TKS       ;RESET TTY IE
1122 007110 042737 000003 001760          BIC    #BIT0+BIT1,FLAG  ;CLEAR FLAG BITS
1123 007116 000002          TOOUT: RTI              ;EXIT
1124
1125 007120 042766 000340 000002 .STPS:  BIC    #PRTY7,2(SP) ;CLEAR OUT PRIORITY BITS
1126 007126 057666 000000 000002          BIS    2(SP),2(SP)      ;SET NEW PRIORITY
1127 007134 062716 000002          ADD   #2,(SP)           ;SETUP EXIT
1128 007140 000002          RTI                    ;EXIT
1129
1130 007142 013746 000006          .SUSWR: MOV    6,-(SP)    ;SAVE 6 ON STACK
1131 007146 013746 000004          MOV    4,-(SP)         ;SAVE 4 ON STACK
1132 007152 012737 007200 000004          MOV    #15,4          ;SETUP TIMEOUT
1133 007160 012737 000340 000006          MOV    #340,6         ;SET PRIORITY
1134 007166 022777 177777 172566          CMP    8-1,#SWR        ;TEST FOR 177570
1135 007174 001402          BEQ    25              ;NOT ALL 1'S
1136 007176 000407          BR     35              ;IT'S THERE - EXIT
1137 007200 022526          IS:    CMP    (SP)+,(SP)+ ;ADJUST STACK AFTER TRAP
1138 007202 012737 000176 001762          25:    MOV    #SWREG,SWR ;REPLACE HARDWARE REGISTERS

```

```

1139 007210 012737 000174 001764      MOV    #DISPREG,DISPLAY ;WITH SOFTWARE REGISTERS
1140 007216 012637 000004      3$:   MOV    (SP)+,4       ;RESTORE 4
1141 007222 012637 000006      MOV    (SP)+,6       ;RESTORE 6
1142 007226 000002      RTI                    ;EXIT

```

```

;*****
;RANDOM CHARACTER BUFFER GENERATOR
;*****

```

```

1150 007230 010546      SETBUF: MOV    R5,-(SP)      ;SAVE R5
1151 007232 010346      MOV    R3,-(SP)      ;SAVE R3
1152 007234 010046      MOV    R0,-(SP)      ;SAVE R0
1153 007236 005701      TST    R1            ;VARIABLE DATA?
1154 007240 001105      BNE    10$          ;YES,NOT RANDOM PATTERN
1155 007242 012703 003224      MOV    #DP1,R3      ;SETUP BUFFER ADDR
1156 007246 012705 000400      MOV    #256,R5      ;SETUP BUFF COUNT
1157 007252 010537 001650      MOV    R5,COUNT+2
1158 007256 005205      INC    R5            ;ADJUST COUNTER
1159 007260 004737 007322      1$:   JSR    PC,3$        ;GENERATE RANDOM CHAR
1160 007264 120027 000040      CMPB  RO,#40 ;LOW BYTE OVER 40
1161 007270 002773      BLT    1$           ;NO TRY AGAIN
1162 007272 120027 000177      CMPB  RO,#177 ;LOW BYTE UNDER 177
1163 007276 003370      BGT    1$           ;NO TRY AGAIN
1164 007300 005305      DEC    R5            ;DEC COUNT
1165 007302 001402      BEQ    2$           ;ALL DONE
1166 007304 110023      MOVB  RO,(R3)+      ;MOVE CHAR TO BUFF
1167 007306 000764      BR    1$            ;GET NEXT
1168 007310 105013      2$:   CLRB  (R3)         ;SET TERMINATOR TO ZERO
1169 007312 012600      MOV    (SP)+,R0     ;RESTORE R0
1170 007314 012603      MOV    (SP)+,R3     ;RESTORE R3
1171 007316 012605      MOV    (SP)+,R5     ;RESTORE R5
1172 007320 000207      RTS    PC
1173
1174 007322 010146      3$:   MOV    R1,-(SP)      ;SAVE R1
1175 007324 010346      MOV    R3,-(SP)      ;SAVE R3
1176 007326 010246      MOV    R2,-(SP)      ;SAVE R2
1177 007330 013700 001642      MOV    L,R0
1178 007334 013701 001644      MOV    H,R1
1179 007340 012703 000010      MOV    #10,R3
1180 007344 005002      CLR    R2
1181 007346 006302      4$:   ASL    R2
1182 007350 000241      CLC
1183 007352 006000      ROR    R0
1184 007354 006001      ROR    R1
1185 007356 103412      BCS    5$
1186 007360 042702 000001      BIC    #1,R2
1187 007364 042700 010000      BIC    #10000,R0
1188 007370 032700 002000      BIT    #2000,R0
1189 007374 001414      BEQ    6$
1190 007376 052700 010000      BIS    #10000,R0
1191 007402 000411      BR    6$
1192 007404 052702 000001      5$:   BIS    #1,R2
1193 007410 052700 010000      BIS    #10000,R0
1194 007414 032700 002000      BIT    #2000,R0

```

1195	007420	001402			BEQ	6\$	
1196	007422	042700	010000		BIC	#10000,R0	
1197	007426	005303		6\$:	DEC	R3	
1198	007430	001346			BNE	4\$	
1199	007432	010037	001642		MOV	R0,L	
1200	007436	010137	001644		MOV	R1,H	
1201	007442	010200			MOV	R2,R0	
1202	007444	012602			MOV	(SP)+,R2	;RESTORE R2
1203	007446	012603			MOV	(SP)+,R3	;RESTORE R3
1204	007450	012601			MOV	(SP)+,R1	;RESTORE R1
1205	007452	000207			RTS	PC	;EXIT
1206							
1207	007454	005005		10\$:	CLR	R5	
1208	007456	012703	003024		MOV	#VDB,R3	;SET BUFFER ADDRESS
1209	007462	105723		11\$:	TSTB	(R3)+	;CHECK FOR TERM CHAR
1210	007464	001402			BEQ	12\$	
1211	007466	005205			INC	R5	;INC BYTE COUNT
1212	007470	000774			BR	11\$;START OVER
1213	007472	010537	001650	12\$:	MOV	R5,COUNT+2	
1214	007476	000704			BR	2\$	
1215							
1216							

1217
1218
1219
1220
1221
1222
1223
1224
1225

AREA RESERVED FOR MOST ASCIZ MESSAGES.

007500	000040			MSG00:	.ASCIZ	/ /
007502	005015	052126	030466	MSG01:	.ASCIZ	<15><12>/VT61-T AND VT71-T LOOPBACK TEST/<15><12>
007546	005015	051511	020122	MSG02:	.ASCIZ	<15><12>/ISR NOT LOADED!!/
007571	015	041012	051525	MSG03:	.ASCIZ	<15><12>/BUS ADDRESS=/
007610	005015	042526	052103	MSG04:	.ASCIZ	<15><12>/VECTOR ADDRESS=/
007632	005015	051120	047511	MSG05:	.ASCIZ	<15><12>/PRIORITY=/
007646	005015	044514	042516	MSG06:	.ASCIZ	<15><12>/LINE # (OCTAL)=/
007670	005015	040520	040522	MSG07:	.ASCIZ	<15><12>/PARAMS #1=/
007705	015	040412	041523	MSG08:	.ASCIZ	<15><12>/ASCII PARAM=/
007724	005015	042523	020124	MSG09:	.ASCIZ	<15><12>/SET SWITCHES.../
007746	005015	047105	042524	MSG10:	.ASCIZ	<15><12>/ENTER DATA/<15><12>
007765	015	042412	042116	MSG11:	.ASCIZ	<15><12>/END OF PASS /
010004	040440	020124	047514	MSG12:	.ASCIZ	/ AT LOC /
010015	040	041522	020126	MSG13:	.ASCIZ	/ RCV CSR=/
010027	040	041530	051123	MSG14:	.ASCIZ	/ XCSR=/
010036	020440	020040	051503	MSG17:	.ASCIZ	/ ! CSR= /
010051	056	005015	000	MSG18:	.ASCIZ	./.<15><12>
010055	015	047012	020117	MSG19:	.ASCIZ	<15><12>/NO CLOCKS AVAILABLE/
010103	015	050012	053517	MSG20:	.ASCIZ	<15><12>/POWER FAILED../
010124	020073	040502	020104	MSG23:	.ASCIZ	/; BAD DATA=/
010140	020040	043440	047517	MSG24:	.ASCIZ	/ GOOD DATA=/
010156	005015	051124	047101	MSG26:	.ASCIZ	<15><12>/TRANSMITTED DATA=/<15><12>
010204	005015	042522	042503	MSG27:	.ASCIZ	<15><12>/RECEIVED DATA=/<15><12>
010227	015	051412	051127	SMEQ:	.ASCIZ	<15><12>/SMR= /
010237	040	042516	036527	NEQ:	.ASCIZ	/ NEW= /
010246	052536	005015	000	CTLU:	.ASCIZ	/↑U/<15><12>
010253	177	000177		MFILL:	.ASCIZ	<177><177>
010256	037440	000040		QUES:	.ASCIZ	" ? "
010262	005015	000052		ASTRK:	.ASCIZ	<15><12>"*"
010266	015			CRLF:	.ASCII	<15>
010267	012	000		LF:	.ASCIZ	<12>
010271	015	005012	000	TUCRLF:	.ASCIZ	<15><12><12>
010275	105	051122	051117	EM0:	.ASCIZ	"ERROR HALT ON LINE #"
010322	040527	052111	047111	EM1:	.ASCIZ	"WAITING TO RCV ON LINE #"
010353	127	044501	044524	EM2:	.ASCIZ	"WAITING TO XMIT ON LINE #"
010405	122	053103	041040	EM6:	.ASCIZ	"RCV BUFFER FULL"
010425	104	052101	020101	EM7:	.ASCIZ	"DATA COMPARE ERROR LINE(8) #"
010462	051105	047522	020122	EM10:	.ASCIZ	"ERROR RCV CSR=CONTENTS OF SELECT 0 REGISTER"
010537	015	052012	042510	MSG0:	.ASCIZ	<15><12>/THERE WAS A RECEIVER ERROR. REGISTER (SEL 2) =/
010620	005015	000		MSG1:	.ASCIZ	<15><12>
010623	015	051012	041505	MSG2:	.ASCIZ	<15><12>/RECEIVED DATA = /<15><12>
010650	005015	040504	040524	MSG3:	.ASCIZ	<15><12>/DATA SHOULD BE/<15><12>
010673	015	042412	051122	MFULL:	.ASCIZ	<15><12>/ERROR! RECEIVER BUFFER FULL/
010731	015	042412	051122	SILO:	.ASCIZ	<15><12>/ERROR! SILO OVERFLOW/
010760	005015	047516	020116	NONEX:	.ASCIZ	<15><12>/NON EXISTENT MEMORY ERROR/
011014	005015	042524	052123	TESTXX:	.ASCIZ	<15><12>/TESTING LINE (8) #/
011040	000003			CTLC:	.ASCIZ	<003>
				.EVEN		

```

1226
1227
1228
1229
1230
1231 011042 000240
1232 011044 017700 170712
1233 011050 042700 177400
1234 011054 013702 001722
1235 011060 012722 012412
1236 011064 013722 001724
1237 011070 012722 012200
1238 011074 013722 001724
1239 011100 013704 001720
1240 011104 005737 001726
1241 011110 001001
1242 011112 000000
1243 011114 012714 004000
1244 011120 004737 012560
1245
1246
1247 011124 005037 001746
1248 011130 005037 001670
1249 011134 005037 001674
1250
1251 011140 104412
1252 011142 013737 001650 001646
1253 011150 013705 001650
1254 011154 032737 040000 001760
1255 011162 001403
1256 011164 012737 000004 001646
1257 011172 004737 012324
1258 011176 004737 012110
1259 011202 005037 001746
1260 011206 032700 100000
1261 011212 001016
1262 011214 032700 040000
1263 011220 001024
1264 011222 023737 001746 001572
1265 011230 103766
1266 011232 011402
1267 011234 016403 000000
1268 011240 104001
1269 011242 005037 001746
1270 011246 000757
1271 011250 032737 040000 001730
1272 011256 001756
1273 011260 042700 100000
1274 011264 004737 012324
1275 011270 000746
1276 011272 032700 100000
1277 011276 001013
1278 011300 023737 001746 001572
1279 011306 103771
1280 011310 011402

;*****
; DH11-X INTERFACE SERVICE ROUTINE
;*****
START:  NOP
        MOV     @SWR, R0      ;SETUP MODE IN R0
        BIC     #177400, R0   ;RO ;STRIP JUNK
        MOV     VA, R2        ;SETUP
        MOV     @RISR, (R2)+  ;INTERRUPT
        MOV     PRIOR, (R2)+ ;VECTORS
        MOV     @XISR, (R2)+
        MOV     PRIOR, (R2)+
        MOV     BA, R4        ;SETUP BUS ADDR INDEX
        TST     PARAM1
        BNE     4$
        HALT
4$:     MOV     @MC,@RCSR
        JSR     PC,@NEEXT
GO:     CLR     TIME
        CLR     DELAY
        CLR     STOP
$XLB:   KBDIN
        MOV     COUNT+2,COUNT
        MOV     COUNT+2,R5
        BIT     @ESC,FLAG
        BEQ     1$
        MOV     #4,COUNT
        JSR     PC,STARTR
        JSR     PC,STARTX
        CLR     TIME
2$:     BIT     @XFLG,STAT
        BNE     3$
7$:     BIT     @RFLG,STAT
        BNE     4$
        CMP     TIME,$TYME
        BLO     2$
        MOV     @RCSR,R2
        MOV     XCSR(R4),R3
        HLT     1
        CLR     TIME
        BR     2$
3$:     BIT     @HALF.DUPLEX,PARAM2
        BEQ     7$
        BIC     @XFLG,STAT
        JSR     PC,STARTR
        BR     2$
4$:     BIT     @XFLG,STAT
        BNE     6$
        CMP     TIME,$TYME
        BLO     4$
        MOV     @RCSR,R2
; ERROR MESSAGE HERE COULD IMPLY THAT
; TERMINAL IS OFFLINE-MESSAGE NOT
; LOOPBACKED BEFORE TIMER EXPIRED

```

```

1281 011312 016403 000000      MOV      XCSR(R4),R3
1282 011316 104002      HLT      2 ; COULD BE PROBLEM WITH DH-MESSAGE
1283 011320 005037 001746      CLR     TIME ; NOT SENT BEFORE TIMER EXPIRED
1284 011324 000762      BR      4$
1285 011326 042700 100000      6$: BIC     #XFLG,STAT
1286 011332 042700 040000      8$: BIC     #RFLG,STAT
1287 011336 005037 001746      CLR     TIME
1288 011342 032777 000200 170412      BIT     #NODAT,@SWR
1289 011350 001002      BNE     5$
1290 011352 004737 011522      JSR     PC,TESTD
1291 011356 012737 011140 001672      5$: MOV     #5$LB,BACK
1292
1293
1294 011364      EOP:
1294 011364 104410 000340      STPS,PRTY7 ; SET PS PRIORITY TO 7
1295 011370 016437 000000 001652      MOV     XCSR(R4),@TPIE ; SAVE TX CSR
1296 011376 042737 157777 001652      BIC     #1<TIE>,@TPIE ; CLEAR ALL BUT TX IE.
1297 011404 042764 020000 000000      BIC     #TIE,XCSR(R4) ; CLEAR TX IE (EVEN IF IT WASN'T SET)
1298 011412 012766 011452 000002      MOV     #ENTER,2(SP) ; SET FOR RETURN IF SW 14=1
1299 011420 010037 001654      MOV     R0,SAVR0 ; SAVE REGISTER 0
1300 011424 010137 001656      MOV     R1,SAVR1 ; SAVE REGISTER 1
1301 011430 010237 001660      MOV     R2,SAVR2 ; SAVE REGISTER 2
1302 011434 010337 001662      MOV     R3,SAVR3 ; SAVE REGISTER 3
1303 011440 010437 001664      MOV     R4,SAVR4 ; SAVE REGISTER 4
1304 011444 010537 001666      MOV     R5,SAVR5 ; SAVE REGISTER 5
1305 011450 000207      RTS     PC ; RETURN TO CONTROL PROGRAM
1306
1307 011452      ENTER:
1308 011452 013700 001654      MOV     SAVR0,R0 ; RESTORE R0
1309 011456 013701 001656      MOV     SAVR1,R1 ; RESTORE R1
1310 011462 013702 001660      MOV     SAVR2,R2 ; RESTORE R2
1311 011466 013703 001662      MOV     SAVR3,R3 ; RESTORE R3
1312 011472 013704 001664      MOV     SAVR4,R4 ; RESTORE R4
1313 011476 013705 001666      MOV     SAVR5,R5 ; RESTORE R5
1314 011502 012737 177777 001670      MOV     #-1,DELAY
1315 011510 053764 001652 000000      BIS     @TPIE,XCSR(R4) ; IF ORIGINALLY SET; SET TX IE
1316 011516 000177 170150      JMP     @BACK
1317
1318
1319
1320
1321
1322
1323
1324 011522 013746 001774      TESTD: MOV     ERDR, -(SP) ; WAS THERE A RECEIVE ERROR?
1325 011526 001413      BEQ     TSTDAT ; BR IF NO
1326 011530 032777 020000 170224      BIT     #BIT13,@SWR ; INHIBIT PRINTOUTS?
1327 011536 001007      BNE     TSTDAT ; BR IF YES
1328 011540 104400 010537      TYPE   MSG0 ; <15><12>THERE WAS A RECEIVE ERROR. RBUF=
1329 011544 004077 170174      JSR     R0,@B2016 ; PRINT CONTENTS OF RBUF
1330 011550 005746      TST    -(SP)
1331 011552 104400 010620      TYPE   MSG1 ; <15><12>
1332 011556 032737 040000 001760      TSTDAT: BIT     #ESC,FLAG ; ESC MODE?
1333 011564 001405      BEQ     1$ ; NO
1334 011566 012701 004224      MOV     #LPB,R1
1335 011572 012705 000004      MOV     #4,R5 ; SET COUNT
1336 011576 000404      BR      2$
    
```

VT61/T LOOPBACK TEST MACY11 27(1006) 02-MAR-77 16:44 PAGE 31
 DZVTLA.P11 02-MAR-77 16:43

```

1337 011600 013701 001736 1$: MOV IXDA, R1 ; SETUP XMIT DATA ADDR
1338 011604 013705 001650 MOV COUNT+2, R5 ; SETUP COUNT
1339 011610 005205 2$: INC R5 ; ADJUST COUNT
1340 011612 005037 001676 CLR $ERROR ; CLEAR ERROR CNT
1341 011616 013702 001734 MOV IRDA, R2 ; SETUP RCV DATA ADDR
1342 011622 005305 SCAN4: DEC R5 ; DEC COUNT
1343 011624 001422 BEQ TESTDX ; ALL DONE
1344 011626 122122 CMPB (R1)+, (R2)+ ; DATA OK
1345 011630 001774 BEQ SCAN4 ; YES NEXT CHAR PLEASE
1346 011632 032777 000010 170122 BIT #FSTERR, $SWR ; PRINT JUST FIRST ERROR
1347 011640 001403 BEQ 1$ ; NO
1348 011642 005737 001676 TST $ERROR ; IS THIS THE FIRST
1349 011646 001365 BNE SCAN4 ; NO
1350 011650 005237 001676 1$: INC $ERROR ; INC ERROR CNT
1351 011654 010246 MOV R2, -(SP) ; SAVE R2
1352 011656 114103 MOVB -(R1), R3
1353 011660 114202 MOVB -(R2), R2
1354 011662 104003 HLT 3
1355 011664 012602 MOV (SP)+, R2 ; RESTORE R2
1356 011666 105721 TSTB (R1)+ ; ADJUST POINTER
1357 011670 000754 BR SCAN4
1358 011672 005726 TESTDX: TST (SP)+ ; POP STACK
1359 011674 042737 040000 001760 BIC #ESC, FLAG
1360 011702 000207 RTS ; RETURN FROM SUB/ROUT
1361
1362
1363 : ROUTINE TO INPUT LINE NUMBERS TO TEST
1364 011704 104400 007646 GETPRM: TYPE MSG06 ; ASK FOR LINE NUMBERS
1365 011710 005010 CLR (R0) ; CLEAR LINE #
1366 011712 052737 100000 001570 BIS #BIT15, FLAG
1367 011720 104404 GETSTR
1368 011722 002024 IBUF ; STORE THEN AT IBUF
1369 011724 010146 MOV R1, -(SP)
1370 011726 010246 MOV R2, -(SP)
1371 011730 010346 MOV R3, -(SP)
1372 011732 010446 MOV R4, -(SP)
1373 011734 012701 002024 MOV #IBUF, R1 ; SET UP POINTERS
1374 011740 012702 001546 9$: MOV #WORK, R2
1375 011744 020227 001550 1$: CMP R2, #WORK+2 ; HAVE TWO CHARS YET
1376 011750 001002 BNE 4$
1377 011752 000342 SWAB -(R2)
1378 011754 000414 BR 5$
1379 011756 112122 4$: MOVB (R1)+, (R2)+ ; GET A CHARACTER
1380 011760 124227 000015 CMPB -(R2), #15 ; <CR>?
1381 011764 001443 BEQ 8$
1382 011766 123727 001546 000054 CMPB WORK, #54 ; IS 1ST BYTE A SPACE
1383 011774 001761 BEQ 9$
1384 011776 122227 000054 7$: CMPB (R2)+, #54 ; SPACE DELIMITER?
1385 012002 001360 BNE 1$ ; RECYCLE
1386 012004 105042 CLRB -(R2)
1387 012006 012702 001546 5$: MOV #WORK, R2 ; PROCESS LINE NUMBER
1388 012012 042712 177370 BIC #177370, (R2) ; CLEAR JUNK
1389 012016 032712 000400 BIT #400, (R2) ; LINE BETWEEN 10-17?
1390 012022 001402 BEQ 2$ ; NO
1391 012024 052712 000010 BIS #10, (R2) ; ELSE ADD 10 TO LOW BYTE
1392 012030 005003 2$: CLR R3

```

```

1393 012032 105212          INCB      (R2)
1394 012034 000261          SEC
1395 012036 006103          3$:     ROL      R3          ;SET BIT IN PARAM1 FOR LINE #
1396 012040 000241          CLC
1397 012042 105312          DECB     (R2)
1398 012044 001374          BNE      3$
1399 012046 050310          BIS      R3,(R0)          ;DO IT HERE
1400 012050 000735          BR       1$              ;RECYCLE
1401 012052 005720          11$:    TST      (R0)+          ;BUMP THE POINTER BY 2
1402 012054 012604          MOV      (SP)+,R4
1403 012056 012603          MOV      (SP)+,R3
1404 012060 012602          MOV      (SP)+,R2
1405 012062 012601          MOV      (SP)+,R1
1406 012064 042737 100000 001570      BIC      #BIT15,FLAGS
1407 012072 000207          RETURN
1408 012074 123727 001546 000015 8$:    CMPB     WORK,#15          ;EXIT
1409 012102 001763          BEQ      11$
1410 012104 105012          CLRB     (R2)
1411 012106 000737          BR       5$
1412

```

```

1413
1414
1415
1416
1417
1418 012110 032737 040000 001760 STARTX: BIT      #ESC,FLAG      ;ESCAPE MODE YET
1419 012116 001407          BEQ      7$
1420 012120 012764 004224 000006          MOV      #LPB,CAR(R4)
1421 012126 012764 177774 000010          MOV      #-4,BCR(R4)
1422 012134 000411          BR       8$
1423 012136 013737 001736 002006 7$:      MOV      IXDA,XDA
1424 012144 013764 002006 000006          MOV      XDA,CAR(R4)      ;LOAD CURRENT ADDRESS REG
1425 012152 012764 177777 000010          MOV      #-1,BCR(R4)      ;LOAD BYTE COUNT REG
1426 012160 042700 100000          BIC      #XFLG,STAT      ;CLEAR XMIT DONE FLAG
1427 012164 052714 020000          BIS      #TIE,ARCSR      ;SET INTERRUPT ENABLE
1428 012170 013764 001704 000012          MOV      BARTMP,BAR(R4)   ;LOAD BAR REG
1429 012176 000207          RTS      PC
1430
1431 012200 042714 100000          XISR:   BIC      #TI,ARCSR      ;CLEAR XMIT DONE
1432 012204 032714 002000          BIT      #NEM,ARCSR      ;NON-EXISTENT MEM ERROR?
1433 012210 001407          BEQ      1$
1434 012212 011402          MOV      ARCSR,R2        ;SAVE CSR FOR TYPE OUT
1435 012214 005003          CLR      R3
1436 012216 104005          HLT      5
1437 012220 104400 010760          TYPE    ,NONEX          ;TYPE ERROR MESS
1438 012224 000000          HALT
1439 012226 000776          BR       -2              ;BR HALT
1440 012230 032737 040000 001760 1$:      BIT      #ESC,FLAG
1441 012236 001002          BNE
1442 012240 005305          DEC      R5
1443 012242 001005          BNE
1444 012244 052700 100000          4$:      BIS      #XFLG,STAT      ;SET XMIT DONE FLAG
1445 012250 042714 020000          BIC      #TIE,ARCSR      ;CLEAR INTERRUPT ENABLE
1446 012254 000416          3$:      BR       XISR2
1447 012256 005237 002006          NOXMON: INC      XDA            ;INC TXBUF POINTER
1448 012262 013764 002006 000006          MOV      XDA,CAR(R4)      ;LOAD CURRENT ADDRESS REG
1449 012270 005737 001710          TST      TRNFLG          ;IS THIS FIRST TIME?
1450 012274 001006          BNE      XISR2          ;BR IF YES
1451 012276 012764 177777 000010          MOV      #-1,BCR(R4)      ;LOAD BYTE COUNT REG
1452 012304 013764 001704 000012          MOV      BARTMP,BAR(R4)   ;SET BAR BIT
1453 012312 005037 001746          XISR2: CLR      TIME
1454 012316 005037 001710          CLR      TRNFLG
1455 012322 000002          RTI
1456
1457
1458
1459
1460 012324 032737 040000 001730 STARTR: BIT      #HALF.DUPLEX,PARAM2      ;HALF DUPLEX?
1461 012332 001005          BNE      2$
1462 012334 005037 001700          CLR      TEMP1          ;BR IF YES
1463 012340 005237 001700          1$:      INC      TEMP1          ;START DELAY
1464 012344 001375          BNE      1$
1465 012346 042700 040000          2$:      BIC      #RFLG,STAT      ;CLEAR RFLG
1466 012352 013737 001734 002004          MOV      IRDA,RDA        ;SET UP RECEIVER DATA ADD
1467 012360 012737 001000 002002          MOV      #1000,RCC       ;SET UP BUFFER LIMIT
1468 012366 012737 177777 001706          MOV      #-1,SACFLG      ;SET SOFTWARE FLAG

```

1469	012374	005037	001772		CLR	ERCSR		;CLEAR ERROR RECORDS
1470	012400	005037	001774		CLR	ERDBR		
1471	012404	052714	010100		BIS	#RIE+SIE, @RCSR		;SET INTERRUPT ENABLES
1472	012410	000207			RTS	PC		
1473								
1474	012412	032714	040000		RISR:	BIT	#SI, @RCSR	;SILO OVERFLOW?
1475	012416	001407				BEQ	1\$;BR IF NO
1476	012420	011402				MOV	@RCSR, R2	;SAVE CSR FOR TYPEOUT
1477	012422	005003				CLR	R3	
1478	012424	104005				HLT	5	;ERROR HLT
1479	012426	104400	010731			TYPE	, SILO	;TYPE ERROR MESS
1480	012432	000000				HALT		
1481	012434	000776				BR	.-2	;BR HALT
1482	012436	016401	000002		1\$:	MOV	MRCR(R4), R1	;PUT CHAR IN R1
1483	012442	042701	000200			BIC	#200, R1	;STRIP A BIT
1484	012446	005701				TST	R1	;VALID DATA?
1485	012450	100403				BMI	4\$;BR IF YES
1486	012452	011402				MOV	@RCSR, R2	;SAVE CSR FOR TYPEOUT
1487	012454	005003				CLR	R3	
1488	012456	104005				HLT	5	;ERROR HLT
1489	012460	032701	070000		4\$:	BIT	#DO+FE+PE, R1	;OVERRUN, FRAMING OR PARITY ERROR?
1490	012464	001404				BEQ	3\$;BR IF NO
1491	012466	011437	001772			MOV	@RCSR, ERCSR	;SAVE CSR
1492	012472	010137	001774			MOV	R1, ERDBR	;SAVE CHAR
1493	012476	110177	167302		3\$:	MOVB	R1, @RDA	;STORE CHAR IN BUFFER
1494	012502	005237	002004		2\$:	INC	RDA	;INC RECEIVER BUFFER POINTER
1495	012506	105077	167272			CLRB	@RDA	;CLEAR NEXT LOCATION
1496	012512	005337	002002			DEC	RCC	;DEC CHAR COUNT
1497	012516	001004				BNE	5\$;BR IF BUFFER NOT FULL
1498	012520	000005				RESET		
1499	012522	104004				HLT	4	
1500	012524	000000				HALT		
1501	012526	000776				BR	.-2	;BR HALT
1502	012530	005337	001646		5\$:	DEC	COUNT	
1503	012534	001004				BNE	6\$	
1504	012536	042714	010100		8\$:	BIC	#RIE+SIE, @RCSR	;CLEAR INTERRUPT ENABLES
1505	012542	052700	040000			BIS	#RFLG, STAT	;SET RCV DONE FLAG
1506	012546	005037	001746		6\$:	CLP	TIME	
1507	012552	005037	001706			CLR	SNCFLG	;CLEAR FLAG
1508	012556	000002				RTI		
1509								
1510								
1511	012560	105737	001565		NEEXT:	TSTB	LINE+1	;FIRST TIME THRU
1512	012564	001010				BNE	1\$;NO BR
1513	012566	105137	001565			COMB	LINE+1	;YES SET TO NONZERO
1514	012572	012737	000001	001566		MOV	#1, LINMSK	;SET FOR LINE 0
1515	012600	105037	001564			CLRB	LINE	;CLEAR OUT LINE #
1516	012604	000405				BR	2\$	
1517	012606	105237	001564		1\$:	INCB	LINE	;NEXT?
1518	012612	006337	001566			ASL	LINMSK	;SHIFT FOR NEXT LINE
1519	012616	001405				BEQ	3\$;ALL DONE EXIT
1520	012620	033737	001566	001726	2\$:	BIT	LINMSK, PARAM1	;THIS LINE?
1521	012626	001767				BEQ	1\$;NO
1522	012630	000410				BR	4\$;PROCEED
1523	012632	005037	001564		3\$:	CLR	LINE	;INIT TAGS
1524	012636	052737	000100	001760		BIS	#BIT6, FLAG	;SET ONCE THRU FLAG

1525	012644	162716	000004			SUB	#4,(SP)	;ADJUST STACK TO CALL SELINE
1526	012650	000420				BR	SS	;DO IT
1527	012652	052714	004000		4\$:	BIS	#MC,ARCSR	;CLEAR OUT LINE #
1528	012656	153714	001564			BISB	LINE,ARCSR	;SET NEW LINE #
1529	012662	053764	001730	000004		BIS	PARAM2,LPR(R4)	;SET PARAMS
1530	012670	013737	001566	001704		MOV	LINMSK,BARTMP	;SET BAR BITS
1531	012676	005737	001576			TST	\$PASS	;FIRST PASS
1532	012702	001003				BNE	SS	;IF NOT
1533	012704	052737	040000	001760		BIS	#ESC,FLAG	;SET ESCAPE FLAG
1534	012712	000207				RTS	PC	;EXIT
1535		000001						

5\$:
.END

MSG0	010537	1225#	1328				
MSG00	007500	444	454	1225#			
MSG01	007502	592	1225#				
MSG02	007546	1225#					
MSG03	007571	596	1225#				
MSG04	007610	599	1225#				
MSG05	007632	602	1225#				
MSG06	007646	1225#	1364				
MSG07	007670	607	1225#				
MSG08	007705	1225#					
MSG09	007724	623	626	1225#			
MSG1	010620	1225#	1331				
MSG10	007746	650	1225#				
MSG11	007765	727	1225#				
MSG12	010004	801	1225#				
MSG13	010015	830	1225#				
MSG14	010027	836	1225#				
MSG17	010036	1225#					
MSG18	010051	1225#					
MSG19	010055	691	1225#				
MSG2	010623	1225#					
MSG20	010103	1104	1225#				
MSG23	010124	812	1225#				
MSG24	010140	819	1225#				
MSG26	010156	733	1225#				
MSG27	010204	737	1225#				
MSG3	010650	1225#					
NEEXT	012560	571	718	1244	1511#		
NEH =	002000	304#	1432				
NEQ	C10237	1114	1225#				
NODAT =	000200	288#	1288				
NOLC	005122	672	679#				
NONEX	010760	1225#	1437				
NORTC	005224	675	686	694#			
NOXPION	012256	1443	1447#				
NRCR =	000002	290#	1482				
NULL	001574	382	483#				
OCT	006200	901#	904				
PARAM1	001726	529#	1240	1520			
PARAM2	001730	530#	861	1271	1460	1529	
PARAM3	001732	531#					
PE =	010000	307#	1489				
PRIOR	001724	528#	1236	1238			
PRTY0 =	000000	249#	688	698	754	1105	
PRTY1 =	000040	250#					
PRTY2 =	000100	251#					
PRTY3 =	000140	252#	571				
PRTY4 =	000200	253#					
PRTY5 =	000240	254#					
PRTY6 =	000300	255#					
PRTY7 =	000340	256#	577	1125	1294		
PRVFC =	000024	321#					
QPIE	001652	506#	1295#	1296#	1315		
QUES	010256	419	771	953	1225#		
RCC	002002	555#	1467#	1496#			
RDA	002004	556#	1466#	1493#	1494#	1495#	

E04

VT61/T LOOPBACK TEST MACY11 27(1006) 02-MAR-77 16:44 PAGE 44
DZVTLA.P11 02-MAR-77 16:43 CROSS REFERENCE TABLE -- MACRO NAMES

BOX	326#	395	466	571	572	586	634	647	708	766	777	857	871	885	971
HLT	1146 205#	1268	1282	1354	1436	1478	1488	1499							

. ABS. 012714 000

ERRORS DETECTED: 0
DEFAULT GLOBALS GENERATED: 0

DZVTLA, DZVTLA+DZVTLA
RUN-TIME: 13.5 SECONDS
RUN-TIME RATIO: 29/5=5.4
CORE USED: 7K (14 PAGES)