

RP11C,RP03

MULTIDRIVE DRAG
CZRPCD0

AH-9283D-MC

COPYRIGHT © 72-73

FICHE 1 OF 1

DEC 1978

digital

MADE IN USA

.REM %

IDENTIFICATION

PRODUCT CODE: AC-9272D-MC
PRODUCT NAME: CZRPCDO RP11C MULTI DRIVE DIAGNOSTIC
DATE RELEASED: 1-FEBRUARY-1978
MAINTAINER: DIAGNOSTIC GROUP
AUTHOR: JOE STUBBLEBINE

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

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

COPYRIGHT (C) 1972,1978 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DIGITAL	PDP	UNIBUS	MASSBUS
DEC	DECUS	DECTAPE	

CONTENTS

- 1.0 ABSTRACT
- 2.0 REQUIREMENTS
- 2.1 EQUIPMENT
- 2.2 STORAGE
- 2.3 PRELIMINARY PROGRAMS
- 3.0 LOADING PROCEDURE
- 4.0 STARTING PROCEDURE
- 4.1 CONTROL SWITCH SETTINGS
- 4.2 STARTING ADDRESS
- 4.3 PROGRAM AND/OR OPERATOR ACTION
- 5.0 OPERATING PROCEDURE
- 5.1 OPERATION SWITCH SETTINGS
- 5.2 SUBROUTINE ABSTRACT
- 6.0 ERRORS
- 7.0 RESTRICTIONS
- 8.0 MISCELLANEOUS
- 8.1 EXECUTION TIME
- 8.2 STACK POINTER
- 8.3 ERROR INFORMATION
- 9.0 PROGRAM DESCRIPTION

1. ABSTRACT

THIS PROGRAM WILL TEST UP TO EIGHT RP02/RP03 DRIVES ON AN RP11C DISK CONTROLLER. BASICALLY, THE PROGRAM WILL SEEK TO A RANDOM ADDRESS AND THEN WRITE AND READ RANDOM DATA. WHILE DATA IS BEING TRANSFERRED, SEEK OPERATIONS WILL BE IN PROGRESS ON THE OTHER DRIVES. THE PURPOSE OF THE TEST IS TO CHECK FOR ANY INTERACTION ON THE BUS WHILE TRYING TO KEEP ALL THE DRIVES BUSY.

2. REQUIREMENTS

2.1 EQUIPMENT

PDP-11 STANDARD FAMILY PROCESSOR
RP11C DISK CONTROLLER WITH UP TO EIGHT RP02/RP03
DISK DRIVES

2.2 STORAGE

4K OF STORAGE IS REQUIRED TO RUN THIS TEST

2.3 PRELIMINARY PROGRAMS

DZRPA DISKLESS DIAGNOSTIC
DZRPB DISK RELIABILITY DIAGNOSTIC

3. LOADING PROCEDURE

USE STANDARD PROCEDURE FOR ABS TAPES.

4. STARTING PROCEDURE

4.1 CONTROL SWITCH SETTINGS

SEE 5.1.1

4.2 STARTING ADDRESS

THE PROGRAM SHOULD ALWAYS BE STARTED AT 200.

4.3 PROGRAM AND/OR OPERATOR ACTION

1. LOAD PROGRAM INTO MEMORY USING ABS LOADER
2. LOAD ADDRESS 200
3. SET SWITCHES (SEE SEC. 5.1.1)
4. PRESS START.
5. THE PROGRAM WILL LOOP UNTIL STOPPED
6. DUE TO THE RANDOM NATURE OF THE PROGRAM THERE IS NO MEANINGFULL PASSCOUNT. IT IS RECOMMENDED THAT THE PROGRAM RUN AT LEAST HALF AN HOUR.
(NOTE: THE FIRST PASS, A 'QUICK VERIFY' PASS, PERFORMS EACH OF THE FOUR TEST FUNCTIONS ON EACH DRIVE ONLY ONCE. THIS IS TO ENSURE THAT THE DRIVES ARE BASICALLY 'THERE' AND RUNNING.)

5. OPERATING PROCEDURE

5.1 OPERATIONAL SWITCH SETTINGS

AT STARTING ADDRESS 200, THE SETTING OF THE SWITCHES WILL DETERMINE WHICH UNITS ARE TO BE TESTED.

5.1.1 SWITCH SETTING ARE:

SW<15>=1.....HALF ON ERROR
SW<14>NOT USED
SW<13>=1.....INHIBIT PRINTOUT
SW<12>NOT USED
SW<11>NOT USED
SW<10>=1.....BELL ON ERROR
SW<07> THRU SW<00>=1.....SELECT UNIT FOR TEST

SW<00> CORRESPONDS TO UNIT 0
SW<07> CORRESPONDS TO UNIT 7

5.2 SUBROUTINE ABSTRACTS

5.2.1 HLT

THIS ROUTINE IS ENTERED UPON DETECTION OF AN ERROR. IT WILL TYPE THE PC OF THE ERROR AND ADDITIONAL ERROR INFORMATION. THIS ROUTINE TESTS FOR, HALT ON ERROR, INHIBIT TYPEOUTS, AND RINGS THE BELL.

5.2.2 TRAP CATCHER

A ".+2" - "HALT" SEQUENCE IS REPEATED FROM 0-776 TO CATCH ANY UNEXPECTED TRAPS. THESE UNEXPECTED TRAPS OR INTERRUPTS WILL HALT AT THE VECTOR +2.

6.0 ERRORS

6.1 WHEN ERRORS ARE ENCOUNTERED, THE ADDRESS OF THE ERROR ALONG WITH THE CONTENTS OF RPDS, RPER, AND RPCS ARE TYPED. ALSO, THE CONTENTS OF THE SELECTED CYLINDER, HEAD AND SECTOR ADDRESS ARE TYPED. BY REFERRING TO THE LISTING, ADDITIONAL INFORMATION CAN BE FOUND REGARDING THE CAUSE OF THE ERROR IN THE COMMENT FIELD. WHEN APPROPRIATE, ADDITIONAL INFORMATION IS TYPED OUT, SUCH AS THE EXPECTED AND RECEIVED RESULTS OF AN OPERATION. ALL INFORMATION IS IN OCTAL.

ERROR MESSAGE FORMAT

1. PC= ADDRESS OF FAILURE
UNIT UNIT WHICH FAILED
RPDS= CONTENTS OF RPDS
RPER= CONTENTS OF RPER
RPCS= CONTENTS OF RPCS
CYLINDER SELECTED CYLINDER
HEAD SELECTED HEAD
SECTOR SELECTED SECTOR
EXPECTED EXPECTED DATA
RECEIVED RECEIVED DATA

7.0 RESTRICTIONS

SINCE THIS IS AN INTERACTION TEST, THERE IS NO LOOPING ON ERRORS.

8.0 MISCELLANEOUS

8.1 EXECUTION TIME

DUE TO THE RANDOM NATURE OF THE PROGRAM THERE IS NO MEANINGFUL PASS COUNT. IT IS RECOMMENDED THAT THE PROGRAM SHOULD RUN FOR HALF AN HOUR.
(SEE NOTE IN 4.3 FOR EXPLANATION OF QV PASS)

8.2 STACK POINTER

STACK IS INITIALLY SET TO 500.

8.3 ERROR INFORMATION

IF IT IS DESIRED TO HAVE THE ERROR INFORMATION OUTPUTTED TO THE PUNCH INSTEAD OF THE TELETYPE CHANGE THE FOLLOWING THREE LOCATIONS.

LOCATION	FROM	TO
1304	177564	177554
1332	177566	177556
1336	177564	177554

9.0 PROGRAM DESCRIPTION

WHEN STARTED THE PROGRAM WILL RESTORE THE HEADS FOR EACH OF THE SELECTED UNITS. THEN THE FOLLOWING SEQUENCE IS GONE THRU FOR EACH OF THE SELECTED UNITS. FIRST, A RANDOM DISK SURFACE ADDRESS IS GENERATED AND A SEEK IS ISSUED. THEN A RANDOM BUFFER IS SELECTED AND FILLED WITH RANDOM DATA. A SECTOR IS THEN WRITTEN, READ BACK AND COMPARED. THIS SEQUENCE IS THEN LOOPED UPON. DUE TO THE DIFFERENCE IN SEEK TIMES, WHICH DEPENDS ON THE RANDOM DISK ADDRESS SELECTED, ALL UNITS ARE EXERCISED IN A RANDOM SELECTION. WHILE DATA IS BEING TRANSFERRED, SEEK OPERATIONS ARE IN PROGRESS.

%


```
322      ;COPYRIGHT 1972, 977 DIGITAL EQUIPMENT CORP., MAYNARD, MASS.
323
324      ;CONTAINS DEFINITIONS, REGISTER ASSIGNMENTS AND MACRO CALLS
325
326      ;GENERAL REGISTER ASSIGNMENTS
327      000000      R0=%0
328      000001      R1=%1
329      000002      R2=%2
330      000003      R3=%3
331      000004      R4=%4
332      000005      R5=%5
333      000006      SP=%6
334      000007      PC=%7
335
336      ;STATUS REGISTER (PSW) BIT ASSIGNMENTS
337      000001      C=1          ;C BIT
338      000002      V=2          ;V BIT
339      000004      Z=4          ;Z BIT
340      000010      N=10         ;N BIT
341      000020      T=20         ;T BIT
342      000340      PRI7=340     ;PRIORITY LEVEL 7
343      000300      PRI6=300     ;PRIORITY LEVEL 6
344      000240      PRI5=240     ;PRIORITY LEVEL 5
345      000200      PRI4=200     ;PRIORITY LEVEL 4
346      000140      PRI3=140     ;PRIORITY LEVEL 3
347      000100      PRI2=100     ;PRIORITY LEVEL 2
348      000040      PRI1=40      ;PRIORITY LEVEL 1
349
350      ;VECTOR ADDRESSES
351      000004      ERRVEC=4      ;ERROR VECTOR
352      000010      RESVEC=10    ;RESERVED INST VECTOR
353      000014      TBITVEC=14   ;T BIT VECTOR
354      000020      IOTVEC=20    ;IOT TRAP VECTOR
355      000024      PFVEC=24     ;POWER FAIL VECTOR
356      000030      EMTVEC=30    ;EMT VECTOR
357      000034      TRAPVEC=34   ;TRAP VECTOR
358
359      ;REGISTER ADDRESSES
360      177776      PSW=177776   ;PROCESSOR STATUS REGISTER
361      177560      TKS=177560   ;KEYBOARD CSR
362      177562      TKB=177562   ;ADDR OF KEYBOARD BUFFER
363      177564      TPS=177564   ;TELEPRINTER CSR
364      177566      TPB=177566   ;TELEPRINTER BUFFER
365      177570      SWR=177570   ;CONSOLE SWITCH REGISTER
366      177570      DISPLAY=177570 ;CONSOLE DISPLAY REGISTER
367
368      ;INITIAL STACK POINTER
369      000500      STKPTR=500    ;PROGRAM STACK POINTER
370
371      ;BIT ASSIGNMENTS
372      100000      B15=100000
373      040000      B14=40000
374      020000      B13=20000
375      010000      B12=10000
376      004000      B11=4000
```

377	002000	B10=2000
378	001000	B9=1000
379	000400	B8=400
380	000200	B7=200
381	000100	B6=100
382	000040	B5=40
383	000020	B4=20
384	000010	B3=10
385	000004	B2=4
386	000002	B1=2
387	000001	B0=1

;MEMORY MANAGEMENT REGISTER ASSIGNMENTS

391	177572	SRO=177572
392	172340	KIPAR0=172340
393	172342	KIPAR1=172342
394	172344	KIPAR2=172344
395	172346	KIPAR3=172346
396	172350	KIPAR4=172350
397	172352	KIPAR5=172352
398	172354	KIPAR6=172354
399	172356	KIPAR7=172356
400	172300	KIPDR0=172300
401	172302	KIPDR1=172302
402	172304	KIPDR2=172304
403	172306	KIPDR3=172306
404	172310	KIPDR4=172310
405	172312	KIPDR5=172312
406	172314	KIPDR6=172314
407	172316	KIPDR7=172316
408	000006	RW=6
409	000000	UP=00

;INSTRUCTION EQUATES

413	104400	HLT=TRAP	;HLT IS A TRAP TO THE ERROR ROUTINE
415	104000	SCOPE=EMT	;SCOPE IS AN EMT TRAP

;INDEX OF MACROS

419	.SCOPE
420	.SAVE
421	.REST
422	.ERROR
423	.PRINT
424	.DUMP
425	.RAND
426	.READ
427	.PACK

;INDEX OF CALLS

430	SCOPE
431	SAVE
432	REST

```

433      :      HLT
434      :      PRINT
435      :      DUMP
436      :      DUMPF
437      :      SDUMP
438      :      SDUMPF
439      :      RAND
440      :      READ
441      :      PACK
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459      000046      .-46
460      000046      003002      $ENDAD
461      000052      000052      . 52
462      000052      000000      0
463      000200      000200      . 200
464      000200      012707      002350      MOV      #START,PC      ;GO TO START OF TEST
465      001000      001000      .-1000
466      001000      000000      ICNT:    0      ;CONTAINS PASS COUNT
467      001002      000000      LAD:    0      ;PROGRAM TRACE
468      ;SCOPE (EMT) SERVICE ROUTINE
469      ;THIS ROUTINE WILL LOOP IF AN ERROR OCCURED AND
470      ;LOOP ON ERROR SWITCH IS SET (BIT 14). IF LOOPING IS INDICATED
471      ;THE CONTENTS OF 'LAD' EQUAL THE LOOP ADDRESS. IN ORDER
472      ;TO LOOP ON ERROR, BIT 14 OF THE SWITCH REGISTER MUST BE SET AND
473      ;LOCATION 'ERRFLG' MUST BE NEGATIVE INDICATING AN ERROR. ONCE THE
474      ;LOOP IS INITIATED IT WILL CONTINUE UNTIL SWITCH 14 IS CLEARED.
475
476      001004      032737      040000      177570      SCOPE$: BIT      #B14,@#SWR      ;LOOP ON ERRGR?
477      001012      001403      BEQ      2$      ;BRANCH IF NO
478      001014      005767      000232      TST      ERRFLG      ;IS THERE AN ERROR?
479      001020      001003      BNE      1$      ;BRANCH IF YES
480      001022      005067      000224      2$:    CLR      ERRFLG      ;RESET ERROR CONDITION
481      001026      000002      RTI      ;EXIT
482      001030      016716      177746      1$:    MOV      LAD,(SP)      ;MODIFY RETURN ADDRESS
483      001034      000002      RTI      ;EXIT
484      ;ROUTINE TO SAVE REGISTERS ON THE STACK.
485      ;CALLED BY SAVE MACRO
486      001036      012667      000020      SAVE$: MOV      (SP)+,1$      ;SAVE RETURN PC
487      001042      010546      MOV      R5,-(SP)
488      001044      010446      MOV      R4,-(SP)

```

```

489 001046 010346      MOV      R3,-(SP)
490 001050 010246      MOV      R2,-(SP)
491 001052 010146      MOV      R1,-(SP)
492 001054 010046      MOV      R0,-(SP)
493 001056 016707 000000      MOV      1$,PC      ;RETURN
494 001062 000000      1$:      0      ;CONTAINS RETURN ADDRESS
495      ;ROUTINE TO RESTORE REGISTERS SAVED ON THE STACK
496      ;CALLED BY REST MACRO
497 001064 012667 000020      REST$:  MOV      (SP)+,1$      ;SAVE RETURN PC
498 001070 012600      MOV      (SP)+,R0
499 001072 012601      MOV      (SP)+,R1
500 001074 012602      MOV      (SP)+,R2
501 001076 012603      MOV      (SP)+,R3
502 001100 012604      MOV      (SP)+,R4
503 001102 012605      MOV      (SP)+,R5
504 001104 016707 000000      MOV      1$,PC      ;RETURN
505 001110 000000      1$:      0      ;CONTAINS RETURN ADDR
506      ;ERROR SERVICE ROUTINE CALLED BY HLT
507      ;THIS ROUTINE WILL HALT ON ERROR, RING THE BELL, AND
508      ;TRANSFER CONTROL TO A USER SUPPLIED ROUTINE IF SPECIFIED
509 001112 005737 177570      ERROR: TST      @#SWR      ;HALT ON ERROR?
510 001116 100001      BPL      3$      ;BRANCH IF NO
511 001120 000000      HALT
512 001122 032737 004000 177570 3$:  BIT      #B11,@#SWR      ;RING THE BELL?
513 001130 001403      BEQ      1$      ;BRANCH IF NO
514 001132 004567 000156      JSR      R5,PRNTF$      ;FORCE PRINT THE MESSAGE
515 001136 001262      BELL
516 001140 032737 020000 177570 1$:  BIT      #B13,@#SWR      ;SKIP TYPEOUT?
517 001146 001022      BNE      2$      ;BRANCH IF YES
518 001150 004567 000122      JSR      R5,PRINT$      ;PRINT MESSAGE
519 001154 001264      ERRPC
520 001156 011667 000074      MOV      (6),HLTADS$      ;GET ERROR PC+2
521 001162 162767 000002 000066      SUB      #2,HLTADS$      ;MODIFY
522 001170 117767 000062 000056      MOV      @HLTADS$,HLTCT$      ;SAVE HLT ARGUMENT
523 001176 016767 000054 000370      MOV      HLTADS$,ITY
524 001204 004767 000146      JSR      PC,PRINTR      ;TYPE LOCATION WITH LEADING ZEROS
525 001210 004767 003770      JSR      PC,MSG      ;GO TO USER ERROR ROUTINE
526 001214 023737 000042 000046 2$:  CMP      @#42,@#46      ;ARE WE IN ACT11 AUTO MODE?
527 001222 001001      BNE      .+4      ;BRANCH IF NOT
528 001224 000000      HALT      ;HALT ON ERROR IF YES
529 001226 005737 177570      TST      @#SWR      ;HALT ON ERROR IN SWR?
530 001232 100001      BPL      4$      ;BRANCH IF NO
531 001234 000000      HALT
532 001236 052767 100000 000006 4$:  BIS      #B15,ERRFLG      ;SET ERROR FLAG
533 001244 005267 000010      INC      [ERRORS]      ;UPDATE F'JR COUNTER
534 001250 000002      RTI
535 001252 000000      ERRFLG: 0
536 001254 000000      HLTCT$: 0
537 001256 000000      HLTADS$: 0      ;PC OF ERROR
538 001260 000000      ERRORS: 0      ;ERROR COUNT
539 001262 000007      BELL:   .ASCIZ <7>
540 001264 005015 005015 041520      ERRPC:  .ASCIZ <15><12><15><12>'PC= '
541 001272 020075      000
542      001276
543      .EVEN
544      ;THIS ROUTINE WILL PRINT AN ASCIZ MESSAGE.
      ;THE MESSAGE MUST TERMINATE IN 0

```

```

545 001276 032737 020000 177570 PRINTS: BIT      #B13,@#SWR      ;INHIBIT TYPEOUTS?
546 001304 001403          BEQ      PRINTF$    ;BRANCH IF NO
547 001306 062705 000002          ADD      #2,R5      ;UPDATE RETURN ADDR
548 001312 000205          RTS      R5
549 001314 105737 177564          PRNTF$: TSTB   @#TPS      ;WAIT FOR PRINTER TO FINISH
550 001320 100375          BPL      -4
551 001322 010546          MOV      R5,-(SP)
552 001324 062716 000002          ADD      #2,(SP)    ;ADJUST RETURN PC
553 001330 011505          MOV      (R5),R5    ;GET MESSAGE ADDR
554 001332 105715          1$:     TSTB   (R5)      ;CHECK FOR TERMINATOR
555 001334 001002          BNE      2$
556 001336 012605          MOV      (SP)+,R5   ;GET RETURN ADDR
557 001340 000205          RTS      R5        ;RETURN
558 001342 112537 177566          2$:     MOVVB  (R5)+,@#TPB  ;PRINT CHARACTER
559 001346 105737 177564          TSTB   @#TPS      ;WAIT TILL DONE
560 001352 100375          BPL      -4
561 001354 000766          BR      1$
562          ;THIS ROUTINE TYPES A LOCATION IN OCTAL
563 001356 032737 020000 177570 PRINTR: BIT      #B13,@#SWR      ;INHIBIT TYPEOUT?
564 001364 001406          BEQ      PRINTA    ;BRANCH IF NO
565 001366 000207          RTS      PC
566 001370 032737 020000 177570 PRINTS: BIT      #B13,@#SWR      ;INHIBIT TYPEOUT?
567 001376 001405          BEQ      PRINTB    ;BRANCH IF NO
568 001400 000207          RTS      PC
569 001402 112767 000001 000140 PRINTA: MOVVB  #1,.PR    ;SET ZERO FILL SWITCH
570 001410 000402          BR      +6        ;SKIP
571 001412 005067 000132          PRINTB: CLR     .PR    ;SUPPRESS LEADING ZEROS
572 001416 112767 177772 000125          MOVVB  #-6,.PR+1    ;SET COUNT
573 001424 010446          .PTIT: MOV     R4,-(SP) ;SAVE R4
574 001426 012704 001552          MOV     #.PR+2,R4   ;SET POINTER TO FIRST CHARACTER
575 001432 105014          CLRB   (R4)        ;CLEAR FIRST BYTE
576 001434 000413          BR     .PRF        ;ROTATE FIRST BIT
577 001436 105014          .PRL:  CLRB   (R4)   ;CLEAR BYTE OF CHAR
578 001440 032767 000100 000102          BIT     #100,.PR    ;BIT TYPING MODE
579 001446 001006          BNE     .PRF        ;YES SKIP 2 ROTATES
580 001450 006167 000120          ROL     TTY        ;ROTATE BIT INTO C
581 001454 106114          ROLB   (4)         ;PACK IT
582 001456 006167 000112          ROL     TTY
583 001462 106114          ROLB   (4)
584 001464 006167 000104          .PRF:  ROL     TTY
585 001470 106114          ROLB   (4)
586 001472 105714          ROLB   (4)
587 001474 001402          TSTB   (4)         ;IS IT ZERO
588 001476 105267 000046          BEQ     +6         ;SKIP INC
589 001502 105767 000042          INCB   .PR        ;SET FILL SWITCH
590 001506 001402          TSTB   .PR        ;CHECK FILL SWITCH
591 001510 152724 000060          BEQ     +6         ;SKIP BITSET
592 001514 105267 000031          BISB   #'0,(4)+   ;MAKE INTO ASCIZ CHAR
593 001520 001346          INCB   .PR+1      ;INC COUNT
594 001522 022704 001552          BNE     .PRL      ;REPEAT
595 001526 001002          CMP    #.PR+2,R4  ;EMPTY BUFFER
596 001530 112724 000060          BNE     +6         ;SKIP IF NOT
597 001534 105014          MOVVB  #'0,(4)+   ;LOAD ONE ZERO
598 001536 004567 177534          CLRB   (4)        ;NULL TERMINATOR
599 001542 001552          JSR    R5,PRINT$  ;PRINT MESSAGE
600 001544 012604          .PR+2  MOV     (SP)+,R4  ;RESTORE R4

```

601	001546	000207			RTS	PC		
602	001550	000012			.BLKW	12		
603	001574	000000			0			
604	001576							
605	001576	004767	177234		JSR	PC,SAVE\$:SAVE THE REGISTERS
606	001602	016700	000106		MOV	LONUM,R0		:SET R0 WITH LOW
607	001606	016701	000100		MOV	HINUM,R1		:SET R1 WITH HIGH
608	001612	012703	177771		MOV	#-7,R3		:SET SHIFT COUNT
609	001616	005002			CLR	R2		
610	001620	006300		1\$:	ASL	R0		:SHIFT R0 LEFT AND
611	001622	006101			ROL	R1		:ROTATE CARRY INTO R1 AND
612	001624	006102			ROR	R2		:ROTATE CARRY INTO R2
613	001626	005203			INC	R3		:CHECK FOR DONE
614	001630	001373			BNE	1\$		
615	001632	066702	000056		ADD	LONUM,R2		:ADD # TO MAKE X 129
616	001636	005501			ADC	R1		:PROPOGATE CARRY
617	001640	066701	000046		ADD	HINUM,R1		:ADD # TO MAKE X 129
618	001644	005502			ADC	R2		:PROPOGATE CARRY
619	001646	062700	001057		ADD	#1057,R0		
620	001652	005501			ADC	R1		:PROPOGATE CARRY
621	001654	005502			ADC	R2		:PROPOGATE CARRY
622	001656	062701	047401		ADD	#47401,R1		
623	001662	005502			ADC	R2		
624	001664	062702	000006		ADD	#6,R2		
625	001670	060200			ADD	R2,R0		
626	001672	005501			ADC	R1		
627	001674	010067	000014		MOV	R0,LONUM		
628	001700	010167	000006		MOV	R1,HINUM		
629	001704	004767	177154		JSR	PC,REST\$:RESTORE THE REGISTERS
630	001710	000207			RTS	PC		
631								
632	001712	000000			HINUM:	0		
633	001714	000000			LONUM:	0		
634	001716	010346			READ\$:	MOV	R3,-(6)	:SAVE R3
635	001720	012703	002026		1\$:	MOV	#INPUT\$,R3	:GET BUFFER ADDR
636	001724	022703	002046		2\$:	CMP	#INPUT\$+20,R3	:BUFFER FULL?
637	001730	001412				BEQ	4\$:YES..TYPE ?
638	001732	105737	177560			TSTB	@#177560	:WAIT FOR A CHAR
639	001736	100375				BPL	.-4	
640	001740	113713	177562			MOVB	@#177562,(3)	:GET CHAR
641	001744	142713	002200			BICB	#200,(3)	:GET RID OF JUNK
642	001750	122713	000177			CMPB	#177,(3)	:IS IT A RUBOUT?
643	001754	001004				BNE	3\$:SKIP IF NO
644	001756			4\$:				
645	001756	004567	177314			JSR	R5,PRINT\$:PRINT MESSAGE
646	001762	002066				READM\$		
647	001764	000755				BR	1\$:CLEAR BUFFER AND START OVER
648	001766	013737	177562	177566	3\$:	MOV	@#TKB,@#TPB	:ECHO THE CHAR
649	001774	105737	177564			TSTB	@#TPS	
650	002000	100375				BPL	.-4	:WAIT FOR READY
651	002002	122723	000015			CMPB	#15,(3)+	:CHECK FOR RETURN
652	002006	001346				BNE	2\$:LOOP IF NOT RETURN
653	002010	105063	177777			CLRB	-1(3)	:REMOVE THE RETURN
654	002014	004567	177256			JSR	R5,PRINT\$:PRINT MESSAGE
655	002020	002072				READL\$		
656	002022	012603				MOV	(6)+,R3	:RESTORE R3

```

657 002024 000207          RTS      PC          ;RETURN
658
659 002026 000020          INPUT$: .BLKW 20
660 002066 006477 000012    READM$: .ASCIZ '?'<15><12>
661 002072 000012    READL$: .ASCIZ <12>
662
663          ;TAKE THE CONTENTS OF THE TTY INPUT BUFFER AND
664          ;PACK THEM INTO ONE WORD TO CREATE AN OCTAL NUMBER
665
666 002074          PACK$:
667 002074 004767 176736    JSR      PC,SAVES$      ;SAVE THE REGISTERS
668 002100 005067 000242    CLR      NUMS$
669 002104 005000          CLR      R0
670 002106 105760 002026    2$:     TSTB     INPUT$(R0)
671 002112 001402          BEQ
672 002114 005200          INC      R0
673 002116 000773          BR       2$
674 002120 005300          1$:     DEC      R0
675 002122 004767 000166    JSR      PC,PACS$      ;GET OCTAL CHAR
676 002126 016767 000212 000212    MOV      PK$,NUMS$     ;PACK FIRST CHAR
677 002134 004767 000154    JSR      PC,PACS$     ;GET OCTAL CHAR
678 002140 000241          CLC
679 002142 006167 000176    ROL      PK$
680 002146 006167 000172    ROL      PK$
681 002152 006167 000166    ROL      PK$
682 002156 056767 000162 000162    BIS      PK$,NUMS$     ;PACK SECOND CHAR
683 002164 004767 000124    JSR      PC,PACS$     ;GET OCTAL CHAR
684 002170 000241          CLC
685 002172 000367 000146    SWAB     PK$
686 002176 006067 000142    ROR      PK$
687 002202 006067 000136    ROR      PK$
688 002206 056767 000132 000132    BIS      PK$,NUMS$     ;PACK THIRD CHAR
689 002214 004767 000074    JSR      PC,PACS$     ;GET OCTAL CHAR
690 002220 000367 000120    SWAB     PK$
691 002224 000241          CLC
692 002226 006167 000112    ROL      PK$
693 002232 056767 000106 000106    BIS      PK$,NUMS$     ;PACK FOURTH CHAR
694 002240 004767 000050    JSR      PC,PACS$     ;GET OCTAL CHAR
695 002244 000367 000074    SWAB     PK$
696 002250 000241          CLC
697 002252 006167 000066    ROL      PK$
698 002256 006167 000062    ROL      PK$
699 002262 006167 000056    ROL      PK$
700 002266 006167 000052    ROL      PK$
701 002272 056767 000046 000046    BIS      PK$,NUMS$     ;PACK FIFTH CHAR
702 002300 000402          BR       PKEX1$
703 002302 062706 000002    PKEX$:  ADD      #2,SP      ;MODIFY STACK
704 002306          PKEX1$:
705 002306 004767 176552    JSR      PC,REST$     ;RESTORE THE REGISTERS
706 002312 000207          RTS      PC          ;EXIT
707
708 002314 005700          PAC$:   TST      R0
709 002316 10C771          BMI     PKEX$
710 002320 005067 000020    CLR      PK$
711 002324 116067 002026 000012    MOVB     INPUT$(R0),PK$ ;GET INPUT CHAR
712 002332 005300          DEC      R0
  
```

713	002334	042767	177770	000002	BIC	#177770,PK\$:CLEAR UNWANTED BITS
714	002342	000207			RTS	PC	
715							
716	002344	000000			PK\$:	0	
717	002346	000000			NUM\$:	0	
718							


```

719
720 002350 012706 000500          START:  MOV    #STKPTR,SP      ;SET STACK POINTER
721 002354 012737 000340 177776    MOV    #340,@#PSW      ;LOCK UP INTERRUPTS
722 002362 012767 001112 175444    MOV    #ERROR,34      ;SETUP ERROR TRAP
723 002370 012767 000340 175440    MOV    #PRI7,36
724 002376 012767 001004 175424    MOV    #SCOPE$,30     ;SETUP SCOPE TRAP
725 002404 012767 000340 175420    MOV    #PRI7,32
726 002412 012737 005032 000254    MOV    #DSKINT,@#VECTOR ;SET UP DISK INTERRUPT VECTOR
727 002420 012737 000340 000256    MOV    #340,@#STATUS
728 002426 023737 000042 000046    CMP    @#42,@#46      ;ARE WE IN ACT11 AUTO MODE?
729 002434 001403                BEQ    1$              ;YES, SKIP TITLE
730 002436 004567 176634          JSR    R5,PRINT$      ;PRINT MESSAGE
731 002442 005742                TITLE
732 002444 005000          1$:  CLR    R0
733 002446 005060 005474          CLRTAB: CLR   DEVTBL(R0) ;CLEAR THE DEVICE TABLE
734 002452 005720                TST    (R0)+
735 002454 022700 000200          CMP    #128.,R0
736 002460 001372                BNE    CLRTAB
737 002462 005737 000042          TST    @#42           ;UNDER MONITOR CONTROL?
738 002466 001424                BEQ    5$              ;BRANCH IF NO
739 002470 005000                CLR    R0              ;CLEAR MODIFIER
740 002472 005001                CLR    R1
741 002474 012777 000001 003216  7$:  MOV    #1,@RPCS        ;CLEAR THE CONTROLLER
742 002502 110177 003214          MOVVB  R1,@RPCS1      ;SELECT UNIT
743 002506 005777 003224          TST    @RPDS          ;IS UNIT READY?
744 002512 100403                BMI    6$              ;BRANCH IF YES
745 002514 052760 100000 005474  6$:  BIS    #B15,DEVTBL(R0) ;SET UNIT UNAVAILABLE BIT
746 002522 062700 000020          ADD    #16.,R0        ;UPDATE MODIFIER
747 002526 005201                INC    R1              ;UPDATE UNIT NUMBER
748 002530 032701 000010          BIT    #B3,R1         ;ALL UNITS TESTED?
749 002534 001757                BEQ    7$              ;BRANCH IF NO
750 002536 000420                BR     8$
751 002540 012701 000001          5$:  MOV    #1,R1
752 002544 005000                CLR    R0
753 002546 030137 177570          2$:  BIT    R1,@#SWR      ;TEST THE SWITCH REGISTER
754 002552 001003                BNE    1$              ;TO DETERMINE WHICH UNITS
755 002554 052760 100000 005474  1$:  BIS    #B15,DEVTBL(R0) ;TO TEST. IF THE UNIT IS UNAVAILABLE
756 002562 062700 000020          ADD    #16.,R0        ;SET BIT 15 IN THE DEVICE TABLE
757 002566 000241                CLC
758 002570 006101                ROL    R1
759 002572 032701 000400          BIT    #B8,R1         ;HAVE ALL UNITS BEEN SCANNED?
760 002576 001763                BEQ    2$              ;NO-BRANCH
761 002600 000005          8$:  RESET
762 002602 004567 001762          JSR    R5,EXTMEN      ;DETERMINE AMOUNT OF CORE
763 002606 005067 003062          CLR    UNIT           ;INITIALIZE POINTER
764 002612 005067 003060          CLR    PASSCT
765 002616 005005                CLR    R5
766 002620 032765 100000 005474  4$:  BIT    #B15,DEVTBL(R5) ;IS UNIT AVAILABLE?
767 002626 001002                BNE    3$              ;BRANCH IF NO
768 002630 004767 000160          JSR    PC,HOME        ;DO A HOME SEEK
769 002634 005267 003034          3$:  INC    UNIT           ;UPDATE UNIT
770 002640 062705 000020          ADD    #16.,R5        ;UPDATE TABLE POINTER
771 002644 032767 000010 003022  BIT    #B3,UNIT       ;HAVE ALL UNITS BEEN HOMED?
772 002652 001762                BEQ    4$              ;NO-BRANCH
773 002654 005067 003014          LOOP: CLR   UNIT
774 002660 005005                CLR    R5

```

```

775 002662 032765 100000 005474 MAIN: BIT #B15,DEVTL(R5) ;IS THE UNIT AVAILABLE?
776 002670 001004 BNE 1$ ;BRANCH IF NO
777 002672 016504 005474 MOV DEVTBL(R5),R4
778 002676 004774 003172 JSR PC,@JMPTBL(R4) ;PERFORM FUNCTION IN JMPTBL
779 002702 005267 002766 1$: INC UNIT ;UPDATE UNIT
780 002706 062705 000020 ADD #16,R5 ;UPDATE TABLE POINTER
781 002712 032767 000010 002754 BIT #B3,UNIT ;HAVE ALL UNITS BEEN SCANNED?
782 002720 001760 BEQ MAIN ;NO BRANCH
783 002722 005267 002750 INC PASSCT ;INCREMENT ITERATION COUNTER
784 002726 016737 002744 177570 MOV PASSCT,@#SWR ;DISPLAY COUNT
785 002734 022767 000004 002734 CMP #4,PASSCT ;TEST TO SEE IF THIS IS THE FIRST TIME
786 ;WE HAVE DONE THE FOUR TEST FUNCTIONS
787 ;ON EACH DRIVE (SEEK,SEKCK,WRITE,READD).
788 002742 001004 BNE 2$ ;BRANCH IF NO
789 002744 004567 176326 JSR R5,PRINT$ ;PRINT MESSAGE
790 002750 006213 MES20 ;GIVES A QUICK INDICATION THAT THE UNITS
791 ;ARE UP & RUNNING.
792 002752 000404 BR 2$+10
793 002754 022767 005000 002714 2$: CMP #5000,PASSCT ;IS PASS COMPLETE?
794 002762 001013 BNE MEXIT1 ;BRANCH IF NO
795 002764 004567 176306 JSR R5,PRINT$ ;PRINT MESSAGE
796 002770 006243 MES21
797 002772 013701 000042 MOV @#42,R1 ;GET RETURN ADDRESS
798 002776 001405 BEQ MEXIT1
799 003000 000005 RESET
800 003002 004711 $ENDAD: JSR PC,(R1) ;RET JRN TO MONITOR
801 003004 000240 NOP
802 003006 000240 NOP
803 003010 000240 NOP
804 003012 000720 MEXIT1: BR LOOP ;LOOP
805
806 ;THIS ROUTINE WILL SEEK HOME THE PACK WHOSE
807 ;ADDRESS IS IN UNIT.
808
809 003014 116777 002654 002700 HOME: MOVB UNIT,@RPCS1 ;LOAD THE UNIT #
810 003022 005777 002710 TST @RPDS ;IS THE UNIT READY?
811 003026 100401 BMI 5$ ;BRANCH IF READY
812 003030 104400 HLT ;UNIT IS NOT READY
813 003032 112777 000015 002660 5$: MOVB #15,@RPCS ;DO A HOME SEEK
814 003040 012704 000025 MOV #25,R4
815 003044 005304 1$: DEC R4 ;WAIT FOR SEEK TO START
816 003046 001376 BNE 1$
817 003050 032777 002000 002660 BIT #B10,@RPDS ;IS SEEK UNDER WAY SET
818 003056 001001 BNE 2$ ;YES
819 003060 104400 HLT ;SEEK UNDERWAY DID NOT SET
820 003062 016704 002606 2$: MOV UNIT,R4
821 003066 005067 000066 CLR ATTNB
822 003072 116467 003162 000060 MOVB ATTN(R4),ATTNB ;DETERMINE ATTENTION RESPONSE
823 003100 012701 000005 MOV #5,R1 ;BEGINNING OF DELAY
824 003104 005301 DEC R1 ;LOOP TO ALLOW DRIVE
825 ;TIME TO BECOME READY
826 003106 005000 CLR R0
827 003110 005200 7$: INC R0
828 003112 001376 BNE 7$
829 003114 005701 6$: TST R1
830 003116 001372 BNE .-12
    
```

```

831 003120 005200          INC      R0          ;TIME OUT ATTENTION BIT
832 003122 036777 000032 002606  BIT     ATTNB,@RPDS ;DID ATTENTION SET?
833 003130 001003          BNE     3$          ;BRANCH IF YES
834 003132 005700          TST     R0          ;DID IT TIME OUT?
835 003134 001367          BNE     6$          ;BRANCH IF NO
836 003136 104400          HLT                    ;ATTENTION BIT DID NOT SET
837
838 003140 005777 002554    3$:     TST     @RPCS        ;ANY DEVICE STATUS ERRORS?
839 003144 100001          BPL     4$          ;NO-BRANCH
840 003146 104400          HLT                    ;DEVICE STATUS ERROR AFTER HOME COMMAND
841 003150 116777 000004 002560  4$:     MOV     ATTNB,@RPDS ;CLEAR ATTENTION BIT
842 003156 000207          RTS     PC          ;EXIT
843
844 003160 000000          ATTNB: 0            ;CONTAINS ATTENTION BIT FOR CURRENT UNIT
845 003162 001 002 004    ATTN:  .BYTE 1,2,4,10,20,40,100,200
846 003165 010 020 040
847 003170 100 200
848
849          .EVEN
850          ;THIS TABLE DETERMINES WHERE CONTROL WILL GO FOR ANY
851          ;PARTICULAR UNIT. THE FIRST WORD OF THE DEVICE TABLE
852          ;IS USED AS A MODIFIER FOR A JSR INDIRECT INTO
853          ;THE FOLLOWING TABLE.
854
855 003172 003202          JMPTBL: SEEK        ;SEEK A RANDOM CYLINDER
856 003174 003552          SEKCK              ;SEE IF SEEK IS COMPLETE
857 003176 004044          WRITE            ;WRITE RANDOM DATA
858 003200 004220          READD            ;READ AND VERIFY RANDOM DATA
859
860          ;THIS ROUTINE WILL GENERATE A RANDOM CYLINDER
861          ;ADDRESS AND ISSUE A SEEK TO IT. THE FUNCTION
862          ;POINTER IN THE DEVICE TABLE WILL BE UPDATED TO
863          ;CHECK FOR THE ATTENTION BIT.
864
865 003202 016565 005500 005476  SEEK:  MOV     DEVTBL+4(R5),DEVTBL+2(R5) ;SET UP CYLINDER FROM ADDRESS
866 003210 116777 002460 002504          MOV     UNIT,@RPCS1 ;SELECT THE UNIT
867 003216 032777 100000 002512          BIT     #B15,@RPDS ;UNIT READY?
868 003224 001002          BNE     1$          ;YES-BRANCH
869 003226 104400          HLT                    ;SELECTED UNIT NOT READY
870 003230 000207          RTS     PC          ;EXIT
871 003232          1$:
872 003232 004767 176340          JSR     PC,RAND$     ;GENERATE TWO RANDOM NOS.
873 003236 016767 176452 002436          MOV     LONUM,WORK1
874 003244 016767 176442 002432          MOV     HINUM,WORK2
875 003252 032777 020000 002456          BIT     #B13,@RPDS ;IS THIS AN RP03?
876 003260 001410          BEQ     6$          ;BRANCH IF NO
877 003262 042767 177000 002412          BIC     #177000,WORK1
878 003270 022767 000625 002404          CMP     #625,WORK1 ;GENERATE A CYLINDER ADDRESS
879 003276 002755          BLT     1$
880 003300 000407          BR      7$
881 003302 042767 177400 002372  6$:     BIC     #177400,WORK1
882 003310 022767 000312 002364          CMP     #312,WORK1 ;IS NUMBER TOO LARGE?
883 003316 002745          BLT     1$          ;BRANCH IF YES
884 003320 026765 002356 005476  7$:     CMP     WORK1,DEVTBL+2(R5)
885 003326 001741          BEQ     1$
886 003330 016765 002346 005500          MOV     WORK1,DEVTBL+4(R5) ;SAVE CYLINDER ADDRESS

```

```

887 003336 016765 002342 005502      MOV      WORK2,DEVTBL+6(R5)      ;USE A RANDOM DATA BASE
888 003344 004767 176226                JSR      PC,RAND$              ;GENERATE TWO RANDOM NOS.
889 003350 016767 175340 002324      MOV      LONUM,WORK1
890 003356 016767 176330 002320      MOV      HINUM,WORK2
891 003364 042767 177760 002310      BIC      #177760,WORK1
892 003372 022767 000011 002302      CMP      #11,WORK1              ;GENERATE A RANDOM SECTOR
893 003400 002003                BGE      2$
894 003402 162767 000010 002272      SUB      #8,WORK1
895 003410 042767 177740 002266 2$:   BIC      #177740,WORK2          ;GENERATE A RANDOM TRACK
896 003416 022767 000023 002260      CMP      #23,WORK2
897 003424 002003                BGE      3$
898 003426 162767 000014 002250      SUB      #14,WORK2
899 003434 116767 002244 002241 3$:   MOVB     WORK2,WORK1+1          ;MERGE TRACK AND SECTOR ADDR
900
901 003442 016765 002234 005504      MOV      WORK1,DEVTBL+10(R5)    ;STORE RANDOM DISK ADDRESS
902 003450 005065 005506                CLR      DEVTBL+12(R5)          ;CLEAR TIMEOUT COUNTER
903 003454 016704 002214                MOV      UNIT, R4
904 003460 116467 003162 177472      MOVB     ATTN(R4), ATTNB        ;GET ATTN BIT
905 003466 116777 177466 002242      MOVB     ATTNB, @RPDS           ;CLEAR ATTN BIT
906 003474 016577 005500 002226      MOV      DEVTBL+4(R5),@RPCA      ;LOAD CYLINDER ADDRESS
907 003502 016577 005504 002222      MOV      DEVTBL+10(R5),@RPDA    ;LOAD TRACK AND SECTOR
908 003510 112777 000011 002202      MOVB     #11,@RPCS             ;INITIATE A SEEK
909 003516 012704 000025                MOV      #25,R4
910 003522 005304                4$:   DEC      R4                    ;WAIT FOR SEEK TO START
911 003524 001376                BNE      4$
912 003526 032777 002000 002202      BIT      #B10,@RPDS            ;IS THE SEEK UNDERWAY?
913 003534 001001                BNE      5$                    ;YES-BRANCH
914 003536 104400                HLT
915 003540 005265 005474 5$:   INC      DEVTBL(R5)            ;MODIFY FUNCTION POINTER TO
916 003544 005265 005474                INC      DEVTBL(R5)            ;CHECK FOR SEEK COMPLETE
917 003550 000207                RTS      PC                    ;EXIT
918
919                                     ;THIS ROUTINE IS ENTERED AFTER A SEEK HAS BEEN ISSUED.
920                                     ;IT CHECKS THE ATTENTION FLAG - IF CLEAR IT UPDATES THE
921                                     ;TIMEOUT COUNTER AND CHECKS IT. IF SET IT VERIFIES
922                                     ;THE SEEK FUNCTIONED PROPERLY.
923
924 003552 116777 002116 002142  SEKCK:  MOVB     UNIT, @RPCS1        ;SELECT UNIT
925 003560 016704 002110                MOV      UNIT,R4
926 003564 116467 003162 177366      MOVB     ATTN(R4),ATTNB        ;DETERMINE ATTENTION BIT
927 003572 036777 177362 002136      BIT      ATTNB,@RPDS           ;TEST FOR ATTENTION FLAG
928 003600 001011                BNE      1$                    ;BRANCH IF SET
929 003602 005265 005506                INC      DEVTBL+12(R5)          ;UPDATE TIMEOUT COUNTER
930 003606 022765 005000 005506      CMP      #5000,DEVTBL+12(R5)    ;DID OPERATION TIMEOUT?
931 003614 101002                BHI      2$                    ;NOT YET-BRANCH
932 003616 104400                HLT
933 003620 000437                BR       4$                    ;SEEK TIMED OUT
934 003622 000207                2$:   RTS      PC                    ;EXIT
935 003624 116777 177330 002104 1$:   MOVB     ATTNB,@RPDS           ;CLEAR ATTENTION FLAG
936 003632 032777 002000 002076      BIT      #B10,@RPDS            ;IS SEEK UNDERWAY CLEAR?
937 003640 001402                BEQ      5$                    ;IF YES-BRANCH
938 003642 104400                HLT
939 003644 000425                BR       4$                    ;SEEK UNDERWAY DID NOT CLEAR
940 003646 017704 002066 5$:   MOV      @SUCA,R4              ;GET CURRENT CYLINDER ADDRESS
941 003652 020465 005500                CMP      R4,DEVTBL+4(R5)        ;DOES IT MATCH CYLINDER REQUESTED?
942 003656 001440                BEQ      6$                    ;YES-BRANCH
    
```

```

943 003660 104400          HLT                ;SUCA AND CYL REQUESTED DID NOT COMPARE
944 003662 004567 175410  JSR          R5,PRINT$      ;PRINT MESSAGE
945 003666 003772          MES10
946 003670 016567 005500 175676  MOV          DEVTBL+4(R5),TTY
947 003676 004767 175466  JSR          PC,PRINT$      ;TYPE LOCATION-SUPRESS ZEROS
948 003702 004567 175370  JSR          R5,PRINT$      ;PRINT MESSAGE
949 003706 004021          MES11
950 003710 010467 175660  MOV          R4,TTY
951 003714 004767 175436  JSR          PC,PRINT$      ;TYPE LOCATION WITH LEADING ZEROS
952 003720 032777 004000 002010 4$:  BIT          #B11,@RPDS     ;SEEK INCOMPLETE?
953 003726 001411          BEQ          10$           ;BRANCH IF NO
954 003730 112777 000015 001762  MOVB        #15,@RPCS      ;ISSUE HOME COMMAND
955 003736 105777 001756  TSTB        @RPCS          ;WAIT FOR DONE
956 003742 100375          BPL          -4
957 003744 005777 001766  TST          @RPDS         ;WAIT FOR UNIT READY
958 003750 100375          BPL          -4
959 003752 005065 005474 10$:  CLR          DEVTBL(R5)    ;CLEAR FUNCTION POINTER
960 003756 000207          RTS          PC           ;EXIT
961 003760 005265 005474 6$:  INC          DEVTBL(R5)    ;UPDATE FUNCTION POINTER
962 003764 005265 005474  INC          DEVTBL(R5)    ;UPDATE FUNCTION POINTER
963 003770 000207          RTS          PC           ;EXIT
964
965 003772 005015 042522 052521 MES10: .ASCIZ <15><12>'REQUESTED CYLINDER= '
966 004000 051505 042524 020104
967 004006 054503 044514 042116
968 004014 051105 020075 000
969 004021 015 051412 041525 MES11: .ASCIZ <15><12>'SUCA REGISTER= '
970 004026 020101 042522 044507
971 004034 052123 051105 020075
972 004042 000
973 004044          .EVEN
974
975          ;THIS ROUTINE WILL WRITE A RANDOM SECTOR ON
976          ;A RANDOM TRACK. THE CYLINDER HAS ALREADY
977          ;BEEN SECEYED BY THE SEEK ROUTINE.
978
979 004044 004767 000636          WRITE: JSR          PC,RANADR      ;GENERATE A RANDOM BUFFER ADDR
980 004050 012767 000400 001622  MOV          #400,WORK
981 004056 016701 001630  MOV          BUFF,R1
982 004062 004767 000336          JSR          PC,RANDAT      ;GENERATE A RANDOM PATTERN
983 004066 116777 001602 001626  MOVB        UNIT,@RPCS1    ;SELECT THE UNIT
984 004074 032777 100000 001634  BIT          #B15,@RPDS    ;IS THE SELECTED UNIT READY
985 004102 001003          BNE          1$           ;YES-BRANCH
986 004104 104400          HLT                ;SELECTED UNIT NOT READY
987 004106 000167 000100  JMP          WRTER         ;START SEQUENCE OVER
988
989 004112 012777 177400 001604 1$:  MOV          #-400,@RPWC    ;SETUP WORD COUNT REGISTER
990 004120 016777 001566 001600  MOV          BUFF,@RPBA    ;SETUP BUS ADDR REGISTER
991 004126 016577 005500 001574  MOV          DEVTBL+4(R5),@RPCA ;SET CYLINDER ADDR
992 004134 016577 005504 001570  MOV          DEVTBL+10(R5),@RPDA ;SETUP DISK ADDR.
993 004142 005067 001546          CLR          INT         ;CLEAR INTERRUPT FLAG
994 004146 012737 000200 177776  MOV          #PRI4,@#PSW   ;ALLOW INTERRUPT
995 004154 005067 001530          CLR          INTERR      ;CLEAR ERROR FLAG
996 004160 112777 000113 001532  MOVB        #113,@RPCS    ;INITIATE WRITE WITH INTERRUPT
997 004166 004767 000612          JSR          PC,WAIT      ;TIMEOUT THE OPERATION
998 004172 005767 001512          TST          INTERR      ;ANY ERRORS?
  
```

```

999 004176 001005          BNE    WRTER          ;BRANCH IF YES
1000 004200 005265 005474  INC    DEVTBL(R5)     ;UPDATE FUNCTION POINTER TO READ
1001 004204 005265 005474  INC    DEVTBL(R5)
1002 004210 000403          BR     READD
1003 004212 005065 005474  WRTER: CLR   DEVTBL(R5) ;RESTORE FUNCTION POINTER
1004 004216 000207          RTS    PC             ;EXIT
1005
1006          ;READ AND VERIFY THE DATA WRITTEN
1007
1008 004220 116777 001450 001474 READD: MOVB   UNIT,@RPCS1   ;SELECT THE UNIT
1009 004226 032777 100000 001502  BIT    #B15,@RPDS     ;IS THE SELECTED UNIT READY?
1010 004234 001003          BNE    1$             ;YES-BRANCH
1011 004236 104400          HLT
1012 004240 000167 000152          JMP    RDCNT         ;SELECTED UNIT NOT READY
1013 004244 012777 177400 001452 1$:  MOV    #-400,@RPWC    ;LOAD WORD COUNT REGISTER
1014 004252 016777 001434 001446  MOV    BUFF,@RPBA     ;LOAD BUS ADDR REGISTER
1015 004260 062777 001000 001440  ADD    #1000,@RPBA
1016 004266 016577 005500 001434  MOV    DEVTBL+4(R5),@RPCA ;SET CYLINDER ADDR
1017 004274 016577 005504 001430  MOV    DEVTBL+10(R5),@RPDA ;SETUP DISK ADDR.
1018 004302 005067 001406          CLR   INT             ;CLEAR INTERRUPT FLAG
1019 004306 005067 001376          CLR   INTERR         ;CLEAR ERROR FLAG
1020 004312 112777 000117 001400  MOVB   #117,@RPCS     ;INITIATE READ WITH INTERRUPT
1021 004320 004767 000460          JSR   PC,WAIT        ;TIMEOUT THE OPERATION
1022 004324 032777 040000 001366  BIT    #B14,@RPCS     ;ANY HARD ERRORS?
1023 004332 001031          BNE    RDCNT         ;BRANCH IF YES
1024 004334 016701 001352          MOV   BUFF,R1
1025 004340 016702 001346          MOV   BUFF,R2
1026 004344 005003          CLR   R3
1027 004346 062701 001000          ADD   #1000,R1       ;START OF PATTERN BUFFER
1028 004352 022122          3$:  CMP    (R1)+,(R2)+   ;COMPARE DATA
1029 004354 001006          BNE   2$             ;BRANCH-DATA DID NOT COMPARE
1030 004356 005203          INC   R3             ;INCREMENT COUNTER
1031 004360 022703 000400          CMP   #400,R3       ;HAS BUFFER BEEN SCANNED
1032 004364 001372          BNE   3$             ;BRANCH-NO
1033 004366 000167 000024          JMP   RDCNT
1034
1035 004372 016267 177776 001070 2$:  MOV    -2(R2),EXP$
1036 004400 016167 177776 001064  MOV    -2(R1),REC$
1037 004406 004567 174664          JSR   R5,PRINT$     ;PRINT MESSAGE
1038 004412 006106          MES12
1039 004414 104401          HLT   +1             ;DATA DID NOT VERIFY
1040 004416 005065 005474  RDCNT: CLR   DEVTBL(R5) ;INITIATE FUNCTION POINTER
1041 004422 000207          RTS    PC             ;EXIT
1042
1043          ;GENERATE A RANDOM PATTERN
1044
1045 004424 016567 005502 000132 RANDAT: MOV    DEVTBL+6(R5),RAND1 ;GET RANDOM BASE
1046 004432 016567 005504 000126  MOV    DEVTBL+10(R5),RAND2
1047 004440 016700 000120          MOV   RAND1,R0
1048 004444 016704 000116          MOV   RAND2,R4
1049 004450 012703 000007          RANDA1: MOV   #7,R3   ;SETUP SHIFT COUNT
1050 004454 005002          CLR   R2
1051 004456 006300          SHIFT: ASL   R0       ;SHIFT R0 LEFT AND
1052 004460 006104          ROL   R4             ;ROTATE CARRY INTO LSB OF R0 INTO R4
1053 004462 006102          ROL   R2             ;ROTATE CARRY OUT OF R4 INTO R2
1054 004464 005303          DEC   R3             ;DECREMENT R3

```

```

1055 004466 001373          BNE     SHIFT          ;CONTINUE LOOP
1056 004470 066700 000070  ADD     RAND1,R0       ;ADD IN # TO MAKE X129
1057 004474 005504          ADC     R4             ;PROPOGATE CARRY
1058 004476 066704 000064  ADD     RAND2,R4       ;ADD IN # TO MAKE X129
1059 004502 005502          ADC     R2             ;PROPOGATE CARRY
1060 004504 062700 001057  ADD     #1057,R0       ;ADD LOW CONSTANT
1061 004510 005504          ADC     R4             ;PROPOGATE CARRY
1062 004512 005502          ADC     R2             ;PROPOGATE AGAIN
1063 004514 062704 047401  ADD     #47401,R4      ;ADD HIGH CONSTANT
1064 004520 005502          ADC     R2
1065 004522 062702 000006  ADD     #6,R2
1066 004526 060200          ADD     R2,R0          ;REPRIME R0 WITH HIGH DIGIT
1067 004530 005504          ADC     R4
1068 004532 010067 000026  MOV     R0,RAND1
1069 004536 010021          MOV     R0,(R1)+       ;STORE DATA IN BUFFER
1070 004540 005367 001134  DEC     WORK
1071 004544 001406          BEQ     EXGEN
1072 004546 010467 000014  MOV     R4,RAND2
1073 004552 010421          MOV     R4,(R1)+       ;STORE DATA IN BUFFER
1074 004554 005367 001120  DEC     WORK
1075 004560 001333          BNE     RANDA1         ;FILL ENTIRE BUFFER
1076 004562 000207          EXGEN: RTS            ;EXIT
1077
1078 004564 000000          RAND1: 0
1079 004566 000000          RAND2: 0
1080
1081          ;THIS ROUTINE DETERMINES THE TOTAL AMOUNT OF AVAILABLE
1082          ;CORE WITHOUT USING MEMORY MANAGEMENT.
1083
1084 004570 012737 000340 177776  EXTMEN: MOV     #PRI7,@#PSW ;LOCKUP PRIORITY LEVEL
1085 004576 012767 004646 173200          MOV     #MAXREF,4      ;SETUP IO BUS TRAP
1086 004604 012767 000340 173174          MOV     #PRI7,6
1087 004612 012767 017446 000064          MOV     #17446,SAVE    ;START WITH 4K
1088 004620 005777 000060          EXREF: TST     @SAVE    ;REFERENCE MEMORY
1089 004624 022767 157446 000052          CMP     #157446,SAVE  ;TEST FOR 28K
1090 004632 001001          BNE     1$           ;BRANCH IF LESS THAN 28K
1091 004634 000407          BR     MAXRF1
1092 004636 062767 020000 000040  1$:  ADD     #20000,SAVE ;SETUP FOR NEXT REFERENCE
1093 004644 000765          BR     EXREF
1094
1095          ;ENTER HERE WHEN IO BUS ERROR OCCURS
1096
1097 004646 162767 020000 000030  MAXREF: SUB     #20000,SAVE
1098 004654 012767 000006 173122  MAXRF1: MOV     #6,4    ;RESTORE IO BUS TRAY
1099 004662 005067 173120          CLR     6
1100 004666 005737 000042          TST     @#42          ;UNDER MONITOR CONTROL?
1101 004672 001403          BEQ     1$           ;BRANCH IF NO
1102 004674 162767 005670 000002          SUB     #3000.,SAVE   ;ADJUST TOP OF CORE
1103 004702 000205          1$:  RTS     R5       ;EXIT-SAVE=MAXIMUM MEMORY
1104 004704 000000          SAVE:  0           ;HIGHEST AVAILABLE LOCATION
1105
1106          ;GENERATE A RANDOM BUFFER ADDRESS
1107
1108 004706 016704 177772          RANADR: MOV     SAVE,R4
1109 004712 162704 006256          SUB     #ENDP,R4      ;DETERMINE BUFFER SIZE
1110 004716 162704 002000          SUB     #2000,R4      ;ALLOW ROOM FOR DATA

```

```

1111 004722 004767 174650          JSR    PC,RAND$      ;GENERATE TWO RANDOM NOS.
1112 004726 016767 174762 000746  MOV    LONUM,WORK1
1113 004734 042767 000001 000740  BIC    #B0,WORK1    ;MAKE NUMBER EVEN
1114 004742 012703 100000          MOV    #100000,R3
1115 004746 020467 000730          2$:   CMP    R4,WORK1  ;ENSURE THAT THE RANDOM
1116 004752 101005          BHI    1$           ;ADDRESS FITS WITHIN AVAILABLE
1117 004754 040367 000722          BIC    R3,WORK1    ;MEMORY
1118 004760 000241          CLC
1119 004762 006003          ROR    R3
1120 004764 000770          BR     2$
1121 004766 062767 006256 000706 1$:   ADD    #ENDP,WORK1
1122 004774 016767 000702 000710  MOV    WORK1,BUFF  ;SAVE BUFFER START ADDR.
1123 005002 000207          RTS    PC          ;EXIT
1124          ;TIMEOUT THE OCCURANCE OF AN INTERRUPT
1125
1126 005004 005000          WAIT:  CLR    R0
1127 005006 005200          2$:   INC    R0
1128 005010 005767 000700          TST    INT         ;HAS INTERRUPT OCCURED?
1129 005014 001005          BNE    1$         ;YES-BRANCH
1130 005016 005700          TST    R0         ;HAS OPERATION TIMED OUT?
1131 005020 001372          BNE    2$         ;NO-BRANCH
1132 005022 104400          HLT
1133 005024 005267 000660          INC    INTERR     ;SET ERROR FLAG
1134 005030 000207          1$:   RTS    PC          ;EXIT
1135
1136          ;ENTERED UPON A DEVICE INTERRUPT. THIS ROUTINE
1137          ;WILL CHECK FOR AND REPORT DEVICE ERRORS
1138
1139 005032 032777 100000 000660  DSKINT: BIT    #B15,@RPCS ;WHERE THER ANY ERRORS?
1140 005040 001402          BEQ    1$         ;BRANCH-NO ERRORS
1141 005042 000167 000110          JMP    DSKER     ;REPORT ERROR
1142 005046 016703 000640          1$:   MOV    BUFF,R3
1143 005052 062703 001000          ADD    #1000,R3
1144 005056 022765 000004 005474  CMP    #4,DEVTBL(R5) ;IS THIS A WRITE?
1145 005064 001402          BEQ    3$         ;BRANCH IF YES
1146 005066 062703 001000          ADD    #1000,R3
1147 005072 020377 000630          3$:   CMP    R3,@RPBA
1148 005076 001425          BEQ    2$         ;DID THE BUS ADDR TERMINATE PROPERLY?
1149 005100 104400          HLT             ;YES-BRANCH
1150 005102 004567 174170          JSR    R5,PRINT$ ;CONTENTS OF RPBA INCORRECT
1151 005106 006133          MES13          ;PRINT MESSAGE
1152 005110 004567 174162          JSR    R5,PRINT$ ;PRINT MESSAGE
1153 005114 006163          MES18
1154 005116 010367 174452          MOV    R3,TTY
1155 005122 004767 174230          JSR    PC,PRINTR ;TYPE LOCATION WITH LEADING ZEROS
1156 005126 004567 174144          JSR    R5,PRINT$ ;PRINT MESSAGE
1157 005132 006177          MES19
1158 005134 017767 000566 174432  MOV    @RPBA,TTY
1159 005142 004767 174210          JSR    PC,PRINTR ;TYPE LOCATION WITH LEADING ZEROS
1160 005146 005267 000536          INC    INTERR     ;SET ERROR FLAG
1161 005152 000167 000006          2$:   JMP    EXTINT
1162 005156 104400          DSKER: HLT
1163 005160 005267 000524          INC    INTERR     ;REPORT INTERRUPT DISK ERROR
1164 005164 052767 000001 000522  EXTINT: BIS    #B0,INT ;SET ERROR FLAG
1165 005172 032777 100000 000536  1$:   BIT    #B15,@RPDS ;IS THE UNIT READY
1166 005200 001774          BEQ    1$         ;NO-WAIT

```



```

1167 005202 000002          RTI          ;EXIT INTERRUPT
1168
1169
1170 005204 032767 000002 174042 MSG:  BIT   #B1,HLTCT$  ;TYPE ENTIRE MSG?
1171 005212 001100          BNE   1$          ;BRANCH IF NO
1172 005214 004567 174056          JSR   R5,PRINT$  ;PRINT MESSAGE
1173 005220 006001          MES1
1174 005222 016767 000446 174344          MOV   UNIT,TTY
1175 005230 004767 174134          JSR   PC,PRINTS  ;TYPE LOCATION-SUPRESS ZEROS
1176 005234 004567 174036          JSR   R5,PRINT$  ;PRINT MESSAGE
1177 005240 006033          MES2A
1178 005242 017767 000470 174324          MOV   @RPDS,TTY
1179 005250 004767 174102          JSR   PC,PRINTR
1180 005254 004567 174016          JSR   R5,PRINT$  ;PRINT MESSAGE
1181 005260 006011          MES1A
1182 005262 017767 000446 174304          MOV   @RPER,TTY
1183 005270 004767 174062          JSR   PC,PRINTR  ;TYPE LOCATION WITH LEADING ZEROS
1184 005274 004567 173776          JSR   R5,PRINT$  ;PRINT MESSAGE
1185 005300 006022          MES2
1186 005302 017767 000412 174264          MOV   @RPCS,TTY
1187 005310 004767 174042          JSR   PC,PRINTR  ;TYPE LOCATION WITH LEADING ZEROS
1188 005314 004567 173756          JSR   R5,PRINT$  ;PRINT MESSAGE
1189 005320 006044          MES3
1190 005322 016567 005500 174244          MOV   DEVTBL+4(R5),TTY
1191 005330 004767 174034          JSR   PC,PRINTS  ;TYPE LOCATION-SUPRESS ZEROS
1192 005334 004567 173736          JSR   R5,PRINT$  ;PRINT MESSAGE
1193 005340 006061          MES4
1194 005342 005067 000340          CLR   WORK3
1195 005346 116567 005505 000332          MOVB  DEVTBL+11(R5),WORK3
1196 005354 016767 000326 174212          MOV   WORK3,TTY
1197 005362 004767 174002          JSR   PC,PRINTS  ;TYPE LOCATION-SUPRESS ZEROS
1198 005366 004567 173704          JSR   R5,PRINT$  ;PRINT MESSAGE
1199 005372 006073          MES5
1200 005374 116567 005504 000304          MOVB  DEVTBL+10(R5),WORK3
1201 005402 016767 000300 174164          MOV   WORK3,TTY
1202 005410 004767 173754          JSR   PC,PRINTS  ;TYPE LOCATION-SUPRESS ZEROS
1203 005414 032767 000001 173632 1$:  BIT   #B0,HLTCT$  ;TYPE EXPECTED - RECEIVED?
1204 005422 001001          BNE   2$          ;BRANCH IF YES
1205 005424 000207          RTS   PC
1206 005426          2$:
1207 005426 004567 173644          JSR   R5,PRINT$  ;PRINT MESSAGE
1208 005432 006163          MES18
1209 005434 016767 000030 174132          MOV   EXP$,TTY
1210 005442 004767 173710          JSR   PC,PRINTR  ;TYPE LOCATION WITH LEADING ZEROS
1211 005446 004567 173624          JSR   R5,PRINT$  ;PRINT MESSAGE
1212 005452 006177          MES19
1213 005454 016767 000012 174112          MOV   RECS,TTY
1214 005462 004767 173670          JSR   PC,PRINTR  ;TYPE LOCATION WITH LEADING ZEROS
1215 005466 000207          RTS   PC
1216 005470 000000          EXP$: 0
1217 005472 000000          RECS: 0

```

```

;DEVTBL IS A TABLE CONTAINING SLOTS FOR EACH OF EIGHT
;POSSIBLE UNITS. DURING THE OPERATION OF THE PROGRAM
;R5 IS USED AS A MODIFIER TO POINT INTO THE TABLE TO
;SELECT THE PROPER UNIT. EACH UNIT SLOT CONTAINS
;EIGHT ENTRIES(WORD)

```

1218
 1219
 1220
 1221
 1222

```
1223      ;1 FUNCTION POINTER
1224      :      0=SEEK
1225      :      2=SEEK IN PROGRESS
1226      :      4=WRITE
1227      :      6=READ
1228      :      IF NEGATIVE-UNIT IS NOT TESTED
1229      ;2 CYLINDER FROM ADDRESS-INDICATES PREVIOUS CYLINDER POSITION
1230      ;3 CYLINDER TO ADDRESS-ADDRESS OF THE SEEK COMMAND
1231      ;4 RANDOM BASE FOR PATTERN GENERATION
1232      ;5 RANDOM TRACK AND SECTOR ADDRESS
1233      ;6 CYLINDER SEEK TIMEOUT COUNTER
1234      ;7 SPARE
1235      ;8 SPARE
1236
1237      .EVEN
1238 005474 000000 DEVTBL: 0 ;UNIT 0 SLOT
1239      .=DEVTBL+20
1240 005514 000000 UNIT1: 0 ;UNIT 1 SLOT
1241      .=UNIT1+20
1242 005534 000000 UNIT2: 0 ;UNIT 2 SLOT
1243      .-UNIT2+20
1244 005554 000000 UNIT3: 0 ;UNIT 3 SLOT
1245      .=UNIT3+20
1246 005574 000000 UNIT4: 0 ;UNIT 4 SLOT
1247      .=UNIT4+20
1248 005614 000000 UNIT5: 0 ;UNIT 5 SLOT
1249      .-UNIT5+20
1250 005634 000000 UNIT6: 0 ;UNIT6 SLOT
1251      .-UNIT6+20
1252 005654 000000 UNIT7: 0 ;UNIT 7 SLOT
1253      .=UNIT7+20
1254
1255      ;RP11 CONSTANTS-MEMORY ASSIGNMENTS
1256 005674 000000 UNIT: 0 ;CURRENT UNIT UNDER TEST
1257 005676 000000 PASSCT: 0 ;COUNTS EACH PASS THRU 8 UNITS
1258 005700 000000 WORK: 0 ;TEMPORARY STORAGE AREA
1259 005702 000000 WORK1: 0
1260 005704 000000 WORK2: 0
1261 005706 000000 WORK3: 0
1262 005710 000000 INTERR: 0 ;INTERRUPT ERROR FLAG
1263 005712 000000 BUFF: 0 ;STARTING ADDRESS OF BUFFER
1264 005714 000000 INT: 0 ;INTERRUPT FLAG
1265 005716 000000 FLAG: 0 ;FLAG WORD
```

CZRPCDO, RP11C MULTI-DRIVE
CZRPCD.P11 15-JAN-78 12:36

MACY11 30A(1052) 15-JAN-78 12:41 ^{N 2} PAGE 27

SEQ 0026

1266
1267 005720 176714

;DISK IO REGISTERS
RPCS: 176714

;DISK CONTROL REGISTER

1268	005722	176715	RPCS1:	176715	
1269	005724	176716	RPWC:	176716	:DISK WORD COUNT REGISTER
1270	005726	176720	RPBA:	176720	:CURRENT ADDRSS REGISTER
1271	005730	176722	RPCA:	176722	:CYLINDER ADDRESS REGISTER
1272	005732	176724	RPDA:	176724	:DISK ADDRESS REGISTER
1273	005734	176712	RPER:	176712	:ERROR REGISTER
1274	005736	176710	RPDS:	176710	:DRIVE STATUS REGISTER
1275	005740	176734	SUCA:	176734	:CURRENT CYLINDER ADDRESS
1276		000254	VECTOR=	254	:INTERRUPT VECTOR ADDR.
1277		000256	STATUS=	256	:DISK INTERRUPT STATUS

1278
1279 :MESSAGES

1280
1281 005742 005015 055103 050122 TITLE: .ASCIZ <15><12>/CZRPCDO, RP11C MULTI DRIVE/<15><12>

1282 005750 042103 026060 051040

1283 005756 030520 041461 046440

1284 005764 046125 044524 042040

1285 005772 044522 042526 005015

1286 006000 000

1287 006001 015 052412 044516 MES1: .ASCIZ <15><12>/UNIT /

1288 006006 020124 000

1289 006011 015 051012 042520 MES1A: .ASCIZ <15><12>/RPER= /

1290 006016 036522 000040

1291 006022 005015 050122 051503 MES2: .ASCIZ <15><12>/RPCS= /

1292 006030 020075 000

1293 006033 015 051012 042120 MES2A: .ASCIZ <15><12>/RPDS= /

1294 006040 036523 000040

1295 006044 005015 054503 044514 MES3: .ASCIZ <15><12>/CYLINDER= /

1296 006052 042116 051105 020075

1297 006060 000

1298 006061 015 052012 040522 MES4: .ASCIZ <15><12>/TRACK= /

1299 006066 045503 004475 000

1300 006073 015 051412 041505 MES5: .ASCIZ <15><12>/SECTOR= /

1301 006100 047524 036522 000040

1302 006106 005015 040504 040524 MES12: .ASCIZ <15><12>/DATA COMPARE ERROR/

1303 006114 041440 046517 040520

1304 006122 042522 042440 051122

1305 006130 051117 000

1306 006133 015 041012 051525 MES13: .ASCIZ <15><12>/BUS ADDRESS INCORRECT/

1307 006140 040440 042104 042522

1308 006146 051523 044440 041516

1309 006154 051117 042522 052103

1310 006162 000

1311 006163 015 042412 050130 MES18: .ASCIZ <15><12>/EXPECTED /

1312 006170 041505 042524 020104

1313 006176 000

1314 006177 015 051012 041505 MES19: .ASCIZ <15><12>/RECEIVED /

1315 006204 044505 042526 020104

1316 006212 000

1317 006213 015 042412 042116 MES20: .ASCIZ <15><12>/END QUICK VERIFY PASS/

1318 006220 050440 044525 045503

1319 006226 053040 051105 043111

1320 006234 020131 040520 051523

1321 006242 000

1322 006243 015 042412 042116 MES21: .ASCIZ <15><12>/END PASS/

1323 006250 050040 051501 000123

CZRPCDO, RP11C MULTI-DRIVE
CZRPCD.P11 15-JAN-78 12:36

MACY11 30A(1052) 15-JAN-78 12:41^{C 3} PAGE 29

SEQ 0028

1324 006256 000000
1325 000001

ENDP: 0
.END

;START OF BUFFER AREA

.SAPTY 1#
.SASTA 1#
.SCATC 1#
.SCMTA 1#
.\$DB2D 1#
.\$DB2O 1#
.\$DIV 1#
.\$EOP 1#
.\$ERRO 1#
.\$ERRT 1#
.\$MULT 1#
.\$POWE 1#
.\$RAND 1#
.\$RDDE 1#
.\$RDOC 1#
.\$READ 1#
.\$R2AZ 1#
.\$SAVE 1#
.\$SB2D 1#
.\$SB2O 1#
.\$SCOP 1#
.\$SIZE 1#
.\$SUPR 1#
.\$TRAP 1#
.\$TYPB 1#
.\$TYPD 1#
.\$TYPE 1#
.\$TYPO 1#
.\$4OCA 1#
.1170 1#

. ABS. 006260 000

ERRORS DETECTED: 0

CZRPCD.BIN,CZRPCD.LST/CRF/SOL/NL:TOC=CZRPCD.SML,CZRPCD.P11
RUN-TIME: 89.4 SECONDS
RUN-TIME RATIO: 91/18-5.0
CORE USED: 33K (65 PAGES)