

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|------|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 | 41 | 42 | 43 | 44 | 45 | 46 | 47 | 48 | 49 | 50 | 51 | 52 | 53 | 54 | 55 | 56 | 57 | 58 | 59 | 60 | 61 | 62 | 63 | 64 | 65 | 66 | 67 | 68 | 69 | 70 | 71 | 72 | 73 | 74 | 75 | 76 | 77 | 78 | 79 | 80 | 81 | 82 | 83 | 84 | 85 | 86 | 87 | 88 | 89 | 90 | 91 | 92 | 93 | 94 | 95 | 96 | 97 | 98 | 99 | 100 |
| 101 | 102 | 103 | 104 | 105 | 106 | 107 | 108 | 109 | 110 | 111 | 112 | 113 | 114 | 115 | 116 | 117 | 118 | 119 | 120 | 121 | 122 | 123 | 124 | 125 | 126 | 127 | 128 | 129 | 130 | 131 | 132 | 133 | 134 | 135 | 136 | 137 | 138 | 139 | 140 | 141 | 142 | 143 | 144 | 145 | 146 | 147 | 148 | 149 | 150 | 151 | 152 | 153 | 154 | 155 | 156 | 157 | 158 | 159 | 160 | 161 | 162 | 163 | 164 | 165 | 166 | 167 | 168 | 169 | 170 | 171 | 172 | 173 | 174 | 175 | 176 | 177 | 178 | 179 | 180 | 181 | 182 | 183 | 184 | 185 | 186 | 187 | 188 | 189 | 190 | 191 | 192 | 193 | 194 | 195 | 196 | 197 | 198 | 199 | 200 |
| 201 | 202 | 203 | 204 | 205 | 206 | 207 | 208 | 209 | 210 | 211 | 212 | 213 | 214 | 215 | 216 | 217 | 218 | 219 | 220 | 221 | 222 | 223 | 224 | 225 | 226 | 227 | 228 | 229 | 230 | 231 | 232 | 233 | 234 | 235 | 236 | 237 | 238 | 239 | 240 | 241 | 242 | 243 | 244 | 245 | 246 | 247 | 248 | 249 | 250 | 251 | 252 | 253 | 254 | 255 | 256 | 257 | 258 | 259 | 260 | 261 | 262 | 263 | 264 | 265 | 266 | 267 | 268 | 269 | 270 | 271 | 272 | 273 | 274 | 275 | 276 | 277 | 278 | 279 | 280 | 281 | 282 | 283 | 284 | 285 | 286 | 287 | 288 | 289 | 290 | 291 | 292 | 293 | 294 | 295 | 296 | 297 | 298 | 299 | 300 |
| 301 | 302 | 303 | 304 | 305 | 306 | 307 | 308 | 309 | 310 | 311 | 312 | 313 | 314 | 315 | 316 | 317 | 318 | 319 | 320 | 321 | 322 | 323 | 324 | 325 | 326 | 327 | 328 | 329 | 330 | 331 | 332 | 333 | 334 | 335 | 336 | 337 | 338 | 339 | 340 | 341 | 342 | 343 | 344 | 345 | 346 | 347 | 348 | 349 | 350 | 351 | 352 | 353 | 354 | 355 | 356 | 357 | 358 | 359 | 360 | 361 | 362 | 363 | 364 | 365 | 366 | 367 | 368 | 369 | 370 | 371 | 372 | 373 | 374 | 375 | 376 | 377 | 378 | 379 | 380 | 381 | 382 | 383 | 384 | 385 | 386 | 387 | 388 | 389 | 390 | 391 | 392 | 393 | 394 | 395 | 396 | 397 | 398 | 399 | 400 |
| 401 | 402 | 403 | 404 | 405 | 406 | 407 | 408 | 409 | 410 | 411 | 412 | 413 | 414 | 415 | 416 | 417 | 418 | 419 | 420 | 421 | 422 | 423 | 424 | 425 | 426 | 427 | 428 | 429 | 430 | 431 | 432 | 433 | 434 | 435 | 436 | 437 | 438 | 439 | 440 | 441 | 442 | 443 | 444 | 445 | 446 | 447 | 448 | 449 | 450 | 451 | 452 | 453 | 454 | 455 | 456 | 457 | 458 | 459 | 460 | 461 | 462 | 463 | 464 | 465 | 466 | 467 | 468 | 469 | 470 | 471 | 472 | 473 | 474 | 475 | 476 | 477 | 478 | 479 | 480 | 481 | 482 | 483 | 484 | 485 | 486 | 487 | 488 | 489 | 490 | 491 | 492 | 493 | 494 | 495 | 496 | 497 | 498 | 499 | 500 |
| 501 | 502 | 503 | 504 | 505 | 506 | 507 | 508 | 509 | 510 | 511 | 512 | 513 | 514 | 515 | 516 | 517 | 518 | 519 | 520 | 521 | 522 | 523 | 524 | 525 | 526 | 527 | 528 | 529 | 530 | 531 | 532 | 533 | 534 | 535 | 536 | 537 | 538 | 539 | 540 | 541 | 542 | 543 | 544 | 545 | 546 | 547 | 548 | 549 | 550 | 551 | 552 | 553 | 554 | 555 | 556 | 557 | 558 | 559 | 560 | 561 | 562 | 563 | 564 | 565 | 566 | 567 | 568 | 569 | 570 | 571 | 572 | 573 | 574 | 575 | 576 | 577 | 578 | 579 | 580 | 581 | 582 | 583 | 584 | 585 | 586 | 587 | 588 | 589 | 590 | 591 | 592 | 593 | 594 | 595 | 596 | 597 | 598 | 599 | 600 |
| 601 | 602 | 603 | 604 | 605 | 606 | 607 | 608 | 609 | 610 | 611 | 612 | 613 | 614 | 615 | 616 | 617 | 618 | 619 | 620 | 621 | 622 | 623 | 624 | 625 | 626 | 627 | 628 | 629 | 630 | 631 | 632 | 633 | 634 | 635 | 636 | 637 | 638 | 639 | 640 | 641 | 642 | 643 | 644 | 645 | 646 | 647 | 648 | 649 | 650 | 651 | 652 | 653 | 654 | 655 | 656 | 657 | 658 | 659 | 660 | 661 | 662 | 663 | 664 | 665 | 666 | 667 | 668 | 669 | 670 | 671 | 672 | 673 | 674 | 675 | 676 | 677 | 678 | 679 | 680 | 681 | 682 | 683 | 684 | 685 | 686 | 687 | 688 | 689 | 690 | 691 | 692 | 693 | 694 | 695 | 696 | 697 | 698 | 699 | 700 |
| 701 | 702 | 703 | 704 | 705 | 706 | 707 | 708 | 709 | 710 | 711 | 712 | 713 | 714 | 715 | 716 | 717 | 718 | 719 | 720 | 721 | 722 | 723 | 724 | 725 | 726 | 727 | 728 | 729 | 730 | 731 | 732 | 733 | 734 | 735 | 736 | 737 | 738 | 739 | 740 | 741 | 742 | 743 | 744 | 745 | 746 | 747 | 748 | 749 | 750 | 751 | 752 | 753 | 754 | 755 | 756 | 757 | 758 | 759 | 760 | 761 | 762 | 763 | 764 | 765 | 766 | 767 | 768 | 769 | 770 | 771 | 772 | 773 | 774 | 775 | 776 | 777 | 778 | 779 | 780 | 781 | 782 | 783 | 784 | 785 | 786 | 787 | 788 | 789 | 790 | 791 | 792 | 793 | 794 | 795 | 796 | 797 | 798 | 799 | 800 |
| 801 | 802 | 803 | 804 | 805 | 806 | 807 | 808 | 809 | 810 | 811 | 812 | 813 | 814 | 815 | 816 | 817 | 818 | 819 | 820 | 821 | 822 | 823 | 824 | 825 | 826 | 827 | 828 | 829 | 830 | 831 | 832 | 833 | 834 | 835 | 836 | 837 | 838 | 839 | 840 | 841 | 842 | 843 | 844 | 845 | 846 | 847 | 848 | 849 | 850 | 851 | 852 | 853 | 854 | 855 | 856 | 857 | 858 | 859 | 860 | 861 | 862 | 863 | 864 | 865 | 866 | 867 | 868 | 869 | 870 | 871 | 872 | 873 | 874 | 875 | 876 | 877 | 878 | 879 | 880 | 881 | 882 | 883 | 884 | 885 | 886 | 887 | 888 | 889 | 890 | 891 | 892 | 893 | 894 | 895 | 896 | 897 | 898 | 899 | 900 |
| 901 | 902 | 903 | 904 | 905 | 906 | 907 | 908 | 909 | 910 | 911 | 912 | 913 | 914 | 915 | 916 | 917 | 918 | 919 | 920 | 921 | 922 | 923 | 924 | 925 | 926 | 927 | 928 | 929 | 930 | 931 | 932 | 933 | 934 | 935 | 936 | 937 | 938 | 939 | 940 | 941 | 942 | 943 | 944 | 945 | 946 | 947 | 948 | 949 | 950 | 951 | 952 | 953 | 954 | 955 | 956 | 957 | 958 | 959 | 960 | 961 | 962 | 963 | 964 | 965 | 966 | 967 | 968 | 969 | 970 | 971 | 972 | 973 | 974 | 975 | 976 | 977 | 978 | 979 | 980 | 981 | 982 | 983 | 984 | 985 | 986 | 987 | 988 | 989 | 990 | 991 | 992 | 993 | 994 | 995 | 996 | 997 | 998 | 999 | 1000 |

.REM 2

IDENTIFICATION

PRODUCT CODE: AC-E442A-MC
PRODUCT NAME: CVKDAAC PDT11/150 SYSTEM EXERCISER
PRODUCT DATE: SEPT 1978
MAINTAINER: DIAGNOSTIC ENGINEERING

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

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) 1978 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

| | | | |
|---------|-------|---------|---------|
| DIGITAL | PDP | UNIBUS | MASSBUS |
| DEC | DECUS | DECTAPE | |

TABLE OF CONTENTS

| | |
|-----|-------------------------------------|
| 1.0 | GENERAL PROGRAM INFORMATION. |
| 1.1 | PROGRAM PURPOSE (ABSTRACT). |
| 1.2 | SYSTEM REQUIREMENTS. |
| 1.3 | RELATED DOCUMENTS AND STANDARDS. |
| 1.4 | DIAGNOSTIC HIERARCHY PREREQUISITES. |
| 1.5 | ASSUMPTIONS. |
| 1.6 | RUNNING SYSTEMS PROGRAMS. |
| 2.0 | OPERATING INSTRUCTIONS. |
| 2.1 | LOADING AND STARTING PROCEDURES. |
| 2.2 | SPECIAL ENVIRONMENTS. |
| 2.3 | OPERATIONAL SWITCH SETTINGS |
| 2.4 | PROGRAM OPTIONS. |
| 2.5 | EXECUTION TIMES. |
| 2.6 | POWER FAIL. |
| 2.7 | COMPATABILITY TESTING. |
| 2.8 | COPY UTILITY. |
| 3.0 | ERROR INFORMATION. |
| 3.1 | ERROR REPORTING PROCEDURE. |
| 3.2 | ERROR HALTS. |
| 4.0 | PROGRESS & PERFORMANCE REPORTS. |
| 5.0 | DEVICE INFORMATION TABLES. |
| 6.0 | DEVICE CONNECTOR LAYOUT. |
| 7.0 | SUMMARY OF TESTS. |

1.0 GENERAL PROGRAM INFORMATION.

1.1 PROGRAM PURPOSE (ABSTRACT).

THE PDT-11/150 SYSTEM EXERCISER TESTS THE PROCESSORS ABILITY TO OPERATE ALL ITS PERIPHERALS IN INTERRUPT MODE AT THE SAME TIME WITH EMPHASIS ON THE DISK SUBSYSTEMS. IT SHOULD BE NOTED THAT IT IS NOT A DIAGNOSTIC OF THE PERIPHERALS BUT A SERIES OF SYSTEM INTERACTION TESTS.

THE PROGRAM IS DEFAULTED TO EXERCISE THE CLUSTER TERMINALS & THE COMM PORT IN INTERNAL LOOPBACK (MAINT) MODE. THE DEFAULT CAN BE CHANGED TO EXERCISE ANY OF THE CLUSTER TERMINALS AND/OR THE COMM PORT IN EXTERNAL LOOPBACK. SEE PROGRAM OPTIONS SEC. 2.4.

TESTING OF THE ACTUAL CLUSTER TERMINALS OR COMM DEVICE IS NOT PERFORMED. THESE DEVICES CAN ONLY BE TESTED EITHER IN EXT. OR INT. LOOPBACK.

THE PRINTER LOGIC WILL BE EXERCISED IF SIZING DETERMINES IT TO BE PRESENT OR IF DATA TERM RDY IS ASSERTED ON ITS CONNECTOR.

THE CONSOLE TERMINAL IS NOT TESTED.

AFTER THE MEMORY HAS BEEN TESTED & ALL THE PERIPHERALS ARE BEING EXERCISED & INTERRUPTING AT RANDOM, THE DISK SUBSYSTEM TESTS WILL BEGIN. SEE TEST SUMMARY SEC. 7.0.

THE PROGRAM CONTAINS A UTILITY WHICH CAN COPY THE SYSTEM EXERCISER DISK FROM DX0 ONTO A SCRATCH DISK IN DX1. THIS ENABLES THE OPERATOR TO CREATE BACKUP COPIES OF THE EXERCISER DISK. SEE SECTION 2.8.

THE PROGRAM ALSO CONTAINS A COMPATABILITY TESTING OPTION. SEE SEC. 2.7.

1.2 SYSTEM REQUIREMENTS.

HARDWARE REQUIREMENTS:

PDT-11/150 SYSTEM
8K MEMORY - MINIMUM
CONSOLE TERMINAL
EXTERNAL LOOPBACK CONNECTORS (OPTIONAL) FOR COMM & CLUSTER TERMINAL PORTS.
(SEE PROGRAM OPTIONS SEC. 2.4)
VT-100 CONSOLE MUST BE SETUP FOR JUMP SCROLL.

SOFTWARE REQUIREMENTS:

THIS EXERCISER IS DESIGNED TO RUN IN ANY OF THE FOLLOWING WAYS:

STAND ALONE
WITH APT MONITOR
WITH RT-11 MONITOR

IT IS NOT DESIGNED TO RUN WITH THE DIAGNOSTIC SUPERVISOR.

1.3 RELATED DOCUMENTS AND STANDARDS.

| | |
|--|----------------|
| DIAGNOSTIC ENGINEERING STANDARDS AND CONVENTIONS | 175-003-009-02 |
| APT | MD-11-DZZMA |
| SYSMAC | MD-11-DZQAC |

1.4 DIAGNOSTIC HIERARCHY PREREQUISITES

THIS EXERCISER ASSUMES THAT THE POWER UP SELF TEST RUNS ERROR FREE.

1.5 ASSUMPTIONS

THIS EXERCISER ASSUMES THAT THE OPERATOR HAS MODIFIED THE DEVICE MAP (\$DEVN) AT LOC. 1246 IF THE DEFAULTS DO NOT AGREE WITH THE ACTUAL SYSTEM CONFIGURATION. SEE PROGRAM OPTIONS SEC. 2.4.
THE PROGRAM ALSO ASSUMES THE SOFTWARE SWITCH REGISTER IS PROPERLY SET UP (SWREG) AT LOC. 176. SEE SEC. 2.3.

1.6 RUNNING SYSTEMS PROGRAMS

UNLESS THE SYSTEMS PROGRAMS ARE KNOWN TO INITIALIZE THE BAUD RATES OF EACH DEVICE THRU THE PARAMETER REGISTER, THE OPERATOR SHOULD POWER DOWN & UP AGAIN WHEN FINISHED RUNNING THIS EXERCISER. THIS IS TO RESTORE THE PDT-11/150 TO ITS DEFAULT BAUD RATES. OTHERWISE, THE DEVICES WILL ATTEMPT TO RUN AT WHATEVER BAUD RATES WERE LAST USED BY THIS EXERCISER.

2.0 OPERATING INSTRUCTIONS.

2.1 LOADING AND STARTING PROCEDURES.

USE STANDARD PROCEDURE FOR PDP-11 ABSOLUTE BINARY FORMATTED PAPER TAPE.
THE PROGRAM WILL AUTO-START AFTER LOADING.

THE SYSTEM EXERCISER DISK WILL AUTO-START AFTER BOOTING.

EXAMPLE OF STARTUP ASSUMING PRINTER NOT PRESENT, 24K MEMORY,
& TERM #2 NOT TO BE TESTED, COMM PORT EXT. LOOPBACK, TERM 1 & 3 INT. LOOPBACK.
(20007 IN \$DEVM, SEE SEC. 1.5):

PDT-11/150 SYSTEM EXERCISER

SWR - 000000 NEW = 110000 <CR> (HALT ON ERR, ENABLE PERFORMANCE REPORTS)

DEVM = 000017 NEW = <CR> (KEEP DEFAULTS)

24K MEMORY PRESENT
PRINTER NOT PRESENT
TERM #2 TESTING DROPPED

INSERT SCRATCH DISKS, TYPE 'P' FOR NORMAL TESTING
'240G' FOR NORMAL RESTARTS
'250G' TO COPY SYS EXERCISER DISK
'260G' FOR COMPATABILITY PASS 1: WRITE
'270G' FOR PASS 2: READ

(PROGRAM HALTS & WAITS FOR 'P' & CONTINUES....ABOVE NOTE & HALT OMITTED UNDER APT)

THE PROGRAM WILL BEGIN ACTUAL TESTING AT THIS POINT. (SEE SUMMARY OF TESTS SEC. 7.0)

DURING THE TESTS, THE PROGRAM WILL PRINT PROGRESS REPORTS (SEE SEC. 4.1) IF
BIT 12 IS SET IN THE SWREG (SEE SEC 2.3).
END OF PASS & TOTAL ERROR COUNT IS PRINTED AT THE CONCLUSION OF TESTING.
THE PROGRAM JUMPS TO THE BEGINNING & THE ABOVE SEQUENCE IS REPEATED UNTIL HALTED.

2.2 SPECIAL ENVIRONMENTS.

THIS DIAGNOSTIC FOLLOWS THE STANDARD PROCEDURE FOR RUNNING UNDER APT, RT-11 MONITORS, AS DESCRIBED IN THEIR RESPECTIVE PROCEDURES MANUAL AND SYSMAC PACKAGE.

2.3 OPERATIONAL SWITCH SETTINGS

THIS PROGRAM SUPPORTS THE DYNAMIC LOADING OF THE SOFTWARE SWITCH REGISTER (SWREG AT LOC. 176) FROM THE CONSOLE. THIS CAN BE ACCOMPLISHED BY DOING THE FOLLOWING:

- 1) TYPE CONTROL G <^G>; THIS WILL ALLOW THE CONSOLE TO ENTER DATA INTO LOC. 176 AT SELECTED POINTS WITHIN THE PROGRAM.
- 2) THE MACHINE WILL THEN TYPE: ' SWR=XXXXXX NEW=' (XXXXXX IS THE OCTAL CONTENTS OF THE SOFTWARE SWITCH REGISTER.)
- 3) AFTER THE 'NEW=' HAS BEEN TYPED THEN THE OPERATOR CAN DO ONE OF THE FOLLOWING AT THE CONSOLE:
 - A) TYPE A NUMBER TO BE LOADED INTO LOC. 176 FOLLOWED BY A <CR>. (ONLY NUMBERS BETWEEN 0-7 WILL BE ACCEPTED). LEADING ZEROS NEED NOT BE TYPED, AND IF MORE THAN 6 DIGITS ARE TYPED THE LAST 6 WILL BE USED. IF A <CR> IS THE FIRST KEY DEPRESSED THE SOFTWARE SWITCH REGISTER CONTENTS WILL NOT BE CHANGED.
 - B) IF A CONTROL U <^U> IS DEPRESSED THEN THE PROGRAM WILL SEND YOU BACK TO STEP 3.
 - C) IF THE INPUT CHARACTER IS NOT ONE OF THE CHARACTERS MENTIONED ABOVE THEN A QUESTION MARK (?) WILL BE TYPED FOLLOWED BY A CARRAGE RETURN AND A LINE FEED SEQUENCE THEN PROCEED FROM STEP 3 (ERASING ALL PREVIOUS INPUT).
- 4) THE DIAGNOSTIC WILL CONTINUE ON TYPING <CR>.

SOFTWARE SWITCH REGISTER OPTIONS (SWREG)

BIT 15 SET = 100000 = HALT ON ERROR
13 SET = 20000 = INHIBIT ERROR TYPEOUTS
12 SET = 10000 = ENABLE PERFORMANCE REPORTS
10 SET = 2000 = BELL ON ERROR

2.4 PROGRAM OPTIONS.

A DEVICE MAP (\$DEV) AT LOCATION 1246 (ENTERED BY A CONTROL-G & CONTROL-C) IS EXAMINED BY THE PROGRAM TO DETERMINE THE METHOD OF TESTING THE CLUSTER TERMINALS & THE COMM PORT.

THE DEFAULT VALUE = 000017 (INTERNAL LOOPBACK FOR COMM PORT & CLUSTER TERMINALS) IT CAN BE CHANGED TO DO THE FOLLOWING:

\$DEV (LOC 1246) ENTERED BY CONTROL-G & C

| | | |
|-------------|---------|--|
| BIT 15 SET= | 100000= | DROP PRINTER TESTS |
| 14 SET= | 40000= | DROP CLUSTER TERM #3 TESTS |
| 13 SET= | 20000= | #2 |
| 12 SET= | 10000= | #1 |
| 11 SET= | 4000= | DROP ASYNC COMM TESTS |
| 10 SET= | 2000= | DROP SYNC COMM TESTS |
| 9 SET= | 1000= | DROP DRIVE 1 TESTS |
| 8 SET= | 400= | DROP DRIVE 0 TESTS |
| 7 SET= | 200= | DROP CLOCK TESTS |
| | | |
| BIT 3 SET= | 10= | INT. LOOPBACK (MAINT MODE) FOR COMM PORT (DEFAULT) |
| 2 SET= | 4= | INT. LOOPBACK (MAINT MODE) FOR TERM #1 (DEFAULT) |
| 1 SET= | 2= | INT. LOOPBACK (MAINT MODE) FOR #2 (DEFAULT) |
| 0 SET= | 1= | INT. LOOPBACK (MAINT MODE) FOR #3 (DEFAULT) |
| | | |
| BIT 3 CLR= | | EXT. LOOPBACK FOR COMM PORT. |
| BIT 2 CLR= | | EXT. LOOPBACK FOR TERM #1. |
| BIT 1 CLR= | | EXT. LOOPBACK FOR TERM #2. |
| BIT 0 CLR= | | EXT. LOOPBACK FOR TERM #3. |

NOTES:

1. THE CONSOLE IS NOT TESTED.
2. BITS 15-7 SETTINGS WILL OVERRIDE BIT 3-0 SETTINGS.
3. THE PRINTER LOGIC WILL BE TESTED IF BIT 15 = 0 & THE DATA TERM READY IS ASSERTED ON THE CONNECTOR. (A PHYSICAL PRINTER IS NOT REQ'D)
4. A VT-50,52 MAY BE USED INSTEAD OF A PRINTER FOR A VISUAL CHECK.

2.5 EXECUTION TIMES.

ASSUMING ALL DEVICES & 2 DISK SUBSYSTEMS PRESENT:

1'ST PASS:

- 25 - 30 SEC TO STARTUP CLOCK, PRINTER, COMM DEVICE & CLUSTER TERMINALS.
- 10 - 40 SEC TO WRITE, READ & DATA COMPARE 1'ST TRACK ON BOTH DISKS.
- 5 - 60 SEC TO DO 10 RANDOM SEEKS ON EACH DISK BETWEEN 1'ST 10 TRACKS.

SUBSEQUENT PASSES:

- 30 SEC TO STARTUP CLOCK, PRINTER, COMM DEVICE & CLUSTER TERMINALS.
- 20 MIN TO WRITE, READ & DATA COMPARE ALL TRACKS ON BOTH DISKS.
- 5 MIN TO DO 500 RANDOM SEEKS ON EACH DISK BETWEEN ALL TRACKS.

2.6 POWER FAIL

THERE IS NO POWER FAIL AUTO-RESTART CAPABILITY IN THE PDT-11.

2.7 COMPATABILITY TESTING

PASS 1: ENTERED FROM A '260G'.
WILL WRITE ALL TRACKS & SECTORS ON SELECTED DRIVES
& HALT AFTER AN OPERATOR PROMPT. TYPING 'P' WILL BEGIN PASS 2.

PASS 2: ENTERED BY A 'P' AFTER THE ABOVE HALT, OR A '270G'.
WILL PERFORM SEQUENTIAL & RANDOM READS ONLY, ON THE SELECTED DRIVES.

2.8 COPY UTILITY

TO PROVIDE BACKUP DISKS, A UTILITY IS PROVIDED IN THE PROGRAM TO COPY THE SYSTEM EXERCISER DISK FROM DX0 TO DX1 IN THE FOLLOWING WAY:

1. BOOT THE EXERCISER NORMALLY FROM DX0.
2. ALLOW THE PROGRAM TO HALT AFTER THE WARNING MESSAGE TO USE SCRATCH DISKS.
3. AT THIS POINT, DO A '250G' TO ENTER THE COPY UTILITY.
4. THE OPERATOR WILL BE PROMPTED TO USE A SCRATCH DISK IN DX1 & TYPE 'P' TO PROCEED.
5. WHEN COMPLETED, THERE WILL BE A VERIFYING MESSAGE.
THE OPERATOR CAN THEN DO A 'P' TO COPY ANOTHER IN THE SAME MANNER,
OR A '240G' TO BEGIN NORMAL TESTING. (OPERATOR WILL BE PROMPTED)
6. THE COPY UTILITY CAN BE ENTERED AT ANY TIME BY DOING A 'BREAK' & '250G'.

3.0 ERROR INFORMATION.

3.1 ERROR REPORTING PROCEDURE.

TYPICAL ERROR PRINTOUTS ARE SHOWN BELOW:

```
TERM #3 DATA COMP ERR
ERROR #   ERR PC       EXPECT  RECVD
000011   006516       000200 000100

DX1 SOFT ERR - DATA COMP
DXTST #   ERR PC       RXCS    RXES    RXSA    EXPECT  RECVD  # RETRIES
000003   010636       100337 000230 003405 100520 100120      4

DX0 HARD ERR - AFTER WRITE CMD
ERROR # DXTST #   ERR PC       RXCS    RXES    TRACK  SECTOR  #RETRIES
000016 000001   011246       100337 000230    59     18      10
```

WHERE ALL VALUES TYPED ARE OCTAL EXCEPT :
#RETRIES, TRACK, & SECTOR WHICH ARE IN DECIMAL.

BITS 15,13, & 10 OF THE SWITCH REGISTER (SWREG) CONTROL THE
SEQUENCE OF EVENTS AFTER AN ERROR IS CAUGHT.

BIT 15 SET: CAUSES THE PROGRAM TO HALT IN THE ERROR ROUTINE.
IF THE PROGRAM IS CONTINUED, IT WILL PROCEED
FROM WHERE IT HALTED.

BIT 13 SET: DISABLES THE PRINTING OF THE ERROR MESSAGE.

BIT 10 SET: CAUSES THE BELL TO RING ON ERROR.

THE ERROR ROUTINE SUPPORTS THE CONTROL G <^G> FUNCTION.
REFER TO SECTION 2.3 FOR DETAILS.

NOTE: SINCE THERE ARE NO SPECIFIC TEST NUMBERS, APT WILL
REPORT ALL ERRORS AS TEST 0 ERRORS.

3.2 ERROR HALTS.

THE ONLY HALT IN THE EXERCISER IS IN THE ERROR ROUTINE, AND
IS EXECUTED ONLY IF BIT 15 OF THE SWITCH REGISTER (SWREG) IS SET
WHEN AN ERROR OCCURS.

4.0 PROGRESS & PERFORMANCE REPORTS

THE FOLLOWING REPORTS ARE ENABLED BIT 12 SET IN THE SWITCH REGISTER (SWREG LOC 176)
& WILL APPEAR FOLLOWING LOADING & STARTING AS DESCRIBED
IN SECTION 2.1 (ASSUMING ALL DEVICES ARE TO BE TESTED & NO ERRORS):

MEM TESTS DONE (ALL MEMORY FROM LOCATION 0 TO THE BEGINNING OF THE
MONITOR/ABS LOADER HAS BEEN TESTED.)

CLOCK RUNNING (100 INTERRUPTS HAVE BEEN DETECTED BEFORE
EXERCISING THE NEXT DEVICE.
THE CLOCK CONTINUES RUNNING IN INTERRUPT MODE.)

PRINTER RUNNING (5 LINES HAVE BEEN PRINTED WITH ERROR BIT CHECKING ONLY.
THE PRINTER RUNS CONTINUOUSLY IN INTERRUPT MODE AT 9600 BAUD.
ITS ACTUAL PERFORMANCE IS A VISUAL CHECK IF PRINTER CONNECTED.
NO OTHER CHECKING PERFORMED IF EXTERNAL LOOPBACK USED.)

SYNC COMM DONE (CHARACTERS 0 THRU 377 HAVE BEEN TRANSMITTED & RECEIVED WITH ODD PARITY ENABLED.
INTERRUPTS ARE THEN DISABLED & THE SYNC COMM IS NO LONGER EXERCISED.
THIS IS SO THAT THE ASYNC COMM CAN BE EXERCISED
FOR THE DURATION OF THE PASS.)

ASYNC COMM RUNNING (CHARS 0 THRU 377 HAVE BEEN TRANSMITTED & RECEIVED BEFORE
EXERCISING THE NEXT DEVICE.
IT CONTINUES RUNNING IN INTERRUPT MODE AT 9600 BAUD
WITH ODD PARITY ENABLED & REPEATS THE 0 THRU 377 CYCLE)

TERM #1 RUNNING (CHARACTERS 0 THRU 377 HAVE BEEN TRANSMITTED
& RECEIVED BEFORE EXERCISING THE NEXT DEVICE.
THE TERMINAL KEEPS RUNNING CONTINUOUSLY AT 2400 BAUD
IN INTERRUPT MODE & REPEATS THE 0 THRU 377 CYCLE)

TERM #2 RUNNING (SAME AS #1)
TERM #3 RUNNING (SAME AS #1)

DX0 TRK 0 DONE (DRIVE 0, TRACK 0 IS WRITTEN WITH A DATA PATTERN, READ & DATA COMPARE PERFORMED)
DX1 TRK 0 DONE (DRIVE 1, TRACK 0 IS WRITTEN WITH A DATA PATTERN, READ & DATA COMPARE PERFORMED)
DX0 TRK 1 DONE (ETC)
DX1 TRK 1 DONE (ETC UNTIL...)

DX0 DATA PATT DONE (ALL TRACKS HAVE BEEN WRITTEN WITH A DATA PATTERN IN INTERRUPT MODE.)
DX1 DATA PATT DONE (SAME AS DX0....TRK 0 ONLY FOR 1'ST PASS QUICK VERIFY)

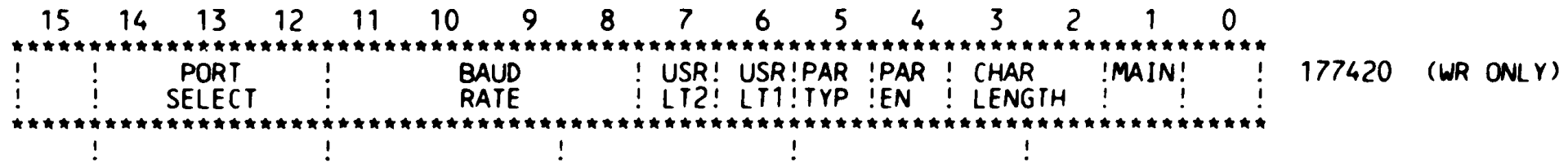
DX0 RANDOM SEEKS DONE (500 RANDOM SEEKS WITH READ & DATA COMPARE HAVE BEEN PERFORMED ON DRIVE 0)
DX1 RANDOM SEEKS DONE (SAME AS DX0....10 RANDOM SEEKS ONLY FOR 1'ST PASS QUICK VERIFY)

END OF PASS #1 TOTAL ERRORS: 0
TOTAL SOFT ERRORS: 0

TOTAL ERRORS THIS PASS: 0
SOFT ERRORS THIS PASS: 0 (...& ENTIRE PROCESS REPEATS)

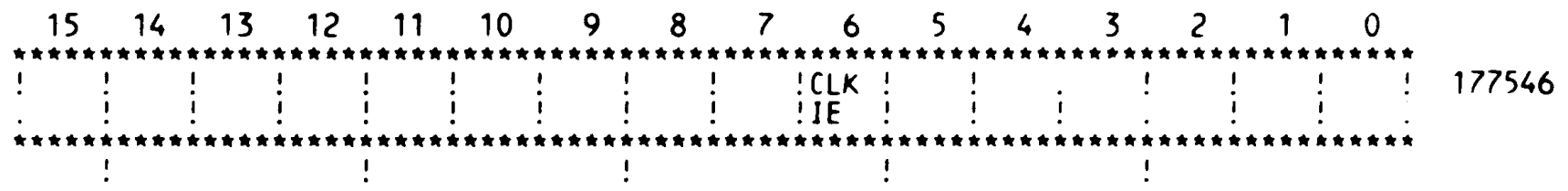
5.0 DEVICE REGISTERS.

PARAMETER REGISTER:

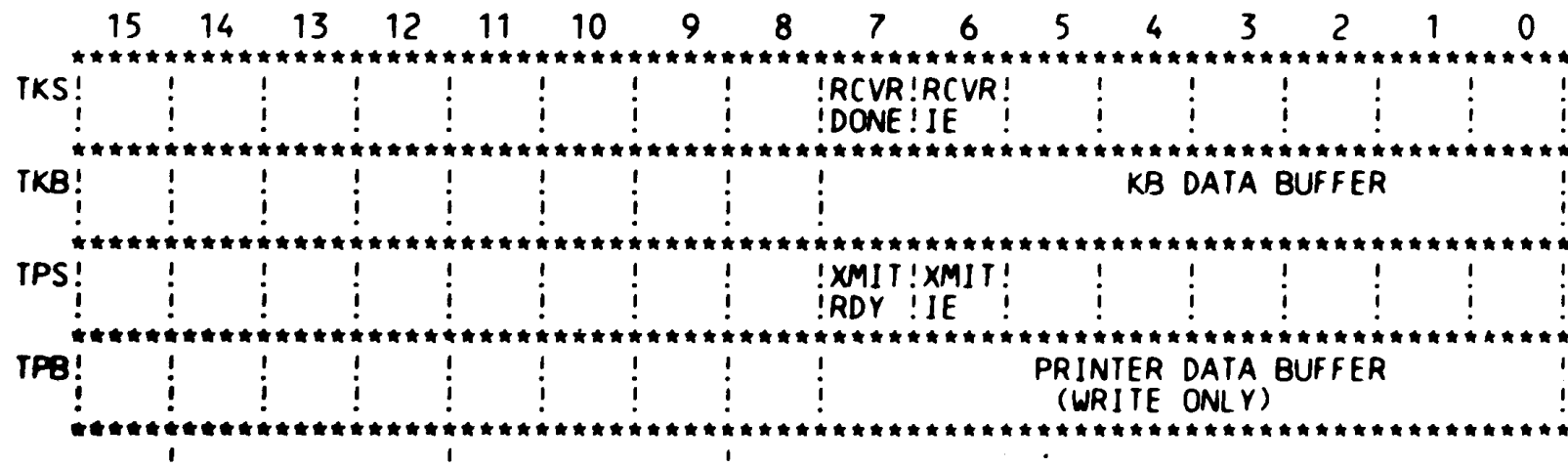


LINE CLOCK:

VECTOR: 100



CONSOLE: ADDR: 177560-177566 VECTOR: 60/64
 CLUSTER #1: 176500-176506 300/304
 #2: 176510-176516 310/314
 #3: 176520-176526 320/324



PRINTER:

VECTOR: 200

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|------|-----|----|----|----|----|----|---|---|------|------|---|---|---|---|---|---|-------------------------------------|
| PRSR | ERR | | | | | | | | PRIN | PRIN | | | | | | | 177514 |
| | | | | | | | | | RDY | IE | | | | | | | |
| PRB | | | | | | | | | | | | | | | | | 177516 |
| | | | | | | | | | | | | | | | | | PRINTER DATA BUFFER (WRITE ONLY) |

ASYNC MODEM:

VECTOR: 330/334

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|------|-----|------|-----|-----|----|-----|------|---|------|------|------|---|------|-----|-----|-----|---|
| RCSR | DSI | RING | CTS | CAR | | SEC | DSET | | RCVR | RCVR | DSET | | SEC | RTS | DTR | | 176610 |
| | | | | DET | | REC | RDY | | DONE | IE | IE | | XMIT | | | | |
| RBUF | ERR | OR | FR | PAR | | | | | | | | | | | | | 176612 |
| | | ERR | ERR | ERR | | | | | | | | | | | | | RECEIVER DATA BUFFER |
| XCSR | | | | | | | | | XMIT | XMIT | | | | | | BRK | 176614 |
| | | | | | | | | | RDY | IE | | | | | | | |
| XBUF | | | | | | | | | | | | | | | | | 176616 |
| | | | | | | | | | | | | | | | | | TRANSMITTER DATA BUFFER (WRITE ONLY) |

SYNC MODEM:

VECTOR: 340/344

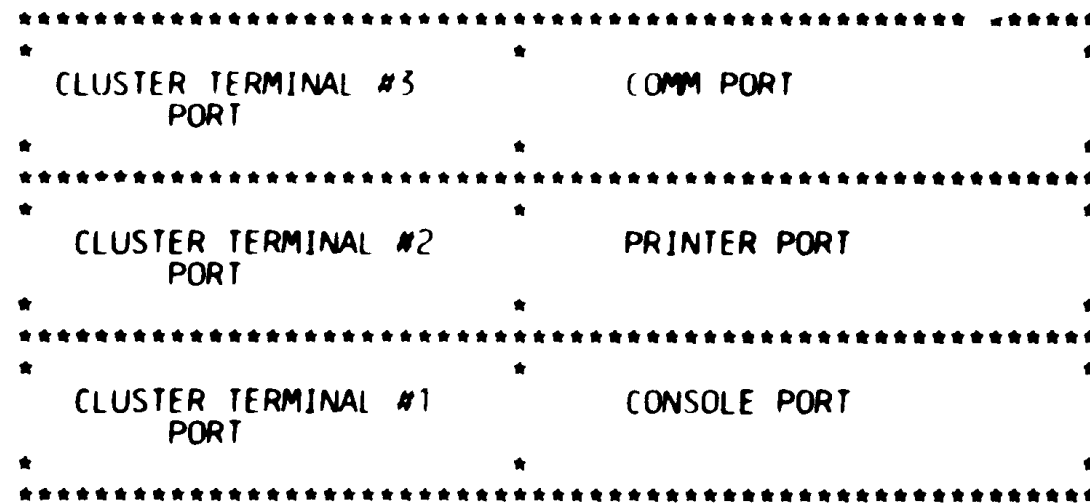
| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|------|-----|------|------|-----|--------|-----|------|------|------|------|------|------|------|-----|-----|---|---|
| RCSR | DSC | RING | CTS | CAR | REC | SEC | DSET | STRP | RCVR | RCVR | DSET | SRCH | SEC | RTS | DTR | | 176620 |
| | | | | DET | ACT | REC | RDY | SYNC | DONE | IE | IE | SYNC | XMIT | | | | |
| RBUF | ERR | OR | | PAR | | | | | | | | | | | | | 176622 (RD ONLY) |
| | | ERR | | ERR | | | | | | | | | | | | | RECEIVER DATA BUFFER |
| PCSR | | SYNC | | | WORD | PAR | EVEN | | | | | | | | | | 176622 (WR ONLY) |
| | | CHAR | | | LENGTH | IE | PAR | | | | | | | | | | SYNC REG |
| XCSR | DNA | | MA | MA | | | MAST | XMIT | XMIT | DNA | SEND | H/F | | | | | 176624 |
| | | | MODE | CLK | | | RS | RDY | IE | IE | | DUP | | | | | |
| XBUF | | | | | | | | | | | | | | | | | 176626 |
| | | | | | | | | | | | | | | | | | TRANSMITTER DATA BUFFER (WRITE ONLY) |

DISK:

VECTOR: 264

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|-------|------------------|----|----|----|----|----|---|------------------|-----------|-------------|-------------|-----|--------------|-------------|----|--------|--------|
| RXCS: | ERR | | | | | | | CTL DONE | DRV IE | | UNIT SEL | | FUNCTION | | GO | | 177170 |
| RXDB: | DATA BUFFER | | | | | | | | | | | | | | | 177172 | |
| RXFS: | | | | | | | | DRV RDY | | DEL DATA | RNF | CRC | LOST DATA | INV ADDR | ID | | 177172 |
| RXSA: | TRACK 0 - 114(8) | | | | | | | SECTOR 1 - 32(8) | | | | | | | | 177174 | |

6.0 DEVICE CONNECTOR LAYOUT




```

664
665 .TITLE PDT-11 EXERCISER
666 ;*COPYRIGHT (C) 1978
667 ;*DIGITAL EQUIPMENT CORP.
668 ;*MAYNARD, MASS. 01754
669 ;*
670 ;*PROGRAM BY GARY PAPAZIAN
671 ;*
672 ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
673 ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
674 ;*
675
676 .SBTTL OPERATIONAL SWITCH SETTINGS
677
678 .          SWITCH          USE
679 .-----
680 .          15             HALT ON ERROR
681 .          13             INHIBIT ERROR TYPEOUTS
682 .          12             ENABLE PERFORMANCE REPORTS
683 .          10             BELL ON ERROR
684 .
685
686
687
688 .SBTTL BASIC DEFINITIONS
689
690 ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
691 001100 STACK= 1100
692 .EQUIV EMT,ERROR          ;;BASIC DEFINITION OF ERROR CALL
693 .EQUIV IOT,SCOPE          ;;BASIC DEFINITION OF SCOPE CALL
694
695 ;*MISCELLANEOUS DEFINITIONS
696 000011 HT= 11             ;;CODE FOR HORIZONTAL TAB
697 000012 LF= 12             ;;CODE FOR LINE FEED
698 000015 CR= 15             ;;CODE FOR CARRIAGE RETURN
699 000200 CRLF= 200          ;;CODE FOR CARRIAGE RETURN-LINE FEED
700 177776 PS= 177776        ;;PROCESSOR STATUS WORD
701 .EQUIV PS,PSW
702 177774 STKLMT= 177774    ;;STACK LIMIT REGISTER
703 177772 PIRQ= 177772      ;;PROGRAM INTERRUPT REQUEST REGISTER
704 177570 DSWR= 177570     ;;HARDWARE SWITCH REGISTER
705 177570 DDISP= 177570    ;;HARDWARE DISPLAY REGISTER
706
707 ;*GENERAL PURPOSE REGISTER DEFINITIONS
708 000000 R0= %0             ;;GENERAL REGISTER
709 000001 R1= %1             ;;GENERAL REGISTER
710 000002 R2= %2             ;;GENERAL REGISTER
711 000003 R3= %3             ;;GENERAL REGISTER
712 000004 R4= %4             ;;GENERAL REGISTER
713 000005 R5= %5             ;;GENERAL REGISTER
714 000006 R6= %6             ;;GENERAL REGISTER
715 000007 R7= %7             ;;GENERAL REGISTER
716 000006 SP= %6             ;;STACK POINTER
717 000007 PC= %7             ;;PROGRAM COUNTER

```

```
718
719          ;*PRIORITY LEVEL DEFINITIONS
720          000000 PR0= 0          ;;PRIORITY LEVEL 0
721          000040 PR1= 40         ;;PRIORITY LEVEL 1
722          000100 PR2= 100        ;;PRIORITY LEVEL 2
723          000140 PR3= 140        ;;PRIORITY LEVEL 3
724          000200 PR4= 200        ;;PRIORITY LEVEL 4
725          000240 PR5= 240        ;;PRIORITY LEVEL 5
726          000300 PR6= 300        ;;PRIORITY LEVEL 6
727          000340 PR7= 340        ;;PRIORITY LEVEL 7
728
729          ;*'SWITCH REGISTER' SWITCH DEFINITIONS
730          100000 SW15= 100000
731          040000 SW14= 40000
732          020000 SW13= 20000
733          010000 SW12= 10000
734          004000 SW11= 4000
735          002000 SW10= 2000
736          001000 SW09= 1000
737          000400 SW08= 400
738          000200 SW07= 200
739          000100 SW06= 100
740          000040 SW05= 40
741          000020 SW04= 20
742          000010 SW03= 10
743          000004 SW02= 4
744          000002 SW01= 2
745          000001 SW00= 1
746          .EQUIV SW09,SW9
747          .EQUIV SW08,SW8
748          .EQUIV SW07,SW7
749          .EQUIV SW06,SW6
750          .EQUIV SW05,SW5
751          .EQUIV SW04,SW4
752          .EQUIV SW03,SW3
753          .EQUIV SW02,SW2
754          .EQUIV SW01,SW1
755          .EQUIV SW00,SW0
756
757          ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
758          100000 BIT15= 100000
759          040000 BIT14= 40000
760          020000 BIT13= 20000
761          010000 BIT12= 10000
762          004000 BIT11= 4000
763          002000 BIT10= 2000
764          001000 BIT09= 1000
765          000400 BIT08= 400
766          000200 BIT07= 200
767          000100 BIT06= 100
768          000040 BIT05= 40
769          000020 BIT04= 20
770          000010 BIT03= 10
771          000004 BIT02= 4
```

```
772      000002      BIT01= 2
773      000001      BIT00= 1
774      .EQUIV      BIT09,BIT9
775      .EQUIV      BIT08,BIT8
776      .EQUIV      BIT07,BIT7
777      .EQUIV      BIT06,BIT6
778      .EQUIV      BIT05,BIT5
779      .EQUIV      BIT04,BIT4
780      .EQUIV      BIT03,BIT3
781      .EQUIV      BIT02,BIT2
782      .EQUIV      BIT01,BIT1
783      .EQUIV      BIT00,BIT0
784
785      ;*BASIC 'CPU' TRAP VECTOR ADDRESSES
786      000004      ERRVEC= 4      ;;TIME OUT AND OTHER ERRORS
787      000010      RESVEC= 10     ;;RESERVED AND ILLEGAL INSTRUCTIONS
788      000014      TBITVEC=14     ;;'T' BIT
789      000014      TRTVEC= 14     ;;TRACE TRAP
790      000014      BPTVEC= 14     ;;BREAKPOINT TRAP (BPT)
791      000020      IOTVEC= 20     ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
792      000024      PWRVEC= 24     ;;POWER FAIL
793      000030      EMTVEC= 30     ;;EMULATOR TRAP (EMT) **ERROR**
794      000034      TRAPVEC=34     ;;'TRAP' TRAP
795      000060      TKVEC= 60      ;;TTY KEYBOARD VECTOR
796      000064      TPVEC= 64     ;;TTY PRINTER VECTOR
797      000240      PIRQVEC=240    ;;PROGRAM INTERRUPT REQUEST VECTOR
798
799
800
801
802      ;:*****
803      ;*      PDT-11/150 DEVICE VECTORS
804      ;:*****
805
806      000200      PRVEC= 200      ;PRINTER
807
808      000100      CLKVEC= 100     ;LINE CLOCK
809
810      000330      AMRVEC= 330     ;ASYNC MODEM RECVR
811      000334      AMXVEC= 334     ;          XMTR
812
813      000340      SMRVEC= 340     ;SYNC MODEM RECVR
814      000344      SMXVEC= 344     ;          XMTR
815
816      000264      RXVEC= 264     ;DISK
817
818      000300      TK1VEC= 300     ;CLUSTER TERM #1 VECTORS
819      000304      TP1VEC= 304
820      000310      TK2VEC= 310     ;          #2
821      000314      TP2VEC= 314
822      000320      TK3VEC= 320     ;          #3
823      000324      TP3VEC= 324
```

```
824
825      ;:*****
826      ;:      PDT-11/150 DEVICE REGISTERS
827      ;:*****
828
829      177420      PARAM= 177420      ;PARAMETER REGISTER      (WRITE ONLY)
830
831      177546      CLK= 177546      ;LINE CLOCK
832
833      177514      PRS= 177514      ;PRINTER STATUS REG
834      177516      PRB= 177516      ;      BUFFER      (WRITE ONLY)
835
836      176610      AMRC= 176610      ;ASYNC MODEM RECVR STATUS REG
837      176612      AMRB= 176612      ;      BUFFER
838      176614      AMXC= 176614      ;      XMITR STATUS REG
839      176616      AMXB= 176616      ;      BUFFER      (WRITE ONLY)
840
841      176620      SMRC= 176620      ;SYNC MODEM RECVR STATUS REG
842      176622      SMRB= 176622      ;      BUFFER      (READ ONLY)
843      176622      SMPAR= 176622      ;      PARAMETER REG      (WRITE ONLY)
844      176624      SMXC= 176624      ;      XMITR STATUS REG
845      176626      SMXB= 176626      ;      BUFFER      (WRITE ONLY)
846
847      177170      RXCS= 177170      ;DISK CSR
848      177172      RXDB= 177172      ;      DATA BUFF
849      177172      RXES= 177172      ;      ERROR & STATUS REG
850      177174      RXSA= 177174      ;      TRK & SECTOR ADDR REG
851
852      176500      TK1S= 176500      ;CLUSTER TERMINAL #1
853      176502      TK1B= 176502
854      176504      TP1S= 176504
855      176506      TP1B= 176506
856
857      176510      TK2S= 176510      ;      #2
858      176512      TK2B= 176512
859      176514      TP2S= 176514
860      176516      TP2B= 176516
861
862      176520      TK3S= 176520      ;      #3
863      176522      TK3B= 176522
864      176524      TP3S= 176524
865      176526      TP3B= 176526
```

```
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
```

```

:*****
:*      PARAMETER REGISTER EQUATES      WRITE ONLY
:*****

;BIT 14-12  PORT SELECT
:
CONSOL= 10000      ;SELECT CONSOLE TERMINAL
CT1=    0          ;      CLUSTER TERMINAL #1
CT2=   50000      ;      #2
CT3=   60000      ;      #3
PRT=   20000      ;      PRINTER PORT
AMOD=  30000      ;      ASYNC COMM PORT
ULITE=  40000      ;      USER LIGHTS

;BIT 11-8  BAUD RATE  (DELTA = 400)
:
B50=    0          ;50 BAUD
B75=    400        ;75
B110=   1000       ;110
B134=   1400       ;134.5
B150=   2000       ;150
B300=   2400       ;300
B600=   3000       ;600
B1200=  3400       ;1200
B1800=  4000       ;1800
B2000=  4400       ;2000
B2400=  5000       ;2400 (CLUSTER DEFAULT)
B3600=  5400       ;3600
B4800=  6000       ;4800
B7200=  6400       ;7200
B9600=  7000       ;9600 (PRINTER & ASYNC DEFAULT)
B19200= 7400       ;19200

;IF 110 BAUD SELECTED, 2 STOP BITS ARE ASSUMED.

;BITS 4-5  PARITY CONTROL
:
EPAR=   60         ;EVEN PARITY ENABLE
OPAR=   20         ;ODD PARITY ENABLE (DEFAULT)

;BITS 3-2  CHARACTER LENGTH (DELTA - 4)
:
CHAR5=  0          ;5 BITS/CHAR
CHAR6=  4          ;6
CHAR7=  10         ;7
CHAR8=  14         ;8 (PRINTER, TERMINAL, MODEM DEFAULT)

;BIT 1  MAINTENANCE BIT
:
MAINT=  BIT1       ;ENABLE MAINT MODE
                        ;PRINTER & DISK DO NOT HAVE MAINT MODE
```



```
920
921      ;*****
922      ;*      ASYNC MODEM BIT DEFINITIONS
923      ;*****
924
925      ;RCSR
926      :
927      100000 DSI= BIT15      ;DATA SET INTERRUPT
928      040000 RING= BIT14      ;RING
929      020000 CTS= BIT13      ;CLEAR TO SEND
930      010000 CDET= BIT12      ;CARRIER DETECTED
931      002000 SREC= BIT10      ;SECONDARY RECD/SUPERVISORY RECD
932      001000 DSRDY= BIT9      ;DATA SET RDY
933      000200 DONE= BIT7      ;RECVR DONE
934      000100 IE= BIT6      ;RECVR IE
935      000040 DSIE= BIT5      ;DATA SET IE
936      000010 SXMIT= BIT3      ;SECONDARY XMIT/SUPERV XMIT
937      000004 RTS= BIT2      ;REQ TO SEND
938      000002 DTR= BIT1      ;DATA TERMINAL RDY
939
940      ;RBUF
941      :
942      100000 ERR= BIT15      ;ERROR
943      040000 ORERR= BIT14      ;OVERRUN ERROR
944      020000 FRERR= BIT13      ;FRAMING ERROR
945      010000 PARERR= BIT12      ;PARITY ERROR
946
947      ;XCSR
948      :
949      000200 RDY= BIT7      ;XMIT RDY
950      000100 IE= BIT6      ;XMIT IE
951      000001 BREAK= BIT0      ;BREAK
952
```

```

953
954
955          :*****
956          :*          SYNC MODEM BIT DEFINITIONS
957          :*****
958          :RCSR
959          :
960          100000 DSC=   BIT15      ;DATA SET CHANGE
961          040000 RING=   BIT14      ;RING
962          020000 CTS=   BIT13      ;CLEAR TO SEND
963          010000 CDET=  BIT12      ;CARRIER DETECTED
964          004000 RACT=  BIT11      ;RECVR ACTIVE (SYNC DETECT)
965          002000 SREC=  BIT10      ;SECONDARY RECD/SUPERVISORY RECD
966          001000 DSRDY= BIT9       ;DATA SET RDY
967          000400 STSYN= BIT8       ;STRIP SYNC
968          000200 DONE=  BIT7       ;RECVR DONE
969          000100 IE=    BIT6       ;RECVR IE
970          000040 DSIE=  BIT5       ;DATA SET IE
971          000020 SRSYN= BIT4       ;SEARCH SYNC
972          000010 SXMIT= BIT3       ;SECONDARY XMIT/SUPERV XMIT
973          000004 RTS=   BIT2       ;REQ TO SEND
974          000002 DTR=   BIT1       ;DATA TERMINAL RDY
975
976          :RBUF
977          :
978          100000 ERR=   BIT15      ;ERROR
979          040000 ORERR= BIT14      ;OVERRUN ERROR
980          010000 PARERR= BIT12     ;PARITY ERROR
981
982          :PARCSR PARAMETER CONT REG
983          :
984          :BIT 14 SYNC CHAR
985          :
986          000000 SYNC2= 0          ;2 SYNC CHARS (DEFAULT)
987          040000 SYNC1= BIT14      ;1
988
989          :BITS 10-11 WORD LENGTH
990          :
991          000000 CHR5= 0          ;5 BITS/CHAR
992          002000 CHR6= 2000       ;6
993          004000 CHR7= 4000       ;7
994          006000 CHR8= 6000       ;8 (DEFAULT)
995
996          :BITS 8-9 PARITY CONTROL
997          :
998          001400 ENVPAR= 1400      ;ENABLE EVEN PARITY
999          001000 ODDPAR= 1000     ;ENABLE ODD PARITY (DEFAULT)
1000
1001          :XCSR
1002          :
1003          100000 DNA=   BIT15      ;DATA NOT AVAIL
1004          010000 MM=   BIT12      ;MAINT MODE
1005          004000 MCLK= BIT11      ;MAINT CLOCK
1006          000400 MR=   BIT8       ;MASTER RESET

```

```
1007      000200      RDY=   BIT7      :XMIT RDY
1008      000100      IE=    BIT6      :XMIT IE
1009      000040      DNAIE= BIT5      :DATA NOT AVAIL IE
1010      000020      SEND=  BIT4      :SEND
1011      000010      HFDUP= BIT3      :1=HALF DUPLEX, 0=FULL DUPLEX (DEFAULT = 0 FOR TESTS)
```

1012
1013
1014
1015

```
::*****
:DISK BIT DEFINITIONS
:*****
```

1019
1020

```
:RXCS
```

1021
1022 100000
1023 000200
1024 000100
1025 000020
1026 000020

```
:
ERR=   BIT15      :ERROR
DONE=  BIT7       :CONTR RDY
IE=    BIT6       :DRIVE IE
DX1=   BIT4       :UNIT SELECT: 0=DX0, 1=DX1
USEL=  BIT4       :UNIT SELECT
```

1027
1028

```
:FUNCTIONS (INCLUDES BIT0 = GO)
```

1029
1030 000001
1031 000003
1032 000005
1033 000007
1034 000011
1035 000013
1036 000015
1037 000017

```
FBUF=  1          :FILL BUFFER
EBUF=  3          :EMPTY BUFFER
WSEC=  5          :WRITE SECTOR
RSEC=  7          :READ SECTOR
INITAL= 11       :INITIALIZE
RSTAT= 13        :READ STATUS
WDDSEC= 15       :WRITE DELETED DATA SECTOR
RESTOR= 17       :RESTORE TO TRACK 0
```

1038
1039

```
:RXES: ERROR & STATUS
```

1040
1041 000200
1042 000040
1043 000020
1044 000010
1045 000004
1046 000002
1047 000001

```
:
RDY=   BIT7      :DRIVE RDY
DD=    BIT5      :DELETED DATA DET.
RNF=   BIT4      :RECORD NOT FOUND
CRC=   BIT3      :CRC ERROR
LDATA= BIT2      :LOST DATA
INVADR= BIT1     :INVALID ADDR
ID=    BIT0      :INITIALIZE DONE
```

1048
1049

```

1050
1051
1052      ;:*****
1053      ;*      SWITCH REGISTER
1054      ;:*****
1055      . =174
1056 000174 000000  DISPREG: .WORD 0      ;SOFTWARE DISPLAY REG
1057 000176 000000  SWREG:  .WORD 0      ;SOFTWARE SWITCH REG
1058
1059      . =100
1060 000100 000102  102      ;SETUP CLOCK VECTOR AREA TO DO RTI
1061 000102 000002  2      ;IF ENABLED
1062
1063      . =200
1064 000200 000137 003174  JMP      START      ;USE ONLY ONCE. WILL BE OVERLAID BY
1065      ;PRINTER VECTOR ADDR.
1066      . =240
1067 000240 000137 003174  JMP      START      ;RESTART ADDR
1068
1069      . =250
1070 000250 000137 003156  JMP      START3     ;ENTER HERE FOR COPY UTILITY
1071      . =260
1072 000260 000137 003136  JMP      START1     ;COMPATABILTIIY PASS 1
1073      . =270
1074 000270 000137 003144  JMP      START2     ;
1075      ;
1076      ;
1077      ;
1078      . =1000
1079      .SBTTL  APT PARAMETER BLOCK
1080
1081      ;:*****
1082      ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
1083      ;:*****
1084      . $X= .      ;;SAVE CURRENT LOCATION
1085      . =24      ;;SET POWER FAIL TO POINT TO START OF PROGRAM
1086 000024 000200  200      ;;FOR APT START UP
1087      . =44      ;;POINT TO APT INDIRECT ADDRESS PNTR.
1088 000044 001000  $APTHDR ;;POINT TO APT HEADER BLOCK
1089      . = $X      ;;RESET LOCATION COUNTER
1090      ;:*****
1091      ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
1092      ;INTERFACE SPEC.
1093
1094 001000  $APTHD:
1095 001000 000000  $HIBTS: .WORD 0      ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
1096 001002 001170  $MBADR: .WORD $MAIL  ;;ADDRESS OF APT MAILBOX (BITS 0-15)
1097 001004 000024  $STMT:  .WORD 20.    ;;RUN TIM OF LONGEST TEST
1098 001006 000226  $PASTM: .WORD 150.   ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
1099 001010 000000  $UNITM: .WORD 0      ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
1100 001012 000030  .WORD $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)

```

1101
1102
1103
1104
1105
1106
1107 001100
1108 001100 001100
1109 001100 000000
1110 001102 000
1111 001103 000
1112 001104 000000
1113 001106 000000
1114 001110 000000
1115 001112 000000
1116 001114 000
1117 001115 001
1118 001116 000000
1119 001120 000000
1120 001122 000000
1121 001124 000000
1122 001126 000000
1123 001130 000000
1124 001132 000000
1125 001134 000
1126 001135 000
1127 001136 000000
1128 001140 177570
1129 001142 177570
1130 001144 177560
1131 001146 177562
1132 001150 177564
1133 001152 177566
1134 001154 000
1135 001155 002
1136 001156 012
1137 001157 000
1138 001160 177607 000377
1139 001164 077
1140 001165 015
1141 001166 000012
1142
1143
1144
1145
1146
1147 001170
1148 001170 000000
1149 001172 000000
1150 001174 000000
1151 001176 000000
1152 001200 000000
1153 001202 000000
1154 001204 000000

.SBTTL COMMON TAGS

::*****
 ::*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
 ::*USED IN THE PROGRAM.

.=1100
 \$CMTAG: .WORD 0 ;:START OF COMMON TAGS
 \$STNM: .BYTE 0 ;:CONTAINS THE TEST NUMBER
 \$ERFLG: .BYTE 0 ;:CONTAINS ERROR FLAG
 \$ICNT: .WORD 0 ;:CONTAINS SUBTEST ITERATION COUNT
 \$LPADR: .WORD 0 ;:CONTAINS SCOPE LOOP ADDRESS
 \$LPERR: .WORD 0 ;:CONTAINS SCOPE RETURN FOR ERRORS
 \$ERTTL: .WORD 0 ;:CONTAINS TOTAL ERRORS DETECTED
 \$ITEMB: .BYTE 0 ;:CONTAINS ITEM CONTROL BYTE
 \$ERMAX: .BYTE 1 ;:CONTAINS MAX. ERRORS PER TEST
 \$ERRPC: .WORD 0 ;:CONTAINS PC OF LAST ERROR INSTRUCTION
 \$GDADR: .WORD 0 ;:CONTAINS ADDRESS OF 'GOOD' DATA
 \$BDADR: .WORD 0 ;:CONTAINS ADDRESS OF 'BAD' DATA
 \$GDDAT: .WORD 0 ;:CONTAINS 'GOOD' DATA
 \$BDDAT: .WORD 0 ;:CONTAINS 'BAD' DATA
 .WORD 0 ;:RESERVED--NOT TO BE USED
 \$AUTOB: .BYTE 0 ;:AUTOMATIC MODE INDICATOR
 \$INTAG: .BYTE 0 ;:INTERRUPT MODE INDICATOR
 \$SWR: .WORD DSWR ;:ADDRESS OF SWITCH REGISTER
 \$DISPLAY: .WORD DDISP ;:ADDRESS OF DISPLAY REGISTER
 \$TKS: 177560 ;:TTY KBD STATUS
 \$TKB: 177562 ;:TTY KBD BUFFER
 \$TPS: 177564 ;:TTY PRINTER STATUS REG. ADDRESS
 \$TPB: 177566 ;:TTY PRINTER BUFFER REG. ADDRESS
 \$NULL: .BYTE 0 ;:CONTAINS NULL CHARACTER FOR FILLS
 \$FILLS: .BYTE 2 ;:CONTAINS # OF FILLER CHARACTERS REQUIRED
 \$FILLC: .BYTE 12 ;:INSERT FILL CHARS. AFTER A 'LINE FEED'
 \$TPFLG: .BYTE 0 ;:'TERMINAL AVAILABLE' FLAG (BIT<07>=0-YES)
 \$BELL: .ASCIZ <207><377><377> ;:CODE FOR BELL
 \$QUES: .ASCII /?/ ;:QUESTION MARK
 \$CRLF: .ASCII <15> ;:CARRIAGE RETURN
 \$LF: .ASCIZ <12> ;:LINE FEED

.SBTTL APT MAILBOX-ETABLE

::*****
 .EVEN
 \$MAIL: ;:APT MAILBOX
 \$MSGTY: .WORD AMSGTY ;:MESSAGE TYPE CODE
 \$FATAL: .WORD AFATAL ;:FATAL ERROR NUMBER
 \$TESTN: .WORD ATESTN ;:TEST NUMBER
 \$PASS: .WORD APASS ;:PASS COUNT
 \$DEVCT: .WORD ADEVCT ;:DEVICE COUNT
 \$UNIT: .WORD AUNIT ;:I/O UNIT NUMBER
 \$MSGAD: .WORD AMSGAD ;:MESSAGE ADDRESS

| | | | | | |
|------|--------|--------|----------------|--------|--|
| 1155 | 001206 | 000000 | \$MSGLG: .WORD | AMSGLG | ::MESSAGE LENGTH |
| 1156 | 001210 | | \$ETABLE: | | ::APT ENVIRONMENT TABLE |
| 1157 | 001210 | 000 | \$ENV: .BYTE | AENV | ::ENVIRONMENT BYTE |
| 1158 | 001211 | 000 | \$ENVM: .BYTE | AENVM | ::ENVIRONMENT MODE BITS |
| 1159 | 001212 | 000000 | \$SWREG: .WORD | ASWREG | ::APT SWITCH REGISTER |
| 1160 | 001214 | 000000 | \$USWR: .WORD | AUSWR | ::USER SWITCHES |
| 1161 | 001216 | 000000 | \$CPUOP: .WORD | ACPUOP | ::CPU TYPE,OPTIONS |
| 1162 | | | .* | | BITS 15-11=CPU TYPE |
| 1163 | | | .* | | 11/04=01,11/05=02,11/20 03,11/40 04,11/45-05 |
| 1164 | | | .* | | 11/70=06,PDQ=07,Q=10 |
| 1165 | | | .* | | BIT 10=REAL TIME CLOCK |
| 1166 | | | .* | | BIT 9=FLOATING POINT PROCESSOR |
| 1167 | | | .* | | BIT 8=MEMORY MANAGEMENT |
| 1168 | 001220 | 000 | \$MAMS1: .BYTE | AMAMS1 | ::HIGH ADDRESS,M.S. BYTE |
| 1169 | 001221 | 000 | \$MTYP1: .BYTE | AMTYP1 | ::MEM. TYPE,BLK#1 |
| 1170 | | | .* | | MEM.TYPE BYTE -- (HIGH BYTE) |
| 1171 | | | .* | | 900 NSEC CORE=001 |
| 1172 | | | .* | | 300 NSEC BIPOLAR=002 |
| 1173 | | | .* | | 500 NSEC MOS=003 |
| 1174 | 001222 | 000000 | \$MADR1: .WORD | AMADR1 | ::HIGH ADDRESS,BLK#1 |
| 1175 | | | .* | | MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF 'TYPE' ABOVE |
| 1176 | 001224 | 000 | \$MAMS2: .BYTE | AMAMS2 | ::HIGH ADDRESS,M.S. BYTE |
| 1177 | 001225 | 000 | \$MTYP2: .BYTE | AMTYP2 | ::MEM. TYPE,BLK#2 |
| 1178 | 001226 | 000000 | \$MADR2: .WORD | AMADR2 | ::MEM.LAST ADDRESS,BLK#2 |
| 1179 | 001230 | 000 | \$MAMS3: .BYTE | AMAMS3 | ::HIGH ADDRESS,M.S.BYTE |
| 1180 | 001231 | 000 | \$MTYP3: .BYTE | AMTYP3 | ::MEM. TYPE,BLK#3 |
| 1181 | 001232 | 000000 | \$MADR3: .WORD | AMADR3 | ::MEM.LAST ADDRESS,BLK#3 |
| 1182 | 001234 | 000 | \$MAMS4: .BYTE | AMAMS4 | ::HIGH ADDRESS,M.S.BYTE |
| 1183 | 001235 | 000 | \$MTYP4: .BYTE | AMTYP4 | ::MEM. TYPE,BLK#4 |
| 1184 | 001236 | 000000 | \$MADR4: .WORD | AMADR4 | ::MEM.LAST ADDRESS,BLK#4 |
| 1185 | 001240 | 000300 | \$VECT1: .WORD | AVECT1 | ::INTERRUPT VECTOR#1,BUS PRIORITY#1 |
| 1186 | 001242 | 000000 | \$VECT2: .WORD | AVECT2 | ::INTERRUPT VECTOR#2BUS PRIORITY#2 |
| 1187 | 001244 | 176500 | \$BASE: .WORD | ABASE | ::BASE ADDRESS OF EQUIPMENT UNDER TEST |
| 1188 | 001246 | 000017 | \$DEV: .WORD | ADEV | ::DEVICE MAP |
| 1189 | 001250 | | \$TEND: | | |
| 1190 | | | .MEXIT | | |

| Index | Item | EM | DH | DT | DF |
|-------|--------|--------|--------|--------|----|
| 1191 | | | | | |
| 1192 | | | | | |
| 1193 | | | | | |
| 1194 | | | | | |
| 1195 | | | | | |
| 1196 | | | | | |
| 1197 | | | | | |
| 1198 | | | | | |
| 1199 | | | | | |
| 1200 | | | | | |
| 1201 | | | | | |
| 1202 | | | | | |
| 1203 | | | | | |
| 1204 | | | | | |
| 1205 | 001250 | | | | |
| 1206 | | | | | |
| 1207 | | | | | |
| 1208 | 001250 | 020322 | 023160 | 023566 | |
| 1209 | 001256 | 024062 | | | |
| 1210 | | | | | |
| 1211 | 001260 | 020322 | 023160 | 023566 | |
| 1212 | 001266 | 024062 | | | |
| 1213 | | | | | |
| 1214 | 001270 | 020341 | 023204 | 023576 | |
| 1215 | 001276 | 024062 | | | |
| 1216 | | | | | |
| 1217 | 001300 | 020322 | 023160 | 023566 | |
| 1218 | 001306 | 024062 | | | |
| 1219 | | | | | |
| 1220 | 001310 | 020341 | 023204 | 023576 | |
| 1221 | 001316 | 024062 | | | |
| 1222 | | | | | |
| 1223 | 001320 | 020363 | 023141 | 023560 | |
| 1224 | 001326 | 024062 | | | |
| 1225 | | | | | |
| 1226 | 001330 | 020427 | 023246 | 023612 | |
| 1227 | 001336 | 024062 | | | |
| 1228 | | | | | |
| 1229 | 001340 | 020452 | 023277 | 023622 | |
| 1230 | 001346 | 024062 | | | |
| 1231 | | | | | |
| 1232 | 001350 | 020500 | 023277 | 023634 | |
| 1233 | 001356 | 024062 | | | |
| 1234 | | | | | |
| 1235 | 001360 | 020526 | 023277 | 023646 | |
| 1236 | 001366 | 024062 | | | |
| 1237 | | | | | |
| 1238 | 001370 | 020554 | 023277 | 023660 | |
| 1239 | 001376 | 024062 | | | |
| 1240 | | | | | |
| 1241 | 001400 | 020605 | 023277 | 023660 | |
| 1242 | 001406 | 024062 | | | |
| 1243 | | | | | |
| 1244 | 001410 | 020635 | 023334 | 023672 | |

.SBTTL ERROR POINTER TABLE

.*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
 .*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
 .*LOCATION \$ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
 .*NOTE1: IF \$ITEMB IS 0 THE ONLY PERTINENT DATA IS (\$ERRPC).
 .*NOTE2: EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:

.* EM ::POINTS TO THE ERROR MESSAGE
 .* DH ::POINTS TO THE DATA HEADER
 .* DT ::POINTS TO THE DATA
 .* DF ::POINTS TO THE DATA FORMAT

\$ERRTB:
 :: GLOBAL DATA
 :ERR 1 EM1,DH2,DT2,DF
 :ERR 2 EM1,DH2,DT2,DF
 :ERR 3 EM2,DH3,DT3,DF
 :ERR 4 EM1,DH2,DT2,DF
 :ERR 5 EM2,DH3,DT3,DF
 :ERR 6 EM3,DH1,DT1,DF
 :ERR 7 EM4,DH4,DT4,DF
 :ERR 10 EM5,DH5,DT5,DF
 :ERR 11 EM6,DH5,DT6,DF
 :ERR 12 EM7,DH5,DT7,DF
 :ERR 13 EM8,DH5,DT8,DF
 :ERR 14 EM9,DH5,DT8,DF
 :ERR 15 EM10,DH6,DT9,DF

| | | | | | | |
|------|--------|--------|--------|--------|---------|-------------------|
| 1245 | 001416 | 024062 | | | | |
| 1246 | | | | | :ERR 16 | |
| 1247 | 001420 | 020662 | 023401 | 023710 | | EM11,DH7,DT10,DF1 |
| 1248 | 001426 | 024070 | | | | |
| 1249 | | | | | :ERR 17 | |
| 1250 | 001430 | 020662 | 023401 | 023710 | | EM11,DH7,DT10,DF1 |
| 1251 | 001436 | 024070 | | | | |
| 1252 | | | | | :ERR 20 | |
| 1253 | 001440 | 020700 | 023401 | 023710 | | EM13,DH7,DT10,DF1 |
| 1254 | 001446 | 024070 | | | | |
| 1255 | | | | | :ERR 21 | |
| 1256 | 001450 | 020731 | 023401 | 023710 | | EM14,DH7,DT10,DF1 |
| 1257 | 001456 | 024070 | | | | |
| 1258 | | | | | :ERR 22 | |
| 1259 | 001460 | 020762 | 023470 | 023732 | | EM15,DH8,DT11,DF2 |
| 1260 | 001466 | 024100 | | | | |
| 1261 | | | | | :ERR 23 | |
| 1262 | 001470 | 021013 | 023470 | 023732 | | EM16,DH8,DT11,DF2 |
| 1263 | 001476 | 024100 | | | | |
| 1264 | | | | | :ERR 24 | |
| 1265 | 001500 | 021044 | 023401 | 023754 | | EM17,DH7,DT12,DF1 |
| 1266 | 001506 | 024070 | | | | |
| 1267 | | | | | :ERR 25 | |
| 1268 | 001510 | 021076 | 023401 | 023754 | | EM18,DH7,DT12,DF1 |
| 1269 | 001516 | 024070 | | | | |
| 1270 | | | | | :ERR 26 | |
| 1271 | 001520 | 021130 | 023401 | 023754 | | EM19,DH7,DT12,DF1 |
| 1272 | 001526 | 024070 | | | | |
| 1273 | | | | | :ERR 27 | |
| 1274 | 001530 | 021161 | 023401 | 023754 | | EM20,DH7,DT12,DF1 |
| 1275 | 001536 | 024070 | | | | |
| 1276 | | | | | :ERR 30 | |
| 1277 | 001540 | 021212 | 023470 | 023776 | | EM21,DH8,DT13,DF2 |
| 1278 | 001546 | 024100 | | | | |
| 1279 | | | | | :ERR 31 | |
| 1280 | 001550 | 021243 | 023470 | 023776 | | EM22,DH8,DT13,DF2 |
| 1281 | 001556 | 024100 | | | | |
| 1282 | | | | | :ERR 32 | |
| 1283 | 001560 | 021274 | 023141 | 023560 | | EM23,DH1,DT1,DF |
| 1284 | 001566 | 024062 | | | | |
| 1285 | | | | | :ERR 33 | |
| 1286 | 001570 | 021305 | 023246 | 023612 | | EM24,DH4,DT4,DF |
| 1287 | 001576 | 024062 | | | | |
| 1288 | | | | | :ERR 34 | |
| 1289 | 001600 | 021322 | 023141 | 023560 | | EM25,DH1,DT1,DF |
| 1290 | 001606 | 024062 | | | | |
| 1291 | | | | | :ERR 35 | |
| 1292 | 001610 | 021337 | 023141 | 023560 | | EM26,DH1,DT1,DF |
| 1293 | 001616 | 024062 | | | | |
| 1294 | | | | | :ERR 36 | |
| 1295 | 001620 | 021354 | 023141 | 023560 | | EM27,DH1,DT1,DF |
| 1296 | 001626 | 024062 | | | | |

| | | | | | | |
|------|--------|--------|--------|--------|---------|-------------------|
| 1297 | | | | | :ERR 37 | |
| 1298 | 001630 | 021371 | 023141 | 023560 | | EM28,DH1,DT1,DF |
| 1299 | 001636 | 024062 | | | | |
| 1300 | | | | | :ERR 40 | |
| 1301 | 001640 | 021410 | 023141 | 023560 | | EM29,DH1,DT1,DF |
| 1302 | 001646 | 024062 | | | | |
| 1303 | | | | | :ERR 41 | |
| 1304 | 001650 | 021430 | 023334 | 024020 | | EM30,DH6,DT14,DF |
| 1305 | 001656 | 024062 | | | | |
| 1306 | | | | | :ERR 42 | |
| 1307 | 001660 | 021441 | 023334 | 024020 | | EM31,DH6,DT14,DF |
| 1308 | 001666 | 024062 | | | | |
| 1309 | | | | | :ERR 43 | |
| 1310 | 001670 | 021452 | 023141 | 023560 | | EM32,DH1,DT1,DF |
| 1311 | 001676 | 024062 | | | | |
| 1312 | | | | | :ERR 44 | |
| 1313 | 001700 | 021515 | 023141 | 023560 | | EM33,DH1,DT1,DF |
| 1314 | 001706 | 024062 | | | | |
| 1315 | | | | | :ERR 45 | |
| 1316 | 001710 | 021556 | 023141 | 023560 | | EM34,DH1,DT1,DF |
| 1317 | 001716 | 024062 | | | | |
| 1318 | | | | | :ERR 46 | |
| 1319 | 001720 | 021430 | 023334 | 024020 | | EM30,DH6,DT14,DF |
| 1320 | 001726 | 024062 | | | | |
| 1321 | | | | | :ERR 47 | |
| 1322 | 001730 | 021430 | 023334 | 024020 | | EM30,DH6,DT14,DF |
| 1323 | 001736 | 024062 | | | | |
| 1324 | | | | | :ERR 50 | |
| 1325 | 001740 | 021430 | 023334 | 024020 | | EM30,DH6,DT14,DF |
| 1326 | 001746 | 024062 | | | | |
| 1327 | | | | | :ERR 51 | |
| 1328 | 001750 | 021430 | 023334 | 024020 | | EM30,DH6,DT14,DF |
| 1329 | 001756 | 024062 | | | | |
| 1330 | | | | | :ERR 52 | |
| 1331 | 001760 | 021615 | 023334 | 023672 | | EM35,DH6,DT9,DF |
| 1332 | 001766 | 024062 | | | | |
| 1333 | | | | | :ERR 53 | |
| 1334 | 001770 | 021662 | 023470 | 023732 | | EM36,DH8,DT11,DF2 |
| 1335 | 001776 | 024100 | | | | |
| 1336 | | | | | :ERR 54 | |
| 1337 | 002000 | 021731 | 023334 | 023672 | | EM37,DH6,DT9,DF |
| 1338 | 002006 | 024062 | | | | |
| 1339 | | | | | :ERR 55 | |
| 1340 | 002010 | 021755 | 023334 | 023672 | | EM38,DH6,DT9,DF |
| 1341 | 002016 | 024062 | | | | |
| 1342 | | | | | :ERR 56 | |
| 1343 | 002020 | 021430 | 023334 | 024020 | | EM30,DH6,DT14,DF |
| 1344 | 002026 | 024062 | | | | |
| 1345 | | | | | :ERR 57 | |
| 1346 | 002030 | 022023 | 023334 | 023672 | | EM39,DH6,DT9,DF |
| 1347 | 002036 | 024062 | | | | |
| 1348 | | | | | :ERR 60 | |
| 1349 | 002040 | 021430 | 023334 | 024020 | | EM30,DH6,DT14,DF |
| 1350 | 002046 | 024062 | | | | |

| | | | | | | |
|------|--------|--------|--------|--------|---------|-------------------|
| 1351 | | | | | :ERR 61 | |
| 1352 | 002050 | 022053 | 023334 | 023672 | | EM40,DH6,DT9,DF |
| 1353 | 002056 | 024062 | | | | |
| 1354 | | | | | :ERR 62 | |
| 1355 | 002060 | 022116 | 023334 | 023672 | | EM41,DH6,DT9,DF |
| 1356 | 002066 | 024062 | | | | |
| 1357 | | | | | :ERR 63 | |
| 1358 | 002070 | 022152 | 023334 | 023672 | | EM42,DH6,DT9,DF |
| 1359 | 002076 | 024062 | | | | |
| 1360 | | | | | :ERR 64 | |
| 1361 | 002100 | 022173 | 023401 | 023710 | | EM43,DH7,DT10,DF1 |
| 1362 | 002106 | 024070 | | | | |
| 1363 | | | | | :ERR 65 | |
| 1364 | 002110 | 022173 | 023401 | 023710 | | EM43,DH7,DT10,DF1 |
| 1365 | 002116 | 024070 | | | | |
| 1366 | | | | | :ERR 66 | |
| 1367 | 002120 | 022225 | 023401 | 023710 | | EM45,DH7,DT10,DF1 |
| 1368 | 002126 | 024070 | | | | |
| 1369 | | | | | :ERR 67 | |
| 1370 | 002130 | 022263 | 023401 | 023710 | | EM46,DH7,DT10,DF1 |
| 1371 | 002136 | 024070 | | | | |
| 1372 | | | | | :ERR 70 | |
| 1373 | 002140 | 022321 | 023401 | 023754 | | EM47,DH7,DT12,DF1 |
| 1374 | 002146 | 024070 | | | | |
| 1375 | | | | | :ERR 71 | |
| 1376 | 002150 | 022360 | 023401 | 023754 | | EM48,DH7,DT12,DF1 |
| 1377 | 002156 | 024070 | | | | |
| 1378 | | | | | :ERR 72 | |
| 1379 | 002160 | 022417 | 023401 | 023754 | | EM49,DH7,DT12,DF1 |
| 1380 | 002166 | 024070 | | | | |
| 1381 | | | | | :ERR 73 | |
| 1382 | 002170 | 022455 | 023401 | 023754 | | EM50,DH7,DT12,DF1 |
| 1383 | 002176 | 024070 | | | | |
| 1384 | | | | | :ERR 74 | |
| 1385 | 002200 | 022513 | 023334 | 023672 | | EM51,DH6,DT9,DF |
| 1386 | 002206 | 024062 | | | | |
| 1387 | | | | | :ERR 75 | |
| 1388 | 002210 | 022537 | 023334 | 024020 | | EM52,DH6,DT14,DF |
| 1389 | 002216 | 024062 | | | | |
| 1390 | | | | | :ERR 76 | |
| 1391 | 002220 | 022563 | 023277 | 024036 | | EM53,DH5,DT15,DF |
| 1392 | 002226 | 024062 | | | | |
| 1393 | | | | | :ERR 77 | |
| 1394 | 002230 | 022563 | 023277 | 024050 | | EM53,DH5,DT16,DF |
| 1395 | 002236 | 024062 | | | | |

| | | | | | |
|------|--------|--------|--------|--------|-------------------------------------|
| 1396 | | | | | |
| 1397 | | | | | : ERRORS FOR COPY UTILITY |
| 1398 | | | | | |
| 1399 | | | | | :ERR 100 |
| 1400 | 002240 | 020700 | 023334 | 024020 | EM13,DH6,DT14,DF |
| 1401 | 002246 | 024062 | | | |
| 1402 | | | | | :ERR 101 |
| 1403 | 002250 | 021044 | 023334 | 024020 | EM17,DH6,DT14,DF |
| 1404 | 002256 | 024062 | | | |
| 1405 | | | | | :ERR 102 |
| 1406 | 002260 | 022632 | 023334 | 024020 | EM54,DH6,DT14,DF |
| 1407 | 002266 | 024062 | | | |
| 1408 | | | | | :ERR 103 |
| 1409 | 002270 | 021130 | 023334 | 024020 | EM19,DH6,DT14,DF |
| 1410 | 002276 | 024062 | | | |
| 1411 | | | | | |
| 1412 | | | | | :BAD SECTOR/TRACK TABLE FULL ERRORS |
| 1413 | | | | | |
| 1414 | | | | | :ERR 104 |
| 1415 | 002300 | 022647 | 000000 | 000000 | EM55,0,0,0 |
| 1416 | 002306 | 000000 | | | |
| 1417 | | | | | :ERR 105 |
| 1418 | 002310 | 022704 | 000000 | 000000 | EM56,0,0,0 |
| 1419 | 002316 | 000000 | | | |
| 1420 | | | | | |
| 1421 | | | | | :CRC ERRORS |
| 1422 | | | | | |
| 1423 | | | | | :ERR 106 |
| 1424 | 002320 | 022741 | 023401 | 023710 | EM57,DH7,DT10,DF1 |
| 1425 | 002326 | 024070 | | | |
| 1426 | | | | | :ERR 107 |
| 1427 | 002330 | 022762 | 023470 | 023732 | EM58,DH8,DT11,DF2 |
| 1428 | 002336 | 024100 | | | |
| 1429 | | | | | :ERR 110 |
| 1430 | 002340 | 023003 | 000000 | 000000 | EM59,0,0,0 |
| 1431 | 002346 | 000000 | | | |
| 1432 | | | | | :ERR 111 |
| 1433 | 002350 | 023041 | 023401 | 023754 | EM60,DH7,DT12,DF1 |
| 1434 | 002356 | 024070 | | | |
| 1435 | | | | | :ERR 112 |
| 1436 | 002360 | 023062 | 023470 | 023776 | EM61,DH8,DT13,DF2 |
| 1437 | 002366 | 024100 | | | |
| 1438 | | | | | :ERR 113 |
| 1439 | 002370 | 023103 | 000000 | 000000 | EM62,0,0,0 |
| 1440 | 002376 | 000000 | | | |

```
1441
1442
1443
1444
1445 002400 000000 DOPAT: 0 ;DX0 PATT TEST DONE FLAG
1446 002402 000000 DOCNT: 0 ;DX0 RANDOM SEEK COUNTER
1447 002404 000000 DOERR: 0 ;DX0 RCSR ERR COUNTER
1448 002406 000000 DOTRK: 0 ;DX0 TRACK
1449 002410 000000 DOSEC: 0 ;DX0 SECTOR
1450 002412 000000 DOCMER: 0 ;DX0 DATA COMPARE ERROR FLAG
1451 002414 000000 DOCERR: 0 ;DX0 TOTAL DATA COMP ERROR CT
1452 002416 000000 DOEXP: 0 ;DX0 EXPECTED DATA
1453 002420 000000 DOREC: 0 ;DX0 RECEIVED DATA
1454 002422 000000 DOCRC: 0 ;DX0 CRC ERROR FLAG
1455 002424 000100 DOBUF: .BLKW 100 ;DX0 DATA BUFFER
1456 002624 000012 DOBAD: .BLKW 10. ;DX0 BAD TRK/SEC TABLE
1457
1458
1459
1460
1461
1462
1463 002650 000000 D1PAT: 0 ;DX1 PATT TEST DONE FLAG
1464 002652 000000 D1CNT: 0 ;DX1 RANDOM SEEK COUNTER
1465 002654 000000 D1ERR: 0 ;DX1 RCSR ERR COUNTER
1466 002656 000000 D1TRK: 0 ;DX1 TRACK
1467 002660 000000 D1SEC: 0 ;DX1 SECTOR
1468 002662 000000 D1CMER: 0 ;DX1 DATA COMPARE ERROR FLAG
1469 002664 000000 D1CERR: 0 ;DX1 TOTAL DATA COMPARE ERROR CT
1470 002666 000000 D1EXP: 0 ;DX1 EXPECTED DATA
1471 002670 000000 D1REC: 0 ;DX1 RECEIVED DATA
1472 002672 000000 D1CRC: 0 ;DX1 CRC ERROR FLAG
1473 002674 000100 D1BUF: .BLKW 100 ;DX1 DATA BUFFER
1474 003074 000012 D1BAD: .BLKW 10. ;DX1 BAD TRK/SEC TABLE
1475
1476 003120 000000 DXTST: 0 ;1 = DATA PATT TEST
1477 ;2 = RANDOM SEEK TEST
1478 ;3 = INITIALIZE TEST
1479 ;4 = RESTORE TEST
1480 ;5 = WRITE DELETED DATA TEST
1481 ;6 = INVALID ADDRESS TEST
1482
1483 003122 000000 FIN: 0 ;COMMON FLG FOR DX0 & DX1 FOR TRK/SEC FINISHED
1484 003124 000000 MAXTRK: 0 ;MAX TRACK #
1485 ;0 FOR 1'ST PASS, 114 FOR SUBSEQUENT PASSES. SET AT START.
1486 003126 000000 MAXSK: 0 ;# OF RANDOM SEEKS TO BE PERFORMED.
1487 ;10 FOR 1'ST PASS, 500 FOR SUBSEQUENT PASSES. SET AT EOP.
1488 003130 000000 COPFLG: 0 ;0 = NORM TEST 1 = COPY UTILITY ACTIVE
1489 003132 000000 COMP1: 0 ;1 - COMPAT TESTS, PASS 1
1490 003134 000000 COMP2: 0 ;1 - PASS 2
1491
1492
```



```

1493      .SBTTL PROGRAM
1494
1495 003136 005237 003132 START1: INC COMP1 ;COMPAT PASS 1
1496 003142 000416 BR S1
1497
1498 003144 005237 003134 START2: INC COMP2 ;COMPAT PASS 2
1499 003150 005037 003132 CLR COMP1
1500 003154 000413 BR S2
1501
1502 003156 005237 003130 START3: INC COPFLG ;COPY UTILITY
1503 003162 005037 003132 CLR COMP1
1504 003166 005037 003134 CLR COMP2
1505 003172 000406 BR S3
1506
1507 003174 005037 003132 START: CLR COMP1 ;NORMAL TESTING
1508 003200 005037 003134 S1: CLR COMP2
1509 003204 005037 003130 S2: CLR COPFLG
1510
1511 003210 012706 001100 S3: MOV #STACK,SP ;SETUP STACK POINTER
1512 003214 042777 000100 175722 BIC #100,@$TKS ;DISABLE ANY TTY INTERRUPTS
1513 ;TO RT-11 MONITOR
1514 ;DISABLE INTERRUPTS
1515 003222 106427 ;MTPS #PR4
1516 003224 000200 .WORD 106427
1517 003226 013737 000042 004440 MOV 42,HLD42 ;SAVE
1518 003234 123727 001210 000001 CMPB $ENV,#1 ;APT?
1519 003242 001402 BEQ 2$ ;BR IF YES
1520 003244 005037 000042 CLR 42 ;ELSE CLR SO WE CAN USE SOFTSWR
1521 003250
1522 2$:
1523 .SBTTL INITIALIZE THE COMMON TAGS
1524 003250 012706 001100 ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
1525 003254 005026 MOV #CMTAG,R6 ;;FIRST LOCATION TO BE CLEARED
1526 003256 022706 001140 CLR (R6)+ ;;CLEAR MEMORY LOCATION
1527 003262 001374 CMP #SWR,R6 ;;DONE?
1528 003264 012706 001100 BNE -6 ;;LOOP BACK IF NO
1529 ;;INITIALIZE A FEW VECTORS
1530 003270 012737 025502 000030 MOV #ERROR,@EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
1531 003276 012737 000340 000032 MOV #340,@EMTVEC+2 ;;LEVEL 7
1532 003304 012737 026534 000034 MOV #STRAP,@TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
1533 003312 012737 000340 000036 MOV #340,@TRAPVEC+2;LEVEL 7
1534 ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
1535 ;;EQUAL TO A '-1', SETUP FOR A SOFTWARE SWITCH REGISTER.
1536 003320 013746 000004 MOV @ERRVEC,-(SP) ;;SAVE ERROR VECTOR
1537 003324 012737 003360 000004 MOV #64$,@ERRVEC ;;SET UP ERROR VECTOR
1538 003332 012737 177570 001140 MOV #DSWR,SWR ;;SETUP FOR A HARDWARE SWICH REGISTER
1539 003340 012737 177570 001142 MOV #DDISP,DISPLAY ;;AND A HARDWARE DISPLAY REGISTER
1540 003346 022777 177777 175564 CMP #-1,@SWR ;;TRY TO REFERENCE HARDWARE SWR
1541 003354 001012 BNE 66$ ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
1542 ;;AND THE HARDWARE SWR IS NOT = -1
1543 003356 000403 BR 65$ ;;BRANCH IF NO TIMEOUT
1544 003360 012716 003366 64$: MOV #65$,(SP) ;;SET UP FOR TRAP RETURN
1545 003364 000002 RTI
1546 003366 012737 000176 001140 65$: MOV #SWREG,SWR ;;POINT TO SOFTWARE SWR

```

```

1547 003374 012737 000174 001142      MOV    #DISPREG,DISPLAY
1548 003402 012637 000004                66$:  MOV    (SP)+, @#ERRVEC    ;;RESTORE ERROR VECTOR
1549
1550 003406 005037 001176                CLR    $PASS                ;;CLEAR PASS COUNT
1551 003412 132737 000200 001211      BITB   #APTSIZE,$ENVM       ;;TEST USER SIZE UNDER APT
1552 003420 001403                BEQ    67$                  ;;YES,USE NON-APT SWITCH
1553 003422 012737 001212 001140      MOV    #$$SWREG,$SWR       ;;NO,USE APT SWITCH REGISTER
1554 003430
1555
1556      .SBTTL  TYPE PROGRAM NAME
      ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
1557 003430 005227 177777                INC    #-1                  ;;FIRST TIME?
1558 003434 001042                BNE    68$                  ;;BRANCH IF NO
1559 003436 104401 003504                TYPE   ,69$                ;;TYPE ASCIZ STRING
1560      .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
1561 003442 005737 000042                TST    @#42                ;;ARE WE RUNNING UNDER XXDP/ACT?
1562 003446 001012                BNE    70$                  ;;BRANCH IF YES
1563 003450 123727 001210 000001      CMPB   $ENV,#1             ;;ARE WE RUNNING UNDER APT?
1564 003456 001406                BEQ    70$                  ;;BRANCH IF YES
1565 003460 023727 001140 000176      CMP    $SWR,#$SWREG        ;;SOFTWARE SWITCH REG SELECTED?
1566 003466 001005                BNE    71$                  ;;BRANCH IF NO
1567 003470 104406                GT$SWR                      ;;GET SOFT-SWR SETTINGS
1568 003472 000403                BR     71$
1569 003474 112737 000001 001134      70$:  MOV$B  #1,$AUTOB          ;;SET AUTO-MODE INDICATOR
1570 003502
1571 003502 000417                71$:  BR     68$                ;;GET OVER THE ASCIZ
1572      ;;69$:  .ASCIZ <CRLF>*PDT-11/150 SYSTEM EXERCISER*<CRLF>
1573      68$:
1574 003542 005737 003130                TST    COPFLG              ;;COPY UTILITY?
1575 003546 001057                BNE    LOOP                ;;BR IF YES
1576
1577 003550 005737 003132                TST    COMP1               ;;COMPAT PASS 1?
1578 003554 001054                BNE    LOOP                ;;BR IF YES
1579 003556 005737 003134                TST    COMP2               ;;COMPAT PASS 2?
1580 003562 001051                BNE    LOOP                ;;BR IF YES
1581
1582 003564 012737 000000 003124      MOV    #0,$MAXTRK          ;;TRK 0 ONLY FOR 1'ST PASS (PATCH-ABLE)
1583 003572 012737 000012 003126      MOV    #10,$MAXSK         ;;10 RANDOM SEEKS FOR 1'ST PASS
1584
1585 003600 005037 026052                CLR    TERR                ;;TOTAL ERR CT
1586 003604 005037 026054                CLR    TSERR               ;;TOTAL SOFT ERR CT FOR EOP TYPEOUT
1587
1588 003610 052737 000006 176610      BIS    #<RTS!DTR>,$AMRC
1589 003616 042737 000006 176610      BIC    #<RTS!DTR>,$AMRC    ;;DTR MUST BE TOGGLED.

```

```

1590
1591 003624 005227 177777          INC      #-1          ;1'ST TIME?
1592 003630 001005          BNE      4$          ;BR IF NO
1593 003632 123727 001210 000001  CMPB    $ENV,#1     ;APT?
1594 003640 001401          BEQ      4$          ;BR IF YES
1595 003642 104412          GTDEVM                    ;SHOW CURRENT DEVM & GET NEW
1596
1597 003644 004737 010256          4$:     JSR      PC,SIZE ;SEE WHO IS ON THE SYSTEM
1598 003650 123727 001210 000001  CMPB    $ENV,#1     ;ARE WE RUNNING UNDER APT?
1599 003656 001413          BEQ      LOOP        ;BR IF YES & DONT HALT
1600 003660 032737 001000 001246  BIT     #1000,$DEVM ;DX0 THERE?
1601 003666 001404          BEQ      1$          ;BR IF YES
1602 003670 032737 000400 001246  BIT     #400,$DEVM  ;DX1 THERE?
1603 003676 001003          BNE      LOOP        ;BR IF NO
1604 003700 104401 017215          1$:     TYPE    ,WARNING ;INSERT SCRATCH DISKS
1605 003704 000000          HALT
1606
1607
1608 003706 012706 001100          LOOP:  MOV     #1100,SP ;LOOP HERE AFTER EOP & RESET STACK
1609
1610 003712 004737 011430          JSR     PC,TIMER1   ;TIMER TO ALLOW 'P' TO PRINT BEFORE RESET
1611
1612 003716 000005          RESET
1613 003720 042737 000100 177546  BIC     #IE,CLK     ;RESET SETS CLK IE. DONT ALLOW YET
1614
1615          ;MTPS    #PRO          ;ALLOW INTERRUPTS
1616 003726 106427          .WORD  106427
1617 003730 000000          PRO
1618
1619 003732 052737 000006 176610  BIS     #<RTS.DTR> ,AMRC ;TO CONDITION ALL TERMINALS &
1620          ;COMM PORT TO OPERATE IN EXT LOOPBACK MODE.
1621          ;NO HARM IF LOOPBACK NOT USED.
1622 003740 004737 011430          JSR     PC,TIMER1   ;TIMER TO ALLOW PREV XFERS TO FINISH
1623
1624 003744 005037 026056          CLR     PERR        ;PASS ERR COUNT FOR EOP TYPEOUT
1625 003750 005037 026060          CLR     PSERR       ;PASS SOFT ERR COUNT FOR EOP TYPEOUT
1626
1627 003754 004737 011526          JSR     PC,CLRBAD   ;CLEAR BAD SECTOR/TRACK TABLES
1628
1629 003760 005737 003132          TST     COMP1        ;COMPAT PASS 1?
1630 003764 001004          BNE     1$          ;BR IF YES
1631 003766 005737 003134          TST     COMP2        ;COMPAT PASS 2?
1632 003772 001001          BNE     1$          ;BR IF YES
1633 003774 000405          BR      2$          ;ELSE TEST FOR COPY
1634
1635 003776 012737 000114 003124  1$:     MOV     #114,MAXTRK
1636 004004 000137 006300          JMP     DXST1        ;COMPATABILITY TESTING
1637
1638 004010 005737 003130          2$:     TST     COPFLG    ;COPY UTILITY?
1639 004014 001402          BEQ     MEMTST       ;BR IF NO...NORMAL TESTING
1640 004016 000137 016220          JMP     COPY         ;ELSE GO COPY
1641

```

```

1642 .SBTTL PROGRAM
1643 :*****
1644 :      MEMORY TESTS
1645 :
1646 :MEMORY FROM 0 TO THE LAST LOC OF PROGRAM (LSTAD) IS TESTED FOR TIMEOUT.
1647 :MEMORY FROM 'LSTAD' TO THE BOTTOM OF THE RT11 MONITOR/ABS LOADER
1648 :IS TESTED BY A PASS OF A 1010 PATT FOLLOWED BY A PASS OF 0101 PATTERN.
1649 :IF THE SYSTEM HAS A 28K MEMORY, 28K TO 30K WILL BE TESTED SIMILARLY.
1650 :*****
1651
1652 004022 004737 010744      MEMTST: JSR      PC,EXAMKB
1653
1654 004026 012737 012004 000004      MOV      #INTSRV,ERRVEC ;SETUP TIMEOUT TRAP ADDR
1655 004034 012737 000200 000006      MOV      #200,ERRVEC+2
1656
1657 004042 013703 010630      MOV      MAXMEM,R3
1658 004046 162703 000300      SJB      #300,R3      ;TEST ONLY AS FAR AS ABS LOADER IF NOT RT-11
1659
1660 004052 005004
1661 004054 005037 012012      1$:     CLR      R4      ;ADDR POINTER
1662 004060 005714      CLR      INTFLG
1663 004062 005737 012012      TST      (R4)      ;TEST LOCS 0 THRU END OF PGM.
1664 004066 001403      TST      INTFLG      ;TIMEOUT?
1665 004070 010437 004436      BEQ      2$      ;BR IF NO
1666 004074 104001      MOV      R4,ADDR      ;FOR ERR TYP
1667 004076 062704 000002      ERROR   1      ;TIMEOUT ON READ
1668 004102 020427 026616      2$:     ADD      #2,R4      ;BUMP ADDR
1669 004106 001362      CMP      R4,#LSTAD      ;AT END?
1670      BNE      1$      ;BR IF NO
1671 004110 005037 004430      CLR      MEMCT
1672 004114 012737 125252 004434      MOV      #125252,PAT ;DATA PATT FOR 1'ST PASS
1673 004122 012704 026616      8$:     MOV      #LSTAD,R4 ;ADDR POINTER. R/W LOCS LSTAD THRU
1674      ;BOT OF RT11 OR BOT OF ABS LOADER
1675 004126 005037 012012      3$:     CLR      INTFLG
1676 004132 013714 004434      MOV      PAT,(R4) ;WRITE
1677 004136 005737 012012      TST      INTFLG      ;TIMEOUT?
1678 004142 001403      BEQ      4$      ;BR IF NO
1679 004144 010437 004436      MOV      R4,ADDR      ;FOR ERR TYP
1680 004150 104002      ERROR   2      ;TIMEOUT ON WRITE
1681 004152 011437 004432      4$:     MOV      (R4),MEMHLD ;READ
1682 004156 023737 004432 004434      CMP      MEMHLD,PAT ;COMPARE
1683 004164 001403      BEQ      6$
1684 004166 010437 004436      MOV      R4,ADDR      ;FOR ERR TYP
1685 004172 104003      ERROR   3      ;COMPARE ERROR
1686
1687 004174 062704 000002      6$:     ADD      #2,R4      ;BUMP POINTER
1688 004200 023727 004440 001000      CMP      HLD42,#1000 ;RT11 IF LOC 42 - 1000
1689 004206 001403      BEQ      7$      ;BR IF RT11
1690
1691 004210 020403      CMP      R4,R3      ;ELSE AT BOT OF ABS LOADER?
1692 004212 001345      BNE      3$      ;BR IN NO
1693 004214 000403      BR      5$      ;ELSE EXIT

```

```

1694
1695 004216 020437 000054      7$:  CMP      R4,54      ;LOC 54 CONTAINS LOW ADDR OF RT11 MONITOR
1696 004222 001341                BNE      3$          ;BR IN NOT THERE
1697 004224 005737 004430      5$:  TST      MEMCT
1698 004230 001006                BNE      9$
1699 004232 005237 004430      INC      MEMCT
1700 004236 012737 052525 004434  MOV      #052525,PAT ;DATA PATT FOR 2'ND PASS
1701 004244 000726                BR       8$          ;REPEAT
1702
1703 004246 023727 010630 157776 9$:  CMP      MAXMEM,#157776 ;DO WE HAVE 28K?
1704 004254 001047                BNE      15$         ;BR IF NO & EXIT
1705
1706 004256 012737 125252 004434  MOV      #125252,PAT ;DATA PATT FOR 1'ST PASS OF HI MEM
1707 004264 012704 160000      10$: MOV      #160000,R4 ;ADDR PTR TO TOP OF 28K
1708 004270 005037 012012      11$: CLR      INTFLG
1709 004274 013714 004434      MOV      PAT,(R4)    ;WRITE
1710 004300 005737 012012      TST      INTFLG
1711 004304 001403                BEQ      12$
1712 004306 010437 004436      MOV      R4,ADDR    ;FOR ERR TYP
1713 004312 104004                FRROR   4           ;TIMEOUT
1714 004314 011437 004432      12$: MOV      (R4),MEMHLD ;READ
1715 004320 023737 004432 004434  CMP      MEMHLD,PAT ;COMPARE
1716 004326 001403                BEQ      14$
1717 004330 010437 004436      MOV      R4,ADDR    ;FOR ERR TYP
1718 004334 104005                ERROR   5           ;COMPARE ERROR
1719 004336 062704 000002      14$: ADD      #2,R4
1720 004342 020427 167776      CMP      R4,#167776 ;AT END OF 30K?
1721 004346 001350                BNE      11$         ;BR IF NO
1722
1723 004350 005237 004430      INC      MEMCT
1724 004354 023727 004430 000003  CMP      MEMCT,#3
1725 004362 001404                BEQ      15$
1726 004364 012737 052525 004434  MOV      #052525,PAT ;DATA PATT FOR 2'ND HALF OF HI MEM
1727 004372 000734                BR       10$        ;REPEAT
1728
1729 004374 012737 000006 000004 15$: MOV      #ERRVEC+2,ERRVEC ;RESTORE TIMEOUT VECTOR
1730 004402 005037 000006      CLR      ERRVEC+2
1731
1732 004406 032777 010000 174524  BIT      #BIT12,@SWR ;SKIP TYPEOUT IF NOT SET
1733 004414 001412                BEQ      CLKTST
1734 004416 104401 017607      TYPE   ,MEMORY
1735 004422 104401 017622      TYPE   ,DUN
1736 004426 000405                BR       CLKTST
1737
1738 004430 000000      MEMCT: 0           ;0 = MEM FROM LSTAD TO MONITOR/ABS LOADER W/ 125252 PATT
1739                                     ;1 = SAME WITH 052525 PATT
1740                                     ;1 - MEM FROM 28K TO 30K WITH 125252 PAT
1741                                     ;2 = SAME WITH 052525 PATT
1742                                     ;3 = EXIT
1743 004432 000000      MEMHLD: 0         ;PUT MEMORY READ HERE
1744 004434 000000      PAT: 0           ;PATT TO BE WRITTEN & READ
1745 004436 000000      ADDR: 0         ;ADDR UNDER TEST
1746 004440 000000      HLD42: 0        ;SAVE LOC 42
1747

```

```
1748  
1749  
1750  
1751  
1752 004442 032737 000200 001246 CLKTST: BIT #BIT7,$DEV ;DROP TEST?  
1753 004450 001031 BNE PRST ;BR IF YES  
1754 004452 012737 012004 000100 MOV #INTSRV,CLKVEC ;SETUP VECTOR AREA  
1755 004460 012737 000200 000102 MOV #200,CLKVEC+2 ;LOCKOUT OTHER INTER.  
1756 004466 005037 012012 CLR INTFLG  
1757 004472 052737 000100 177546 BIS #IE,CLK ;SET CLOCK LOOSE!  
1758  
1759 004500 004537 011160 JSR R5,TIMER ;SEE IF INTFLG GOES TO 100  
1760 004504 012012 INTFLG  
1761 004506 000100 100  
1762 004510 104032 ERROR 32 ;CLK NOT RESPONDING  
1763 004512 000410 BR PRST  
1764  
1765 004514 032777 010000 174416 BIT #BIT12,@SWR ;SKIP TYPEOUT IF NOT SET  
1766 004522 001404 BEQ PRST  
1767 004524 104401 017110 TYPE ,CLOCK  
1768 004530 104401 017534 TYPE ,RUN
```

```

1769
1770
1771
1772
1773
1774 004534 032737 100000 001246 PRTST: BIT #BIT15,$DEV  ;DROP TEST?
1775 004542 001074 BNE SMTST ;BR IF YES
1776 004544 005737 010610 TST PRPRES ;PRINTER THERE?
1777 004550 001014 BNE 1$ ;BR IF YES
1778
1779 004552 042737 000006 176610 BIC #<RTS!DTR>,AMRC ;MUST BE TOGGLED
1780 004560 052737 000006 176610 BIS #<RTS!DTR>,AMRC ;ELSE FORCE DTR IF LOOPBACK CONN INSTALLED
1781 004566 005737 177514 TST PRS ;NOW OK?
1782 004572 100460 BMI SMTST ;EXIT IF DTR HAS NO EFFECT...NO LOOPBACK CONN.
1783 004574 005237 012146 INC PRPORT ;ELSE SET FLAG TO DO JUST 5 LINES
1784 004600 000402 BR 2$
1785
1786 004602 005037 012146 1$: CLR PRPORT
1787 004606 012737 027014 177420 2$: MOV #<PRT!B9600!CHAR8>,PARAM ;SETUP BAUD RATE & CHAR LENGTH
1788 004614 012737 012014 000200 MOV #PRSRV,PRVEC ;ELSE SETUP PRINTER VECTOR
1789 004622 012737 000200 000202 MOV #200,PRVEC+2 ;LOCKOUT INTERR
1790
1791 004630 005037 012142 CLR LINCT ;CLEAR LINE CTR
1792 004634 012737 000015 012140 MOV #CR,PRCHR ;CHAR TO BE PRINTED
1793 ;'DEL' (177) CAN BE USED TO CLR PRINTER BUFF
1794 004642 052737 000100 177514 BIS #IE,PRS ;SET PRINTER LOOSE!
1795 ;WILL PRINT CONTINUOUS 132 COLUMN LINES
1796 ;EA BEGINNING WITH ASCII 40 THRU 176
1797 004650 004537 011160 JSR R5,TIMER ;SEE IF LINCT GOES TO 5
1798 004654 012142 LINCT
1799 004656 000005 5
1800 004660 104033 ERROR 33 ;PRINTER NOT RESPONDING
1801 004662 000424 BR SMTST
1802
1803 004664 005737 012146 TST PRPORT ;5 LINES ONLY?
1804 004670 001403 BEQ 3$ ;BR IF NO
1805 004672 042737 000100 177514 BIC #IE,PRS ;ELSE SHUT DOWN PRINTER
1806
1807 004700 032777 010000 174232 3$: BIT #BIT12,@SWR ;SKIP TYPEOUT IF NOT SET
1808 004706 001412 BEQ SMTST
1809 004710 104401 017050 TYPE ,PRINT
1810 004714 005737 012146 TST PRPORT ;5 LINES ONLY?
1811 004720 001403 BEQ 4$ ;BR IF NO
1812 004722 104401 017061 TYPE ,PORT
1813 004726 000402 BR SMTST
1814 004730 104401 017534 4$: TYPE ,RUN

```

```

1815
1816
1817          ::*****
1818          ::          START SYNC COMM PORT
1819          ::*****
1820 004734 032737 002000 001246 SMTST: BIT    #BIT10,$DEV0    ;DROP TEST?
1821 004742 001402                BEQ    20$              ;BR IF NO
1822 004744 000137 005222                JMP    AMTST          ;ELSE JUMP TEST
1823
1824 004750 005037 005220                20$:  CLR    FREE
1825
1826 004754 005037 012636                8$:   CLR    LSTCHR    ;LAST CHAR FLAG FOR XMIT
1827 004760 052737 000400 176624        BIS    #MR,$MXC      ;MASTER RESET
1828 004766 052737 004000 176624        BIS    #MCLK,$MXC   ;MAINT CLOCK
1829
1830 004774 032737 000010 001246        BIT    #BIT3,$DEV0  ;EXT LOOPBACK?
1831 005002 001403                BEQ    1$            ;BR IF YES
1832 005004 052737 014000 176624        BIS    #<MM'MCLK>,$MXC ;ELSE SET MAINT MODE FOR INT. LOOPBACK
1833 005012 012737 007026 176622        1$:  MOV    #<CHR8!ODDPAR!026>,$MPAR ;SETUP SYNC PARAMETER REGISTER
1834 005020 052737 000026 176620        BIS    #<RTS!DTR!SR SYN>,$SMRC ;SET REQ TO SEND, DATA TERM RDY
1835                                     ;& SEARCH SYNC
1836 005026 052737 000020 176624        BIS    #SEND,$MXC  ;ENABLE XMIT SEND
1837 005034 012737 000026 176626        MOV    #026,$MXB   ;XMIT SYNC CHAR
1838
1839 005042 004537 011442                JSR    R5,TIMER2    ;WAIT FOR DONE
1840 005046 176620                SMRC
1841 005050 000200                DONE
1842 005052 104045                ERROR 4$           ;NO DONE
1843 005054 000435                BR     5$           ;EXIT
1844
1845 005056 013737 176622 012546        MOV    SMRB,$COMHLD ;READ WORD
1846 005064 023727 012546 000026        CMP    $COMHLD,#026 ;IS IT SYNC CHAR?
1847 005072 001402                BEQ    4$           ;BR IF YES
1848 005074 104006                ERROR 6$           ;DID NOT REC SYNC CHAR
1849 005076 000424                BR     5$           ;EXIT
1850
1851 005100 004537 011134                4$:  JSR    R5,$SETVEC  ;SETUP VECTOR AREA
1852 005104 000340                SMRVEC
1853 005106 012640                SMRSRV
1854 005110 012550                SMXSRV
    
```



```

1855
1856 005112 012737 000027 012544      MOV    #027,CCMCHR      ;1'ST CHAR TO BE XMITTED
1857 005120 052737 000100 176620      BIS    #IE,SMRC        ;SET SYNC MODEM LOOSE.
1858 005126 052737 000100 176624      BIS    #IE,SMXC
1859
1860 005134 004537 011160              JSR    R5,TIMER        ;SEE IF COMCHR GOES TO ALL 1'S (DONE)
1861 005140 012544                      COMCHR
1862 005142 177777                      177777
1863 005144 000415                      BR     6$
1864 005146 000240                      NOP
1865
1866 005150 012737 000400 176624 5$:    MOV    #MR,SMXC        ;MASTER RESET
1867 005156 032777 010000 173754      BIT    #BIT12,@SWR     ;DO TYPEOUT IF SET
1868 005164 001416                      BEQ    AMTST           ;EXIT IF NOT
1869 005166 104401 017021                      TYPE  .SCOM
1870 005172 104401 017622                      TYPE  ,DUN
1871 005176 000411                      BR     AMTST           ;EXIT
1872
1873 005200 005737 005220              6$:    TST    FREE          ;ALLOW 1 FREE ERROR FOR APT BREAK IN
1874 005204 001402                      BEQ    7$
1875 005206 104037                      ERROR  37
1876 005210 000757                      BR     5$              ;SYNC COMM NOT RESPONDING
1877
1878 005212 005237 005220              7$:    INC    FREE
1879 005216 000656                      BR     8$
1880
1881 005220 000000                      FREE:  0                ;ERROR COUNTER

```

```

1882
1883
1884      ;:*****
1885      ;: START ASYNC COMM PORT
1886      ;:*****
1887 005222 032737 004000 001246 AMTST: BIT #BIT11,$DEV  ;DROP TEST?
1888 005230 001054 BNE CL1TST ;BR IF YES
1889
1890 005232 032737 000010 001246 BIT #BIT3,$DEV ;EXT LOOPBACK?
1891 005240 001404 BEQ 1$ ;BR IF YES
1892 005242 012737 037036 177420 MOV #<AMOD!B9600!OPAR!CHAR8!MAINT> ,PARAM ;ELSE SET FOR INT LOOPBACK
1893 005250 000403 BR 3$
1894 005252 012737 037034 177420 1$: MOV #<AMOD!B9600!OPAR!CHAR8> ,PARAM ;WAKE UP AMOD IF JUST DID SMOD
1895 005260 052737 000006 176610 3$: BIS #<RTS!DTR> ,AMRC ;SET REQ TO SEND & DATA TERM RDY
1896
1897 005266 004537 011134 JSR R5,SETVEC ;SETUP VECTOR AREA
1898 005272 000330 AMRVEC
1899 005274 012466 AMRSRV
1900 005276 012450 AMXSRV
1901
1902 005300 005037 012544 CLR COMCHR ;CHAR TO BE XMITTED
1903 005304 013737 176612 012546 MOV AMRB,COMHLD ;CLR OUT ANY PREVIOUS STORED WORD
1904 005312 052737 000100 176610 BIS #IE,AMRC ;SET ASYNC MODEM LOOSE!
1905 005320 052737 000100 176614 BIS #IE,AMXC
1906
1907 005326 004537 011160 JSR R5,TIMER ;SEE IF COMCHR GOES TO 377
1908 005332 012544 COMCHR
1909 005334 000377 377
1910 005336 104040 ERROR 40 ;ASYNC COMM NOT RESPONDING
1911 005340 000410 BR CL1TST
1912
1913 005342 032777 010000 173570 BIT #BIT12,@SWR ;SKIP TYPEOUT IF NOT SET
1914 005350 001404 BEQ CL1TST
1915 005352 104401 017034 TYPE ,ACOM
1916 005356 104401 017534 TYPE ,RUN
    
```

```

1917
1918
1919
1920
1921
1922 005362 005737 010606          CL1TST: TST      CLPRES      ;OPTION PRESENT?
1923 005366 001002                   BNE      1$          ;BR IF YES
1924 005370 000137 006036          JMP      DXTST0
1925
1926 005374 032737 004000 001246 1$:  BIT      #4000,$DEV      ;DROP ASYNC?
1927 005402 001406                   BEQ      4$          ;BR IF NO
1928 005404 042737 000006 176610  BIC      #<RTS!DTR>,$AMRC ;MUST BE TOGGLED
1929 005412 052737 000006 176610  BIS      #<RTS!DTR>,$AMRC ;PRIME IT
1930
1931 005420 032737 010000 001246 4$:  BIT      #BIT12,$DEV      ;DROP TEST?
1932 005426 001051                   BNE      CL2TST      ;BR IF YES
1933
1934 005430 032737 000001 001246  BIT      #BIT0,$DEV      ;TERM #1 SET FOR EXT LOOPBACK?
1935 005436 001404                   BEQ      2$          ;BR IF YES
1936 005440 012737 005016 177420  MOV      #<CT1!B2400!CHAR8!MAINT>,$PARAM ;ELSE SET FOR INT LOOPBACK
1937 005446 000403                   BR      3$
1938 005450 012737 005014 177420 2$:  MOV      #<CT1!B2400!CHAR8>,$PARAM ;SETUP FOR EXT LOOPBACK
1939
1940 005456 004537 011134          3$:  JSR      R5,$SETVEC ;SETUP INTERR VECTORS
1941 005462 000300                   TK1VEC
1942 005464 012166                   TK1SRV
1943 005466 012150                   TP1SRV
1944
1945 005470 005037 012244          CLR      T1CHR        ;CHAR TO BE XMITTED
1946 005474 013737 176502 012246  MOV      TK1B,T1HLD   ;READ CHAR (CLEAR STALE DATA)
1947 005502 052737 000100 176500  BIS      #IE,TK1S    ;SET CLUSTER TERM #1 LOOSE!
1948 005510 052737 000100 176504  BIS      #IE,TP1S
1949
1950 005516 004537 011160          JSR      R5,$TIMER   ;SEE IF T1CHR GOES TO 377
1951 005522 012244                   T1CHR
1952 005524 000377                   377
1953 005526 104034                   ERROR      34
1954 005530 000410                   BR      CL2TST      ;CLUSTER TERM 1 NOT RESPONDING
1955
1956 005532 032777 010000 173400  BIT      #BIT12,$SWR  ;SKIP TYPEOUT IF NOT SET
1957 005540 001404                   BEQ      CL2TST
1958 005542 104401 016766          TYPE      ,CLT1
1959 005546 104401 017534          TYPE      ,RUN
  
```

```

1960
1961
1962          ::*****
1963          ::          START CLUSTER TERMINAL #2
1964          ::*****
1965 005552 032737 020000 001246 CL2TST: BIT    #BIT13,$DEVM    ;DROP TEST?
1966 005560 001051                BNE     CL3TST        ;BR IF YES
1967
1968 005562 032737 000002 001246          BIT    #BIT1,$DEVM    ;TERM #2 SET FOR EXT LOOPBACK?
1969 005570 001404                BEQ     1$          ;BR IF YES
1970 005572 012737 055016 177420          MOV    #<CT2!B2400!CHAR8!MAINT> ,PARAM ;ELSE SET FOR INT LOOPBACK
1971 005600 000403                BR     2$
1972 005602 012737 055014 177420 1$:    MOV    #<CT2!B2400!CHAR8> ,PARAM    ;SETUP FOR EXT LOOPBACK
1973
1974 005610 004537 011134                2$:    JSR    R5,SETVEC    ;SETUP INTERR VECTORS
1975 005614 000310                TK2VEC
1976 005616 012266                TK2SRV
1977 005620 012250                TP2SRV
1978
1979 005622 005037 012344                CLR    T2CHR        ;CHAR TO BE XMITTED
1980 005626 013737 176512 012346          MOV    TK2B,T2HLD    ;READ CHAR (CLEAR STALE DATA)
1981 005634 052737 000100 176510          BIS    #IE,TK2S     ;SET CLUSTER TERM #2 LOOSE.
1982 005642 052737 000100 176514          BIS    #IE,TP2S
1983
1984 005650 004537 011160                JSR    R5,TIMER     ;SEE IF T2CHR GOES TO 377
1985 005654 012344                T2CHR
1986 005656 000377                377
1987 005660 104035                ERROR  35           ;CLUSTER TERM 2 NOT RESPONDING
1988 005662 000410                BR     CL3TST
1989
1990 005664 032777 010000 173246          BIT    #BIT12,@SWR  ;SKIP TYPEOUT IF NOT SET
1991 005672 001404                BEQ    CL3TST
1992 005674 104401 016777                TYPE  ,CLT2
1993 005700 104401 017534                TYPE  ,RUN
    
```

```
1994
1995
1996
1997
1998
1999 005704 032737 040000 001246 CL3TST: BIT #BIT14,$DEV ;DROP TEST?
2000 005712 001051 BNE DXTST0 ;BR IF YES
2001
2002 005714 032737 000004 001246 BIT #BIT2,$DEV ;TERM #3 SET FOR EXT LOOPBACK?
2003 005722 001404 BEQ 1$ ;BR IF YES
2004 005724 012737 065016 177420 MOV #<CT3!B2400!CHAR8!MAINT> ,PARAM ;ELSE SET FOR INT LOOPBACK
2005 005732 000403 BR 2$
2006 005734 012737 065014 177420 1$: MOV #<CT3!B2400!CHAR8> ,PARAM ;SETUP FOR EXT LOOPBACK
2007
2008 005742 004537 011134 2$: JSR R5,SETVEC ;SETUP INTERR VECTORS
2009 005746 000320 TK3VEC
2010 005750 012366 TK3SRV
2011 005752 012350 TP3SRV
2012
2013 005754 005037 012444 CLR T3CHR ;CHAR TO BE XMITTED
2014 005760 013737 176522 012446 MOV TK3B,T3HLD ;READ CHAR (CLEAR STALE DATA)
2015 005766 052737 000100 176520 BIS #IE,TK3S ;SET CLUSTER TERM #3 LOOSE!
2016 005774 052737 000100 176524 BIS #IE,TP3S
2017
2018 006002 004537 011160 JSR R5,TIMER ;SEE IF T3CHR GOES TO 377
2019 006006 012444 T3CHR
2020 006010 000377 377
2021 006012 104036 ERROR 36 ;CLUSTER TERM 3 NOT RESPONDING
2022 006014 000410 BR DXTST0
2023
2024 006016 032777 010000 173114 BIT #BIT12,@SWR ;SKIP TYPEOUT IF NOT SET
2025 006024 001404 BEQ DXTST0
2026 006026 104401 017010 TYPE ,CLT3
2027 006032 104401 017534 TYPE ,RUN
```

```

2028
2029
2030          :*****
2031          :          FILL & EMPTY BUFFER TEST
2032          :*****
2032          :A FILL BUFFER COMMAND IS ISSUED WITH ALTERNATING WORDS OF ALL 0'S & ALL 1'S.
2033          :AN EMPTY BUFFER COMMAND IS ISSUED TO FILL 'DOBUF'.
2034          :UPON COMPLETION, DOBUF IS CHECKED FOR ALTERNATING WORDS OF ALL 0'S & ALL 1'S.
2035          :ERRORS IN THIS TEST MAY INDICATE CABLE CROSS-TALK INTERFERENCE.
2036          :*****
2037
2038 006036 005037 003120          DXTST0: CLR          DXTST          :0 = DXTST0
2039
2040 006042 032737 000400 001246          BIT          #BIT8,$DEV          :DROP DX0 TESTS?
2041 006050 001404          BEQ          7$          :BR IF NO
2042 006052 032737 001000 001246          BIT          #BIT9,$DEV          :DROP DX1 TESTS?
2043 006060 001107          BNE          DXTST1          :BR IF YES
2044
2045 006062 004537 011442          7$: JSR          R5,TIMER2          :WAIT FOR DISK DONE
2046 006066 177170          RXCS
2047 006070 000200          DONE
2048 006072 104075          ERROR          7$          :NO DONE
2049 006074 000474          BR          8$          :EXIT
2050
2051 006076 012737 000011 177170          MOV          #INITAL,RXCS          :DO AN INITIALIZE COMMAND
2052
2053 006104 004537 011442          JSR          R5,TIMER2          :WAIT FOR DONE
2054 006110 177170          RXCS
2055 006112 000200          DONE
2056 006114 104075          ERROR          7$          :NO DONE
2057 006116 000463          BR          8$
2058
2059 006120 012700 000100          MOV          #100,R0          :FILL BUFFER
2060 006124 005001          CLR          R1          :WORD CT
2061 006126 012737 000001 177170          MOV          #FBUF,RXCS          :ISSUE FILL BUFF COMMAND
2062 006134 010137 177172          1$: MOV          R1,RXDB
2063 006140 005101          COM          R1          :COMPLEMENT WORD
2064 006142 005300          DEC          R0          :DONE ALL 100 WORDS?
2065 006144 001373          BNE          1$          :BR IF NO
2066
2067 006146 004537 011442          JSR          R5,TIMER2          :WAIT FOR DISK DONE
2068 006152 177170          RXCS
2069 006154 000200          DONE
2070 006156 104075          ERROR          7$          :NO DONE
2071 006160 000442          BR          8$          :EXIT
2072
2073
2074 006162 012700 000100          MOV          #100,R0          :EMPTY BUFFER
2075 006166 012701 002424          MOV          #DOBUF,R1          :WORD CT
2076 006172 012737 000003 177170          MOV          #EBUF,RXCS          :BUFFER ADDR
2077 006200 013721 177172          2$: MOV          RXDB,(R1)+          :ISSUE EMPTY BUFF COMMAND
2078 006204 005300          DEC          R0          :STORE WORD
2079 006206 001374          BNE          2$          :DONE?
                                :BR IF NO
    
```

```

2080
2081 006210 004537 011442      JSR      R5,TIMER2      ;WAIT FOR DISK DONE
2082 006214 177170              RXCS
2083 006215 000200              DONE
2084 006220 104075      ERROR 75      ;NO DONE
2085 006222 000421      BR      8$      ;EXIT
2086
2087                                ;CHECK IT
2088 006224 012701 002424      MOV      #DOBUF,R1      ;BUFFER ADDR
2089 006230 012137 006272      3$:     MOV      (R1)+,EBHLD ;STORE IT
2090 006234 001401              BEQ      4$      ;BR IF ZERO
2091 006236 104076      ERROR 76      ;WORD NOT ALL ZEROS
2092
2093 006240 012137 006272      4$:     MOV      (R1)+,EBHLD
2094 006244 023727 006272 177777  CMP      EBHLD,#-1      ;ALL 1'S?
2095 006252 00140              BEQ      5$      ;BR IF YES
2096 006254 104077      ERROR 77      ;WORD NOT ALL ONES
2097
2098 006256 020127 002624      5$:     CMP      R1,#DOBUF+200 ;END OF DOBUFF?
2099 006262 001362              BNE      3$      ;BR IF NO
2100 006264 000405      BR      DXTST1      ;GO TO NXT TST
2101
2102 006266 000137 010114      8$:     JMP      EOP      ;EXIT ALL DISK TESTS
2103
2104 006272 000000      EBHLD: 0      ;STORAGE      FOR ERROR REPORT
2105 006274 000000      EXPO:  0      ;EXPECT 0'S    FOR ERROR REPORT
2106 006276 177777      EXP1: -1      ;EXPECT 1'S    FOR ERROR REPORT
  
```

2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151

```

*****
START DX0 & DX1 DATA PATTERN
*****
EACH SECTOR IS FILLED WITH ITS TRACK & SECTOR ADDRESS AS DATA.
DX1 WILL ALWAYS HAVE ITS DATA BIT 15 SET TO DISTINGUISH IT FROM DX0.
EACH TRACK IS READ & CHECKED BEFORE GOING TO THE NEXT TRACK.
DATA IS WRITTEN ON ALTERNATE SECTORS TO AVOID LATENCY TIMES.
IE: SECTORS 1,3,5...31 FOLLOWED BY 2,4,6...32.
HARD ERRORS ARE THOSE ERRORS AFTER 10 MORE RETRIES.
1'ST PASS PERFORMED ON 1'ST 10 TRACKS. SUBSEQUENT PASSES ON ALL TRACKS
*****
DXTST1: MOV #1,DXTST ;1 = DATA PATT TESTS
CLR DOPAT ;INIT DX0 FLAGS
CLR DOERR
CLR DOCERR
CLR DOTRK
MOV #1,DOSEC
CLR D1PAT ;INIT DX1 FLAGS
CLR D1ERR
CLR D1CERR
CLR D1TRK
MOV #1,D1SEC
BIT #BIT8,$DEVM ;DROP DX0 TESTS?
BNE 2$ ;BR IF YES
MOV #RXSRV,RXVEC ;SETUP VECTOR AREA
MOV #200,RXVEC+2
1$: TST COMP2 ;DOING COMPAT PASS 2?
BEQ 10$ ;BR IF NO
MOV #DORSEC,DODISP ;ELSE DO READS ONLY
BR 7$
10$: MOV #DOFBUF,DODISP ;DO NORMAL TESTING
7$: CLR FIN ;DO DX1 WHEN SET
BIS #IE,RXCS ;LET INTERRUPTS LOOSE.
JSR PC,TIMDO
ERROR 41 ;DX0 NOT GETTING TRACK DONE FLAG
NOP

```



```

2152
2153 006454 032737 001000 001246 2$: BIT #BIT9,$DEVM ;DROP DX1 TESTS?
2154 006462 001031 BNE 4$ ;BR IF YES
2155
2156 006464 012737 012740 000264 MOV #RXSRV,RXVEC ;SETJP VECTOR AREA
2157 006472 012737 000200 000266 MOV #200,RXVEC+2
2158
2159 006500 005737 003134 3$: TST COMP2 ;DOING COMPAT PASS 2?
2160 006504 001404 BEQ 11$ ;BR IF NO
2161 006506 012737 015216 013046 MOV #D1RSEC,D1DISP ;ELSE DO READ ONLY
2162 006514 000403 BR 8$
2163
2164 006516 012737 014642 013046 11$: MOV #D1FBUF,D1DISP ;DO NORMAL TESTING
2165 006524 005037 003122 8$: CLR FIN ;GO BACK TO DX0 WHEN SET
2166 006530 052737 000120 177170 BIS #<IE!DX1>,RXCS ;LET INTERRUPTS LOOSE!
2167
2168 006536 004737 011340 JSR PC,TIMD1
2169 006542 104042 ERROR 42 ;DX1 NOT GETTING TRACK DONE FLAG
2170 006544 000240 NOP
2171
2172 006546 032737 000400 001246 4$: BIT #BIT8,$DEVM ;DROP DX0?
2173 006554 001007 BNE 5$ ;BR IF YES
2174 006556 005737 002400 TST DOPAT ;ELSE IS DX0 ALL DONE?
2175 006562 001004 BNE 5$ ;BR IF YES
2176 006564 005737 011336 TST DOTMO ;TIMEOUT?
2177 006570 001706 BEQ 1$ ;BR IF NO
2178 006572 000717 BR 7$ ;ELSE CONT LAST COMMAND
2179
2180 006574 032737 001000 001246 5$: BIT #BIT9,$DEVM ;DROP DX1?
2181 006602 001007 BNE 6$ ;BR IF YES
2182 006604 005737 002650 TST D1PAT ;ELSE IS DX1 ALL DONE?
2183 006610 001004 BNE 6$ ;BR IF YES
2184 006612 005737 011426 TST D1TMO ;TIMEOUT?
2185 006616 001730 BEQ 3$ ;BR IF NO
2186 006620 000741 BR 8$ ;ELSE CONT LAST COMMAND
2187
2188 006622 005737 003132 6$: TST COMP1 ;DOING COMPAT PASS 1?
2189 006626 001402 BEQ DXIST2 ;BR IF NO
2190 006630 000137 010114 JMP EOP ;ELSE ALL DONE
2191
  
```

```

2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203 006634 012737 000002 003120 DXTST2: MOV #2,DXTST ;2 = RANDOM SEEK TESTS
2204
2205 006642 005037 002404 CLR DOERR ;INIT DX0 FLAGS
2206 006646 005037 002414 CLR DOCERR
2207 006652 005037 002402 CLR DOCNT
2208
2209 006656 005037 002654 CLR D1ERR ;INIT DX1 FLAGS
2210 006662 005037 002664 CLR D1CERR
2211 006666 005037 002652 CLR D1CNT
2212
2213 006672 032737 000400 001246 BIT #BIT8,$DEV0 ;DROP DX0 TESTS?
2214 006700 001041 BNE 2$ ;BR IF YES
2215
2216 006702 005037 010734 1$: CLR LOLIM
2217 006706 013737 003124 010736 MOV MAXTRK,HILIM
2218 006714 004737 010632 JSR PC,RAND
2219 006720 010137 002406 MOV R1,DOTRK ;GET RANDOM TRACK #
2220
2221 006724 005237 010734 INC LOLIM
2222 006730 012737 000032 010736 MOV #32,HILIM
2223 006736 004737 010632 JSR PC,RAND
2224 006742 010137 002410 MOV R1,DOSEC ;GET RANDOM SECTOR #
2225
2226 006746 004537 011616 JSR R5,CKOBAD ;SEE IF THIS TRK/SEC FLAGGED BAD
2227 006752 000753 BR 1$ ;BR IF RETURN HERE
2228
2229 006754 012737 013432 013040 MOV #DORSEC,DODISP ;SET DISPATCH TABLE TO POINT TO
2230 ;READ SECTOR HANDLER.
2231 006762 005037 003122 CLR FIN ;DO DX1 WHEN SET
2232 006766 052737 000100 177170 BIS #IE,RXCS ;LET INTERRUPTS LOOSE!
2233
2234 006774 004737 011244 JSR PC,TIMDO
2235 007000 104041 ERROR 41 ;DX0 NOT GETTING DONE FLAG
2236 007002 000240 NOP

```

```

2237
2238 007004 032737 001000 001246 2$: BIT #BIT9,$DEVM ;DROP DX1 TESTS?
2239 007012 001041 BNE 4$ ;BR IF YES
2240
2241 007014 005037 010734 3$: CLR LOLIM
2242 007020 013737 003124 010736 MOV MAXTRK,HILIM
2243 007026 004737 010632 JSR PC,RAND
2244 007032 010137 002656 MOV R1,D1TRK ;GET RANDOM TRACK #
2245
2246 007036 005237 010734 INC LOLIM
2247 007042 012737 000032 010736 MOV #32,HILIM
2248 007050 004737 010632 JSR PC,RAND
2249 007054 010137 002660 MOV R1,D1SEC ;GET RANDOM SECTOR #
2250
2251 007060 004537 011730 JSR R5,CK1BAD ;SEE IF THIS TRK/SEC FLAGGED BAD
2252 007064 000753 BR 3$ ;BR IF RETURN HERE
2253
2254 007066 012737 015216 013046 MOV #D1RSEC,D1DISP ;SET DISPATCH TABLE TO POINT TO
2255 ;READ SECTOR HANDLER.
2256 007074 005037 003122 CLR FIN ;GO BACK TO DX0 WHEN SET
2257 007100 052737 000120 177170 BIS #<IE!DX1>,RXCS ;LET INTERRUPTS LOOSE!
2258
2259 007106 004737 011340 JSR PC,TIMD1
2260 007112 104042 ERROR 42 ;DX1 NOT GETTING DONE FLAG
2261 007114 000240 NOP
2262
2263 007116 032737 000400 001246 4$: BIT #BIT8,$DEVM ;DROP DX0?
2264 007124 001004 BNE 5$ ;BR IF YES
2265 007126 023737 002402 003126 CMP DOCNT,MAXSK ;DID ALL SEEKS?
2266 007134 001262 BNE 1$ ;BR IF NO
2267
2268 007136 032737 001000 001246 5$: BIT #BIT9,$DEVM ;DROP DX1?
2269 007144 001004 BNE 6$ ;BR IF YES
2270 007146 023737 002652 003126 CMP D1CNT,MAXSK ;DID ALL SEEKS?
2271 007154 001317 BNE 3$ ;BR IF NO
2272
2273 007156 005737 003134 6$: TST COMP2 ;DOING COMPAT PASS 2?
2274 007162 001402 BEQ DXTST3 ;BR IF NO
2275 007164 000137 010114 JMP EOP ;ELSE ALL DONE
2276

```

2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323

007170 032737 000400 001246
007176 001062
007200 012737 000003 003120
007206 005037 002404
007212 005037 002414
007216 012737 014476 013040
007224 005037 003122
007230 052737 000100 177170
007236 004737 011244
007242 104046
007244 000437
007246 005737 177170
007252 100001
007254 104063
007256 032737 000001 177172
007264 001001
007266 104055
007270 005037 002404
007274 005037 002414
007300 012737 000001 002406
007306 012737 000001 002410
007314 012737 013712 013040
007322 005037 003122
007326 052737 000100 177170
007334 004737 011244
007340 104056
007342 000240

```

:*****
:      INITIALIZE COMMAND TEST
:
:THE INITIALIZE & EMPTY BUFFER COMMAND IS ISSUED TO DX0.
:DATA IS VERIFIED TO BE THAT OF TRACK 1, SECTOR 1.
:HARD ERRORS ARE THOSE ERRORS AFTER 10 MORE RETRIES.
: 'INIT' CANNOT BE ISSUED TO DX1.
:*****
DXTST3: BIT      #BIT8,$DEVM      ;DROP DX0 TESTS?
        BNE      DXTST4          ;BR IF YES
        MOV      #3,DXTST        ;3 = INIT TEST
        CLR      DOERR           ;FLAGS
        CLR      DOCERR
        MOV      #DOINIT,DODISP  ;SET DISPATCH TABLE TO GO TO INIT HANDLER
        CLR      FIN
        BIS      #IE,RXCS        ;LET LOOSE
        JSR      PC,TIMD0
        ERROR    46              ;DX0 HUNG ON INITIALIZE COMMAND
        BR       DXTST4
        TST      RXCS            ;ERROR?
        BPL      1$              ;BR IF NO
        ERROR    63              ;INIT COMMAND ERROR
        BIT      #ID,RXES        ;INIT DONE BIT SET?
        BNE      2$              ;BR IF YES
        ERROR    55              ;INIT DONE NOT SET
        CLR      DOERR
        CLR      DOCERR
        MOV      #1,DOTRK        ;EXP TRK & SEC
        MOV      #1,DOSEC
        MOV      #DOEBUF,DODISP ;SETUP DISPATCH TABLE TO
                                ;EMPTY BUFFER & VERIFY DATA
        CLR      FIN
        BIS      #IE,RXCS        ;LET LOOSE
        JSR      PC,TIMD0
        ERROR    56              ;DX0 HUNG ON EMPTY BUFF COMMAND
        NOP

```

```

2324
2325
2326
2327
2328
2329
2330
2331
2332 007344 032737 000400 001246 DXTST4: BIT #BIT8,$DEV  ;DROP DX0 TESTS?
2333 007352 001121 BNE DXTST5 ;BR IF YES
2334
2335 007354 012737 000004 003120 MOV #4,DXTST ;4 RESTORE TEST
2336 007362 012737 014522 013040 MOV #D0REST,DODISP ;SETUP DISPATCH TABLE
2337 007370 005037 003122 CLR FIN
2338 007374 052737 000100 177170 BIS #IE,RXCS
2339
2340 007402 004737 011244 JSR PC,TIMDO
2341 007406 104047 ERROR 47 ;DX0 HUNG ON RESTORE COMMAND
2342 007410 000431 BR 2$
2343
2344 007412 005737 177170 TST RXCS ;ERROR?
2345 007416 100001 BPL 1$ ;BR IF NO
2346 007420 104054 ERROR 54 ;DX0 RESTORE COMMAND ERROR
2347
2348 007422 005037 002406 1$: CLR D0TRK ;CHK FOR REGRD NOT FOUND (RNF)
2349 007426 012737 000001 002410 MOV #1,D0SEC ;SET TRK/SEC. SHOULD NOT MOVE
2350 ;LOGIC THINKS ITS ALREADY THERE.
2351 007434 005037 002404 CLR D0EKR
2352 007440 005037 002414 CLR D0CERR
2353 007444 005037 003122 CLR FIN
2354 007450 012737 013432 013040 MOV #DORSEC,DODISP ;SETUP TO READ SECTOR
2355 007456 052737 000100 177170 BIS #IE,RXCS ;LET INTERR LOOSE
2356
2357 007464 004737 011244 JSR PC,TIMDO
2358 007470 104041 ERROR 41 ;DX0 HUNG ON READ CMD
2359 007472 000240 NOP
2360
2361 007474 032737 001000 001246 2$: BIT #BIT9,$DEV ;DROP DX1 TESTS?
2362 007502 001045 BNE DXTST5 ;BR IF YES
2363
2364 007504 012737 014540 013046 MOV #D1REST,D1DISP ;REPEAT FOR DX1
2365 007512 005037 003122 CLR FIN
2366 007516 052737 000120 177170 BIS #<IE!DX1>,RXCS ;LET INTERR LOOSE
2367
2368 007524 004737 011340 JSR PC,TIMD1
2369 007530 104042 ERROR 42 ;DX1 HUNG ON RESTORE CMD
2370 007532 000431 BR DXTST5
2371
2372 007534 005737 177170 TST RXCS ;ERROR?
2373 007540 100001 BPL 3$ ;BR IF NO
2374 007542 104074 ERROR 74 ;DX1 RESTORE CMD ERROR

```

```

2375
2376 007544 005037 002656 3$: CLR D1TRK ;READ TO CHECK FOR RNF
2377 007550 012737 000001 002660 MOV #1,D1SEC
2378 007556 005037 002654 CLR D1ERR
2379 007562 005037 002664 CLR D1CERR
2380 007566 005037 003122 CLR FIN
2381 007572 012737 015216 013046 MOV #D1RSEC,D1DISP ;SETUP TO READ SECTOR
2382 007600 052737 000120 177170 BIS #<IE!DX1>,RXCS ;LET INTERR LOOSE
2383
2384 007606 004737 011340 JSR PC,TIMD1
2385 007612 104042 ERROR 42 ;DX1 HUNG ON READ CMD
2386 007614 000240 NOP
2387
2388
2389
2390 ;*****
2391 ; WRITE SECTOR WITH DELETED DATA MARK TEST
2392 ;
2393 ; THIS TEST VERIFIES THAT THE WRITE DELETED DATA COMMAND CAN BE ISSUED
2394 ; & THAT THE 'DELETED DATA' BIT 5 IS SET IN RXCS AFTER READY IS RECV'D.
2395 ; HARD ERRORS ARE THOSE ERRORS AFTER 10 MORE RETRIES.
2396 ;*****
2397
2398 007616 032737 000400 001246 DXTST5: BIT #BIT8,$DEVN ;DROP DX0 TESTS?
2399 007624 001043 BNE DXTST6 ;BR IF YES
2400
2401 007626 012737 000005 003120 MOV #5,DXTST ;5 - WR DEL DATA TEST
2402 007634 005037 002404 CLR D0ERR
2403 007640 005037 002414 CLR D0CERR
2404
2405 007644 012737 013056 013040 MOV #D0FBUF,D0DISP ;SETUP DISPATCH TABLE
2406
2407 007652 005037 003122 CLR FIN
2408 007656 052737 000100 177170 BIS #IE,RXCS ;LET LOOSE
2409
2410 007664 004737 011244 JSR PC,IMD0
2411 007670 104050 ERROR 50 ;DX0 HUNG ON WRITE DEL DATA CMD
2412 007672 000420 BR DXTST6
2413
2414 007674 012737 014556 013040 MOV #D0STAT,D0DISP ;SETUP DISPATCH TABLE
2415
2416 007702 005037 003122 CLR FIN
2417 007706 052737 000100 177170 BIS #IE,RXCS ;LET LOOSE
2418
2419 007714 004737 011244 JSR PC,IMD0
2420 007720 104051 ERROR 51 ;DX0 HUNG ON READ STATUS
2421 007722 000404 BR DXTST6
2422
2423 007724 005737 177170 TST RXCS ;ERROR?
2424 007730 100001 BPL +4 ;BR IF NO
2425 007732 104057 ERROR 57 ;READ STATUS ERROR
    
```

```

2426
2427
2428
2429
2430
2431
2432
2433
2434
2435 007734 032737 000400 001246 DXTST6: BIT #BIT8,$DEVM ;DROP DX0 TESTS?
2436 007742 001062 BNE 4$ ;BR IF YES
2437
2438 007744 005037 010112 CLR INVCT
2439 007750 012737 000003 003120 MOV #6,DXTST ;6 = INV ADDR TESTS
2440 007756 012737 000115 002406 MOV #115,DOTRK
2441 007764 012737 000001 002410 MOV #1,DOSEC
2442 007772 012737 046401 177174 MOV #46401,RXSA ;INVALID TRACK
2443
2444 010000 012737 014574 013040 1$: MOV #DOINV,DODISP ;SETUP DISPATCH TABLE
2445
2446 010006 005037 003122 CLR FIN
2447 010012 052737 000100 177170 BIS #IE,RXCS ;LET LOOSE
2448
2449 010020 004737 011244 JSR PC,TIMDO
2450 010024 104060 ERROR 60 ;DX0 HUNG ON INVALID ADDR
2451 010026 000432 BR EOP
2452
2453 010030 032737 000002 177172 BIT #INVADR,RXES ;INV ADDR BIT SET?
2454 010036 001002 BNE 2$ ;BR IF YES
2455 010040 104061 ERROR 61 ;INVALID ADDR BIT NOT SET
2456 010042 000404 BR 3$
2457
2458 010044 005737 177170 2$: TST RXCS ;ERROR BIT SET?
2459 010050 100401 BMI 3$ ;BR IF YES
2460 010052 104062 ERROR 62 ;ERROR BIT NOT SET
2461
2462 010054 005737 010112 3$: TST INVCT ;DID WE JUST DO INV SECTOR?
2463 010060 001013 BNE 4$ ;BR IF YES...DONE
2464
2465 010062 005237 010112 INC INVCT
2466 010066 005037 002406 CLR DOTRK
2467 010072 012737 000033 002410 MOV #33,DOSEC
2468 010100 012737 000033 177174 MOV #33,RXSA ;INVALID SECTOR
2469 010106 000734 BR 1$ ;REPEAT FOR INVALID SECTOR
2470
2471 010110 000401 4$: BR .+4
2472
2473 010112 000000 INVCT: 0 ;0 - DOING INVALID TRACK
2474 ;1 = DOING INVALID SECTOR
2475

```

```
2476 ::*****
2477 ::      END OF PASS
2478 ::*****
2479
2480 EOP:  INC    $PASS      ;PASS CTR
2481      TYPE  ,ENDPAS
2482      MOV   $PASS,-(SP)  ;;SAVE $PASS FOR TYPEOUT
2483      TYPDS ;GO TYPE--DECIMAL ASCII WITH SIGN
2484      TYPE  ,MSG1
2485      MOV   TERR,-(SP)  ;;SAVE TERR FOR TYPEOUT
2486      TYPDS ;GO TYPE--DECIMAL ASCII WITH SIGN
2487      TYPE  ,MSG2
2488      MOV   TSERR,-(SP) ;;SAVE TSERR FOR TYPEOUT
2489      TYPDS ;GO TYPE--DECIMAL ASCII WITH SIGN
2490      TYPE  ,MSG3
2491      MOV   PERR,-(SP)  ;;SAVE PERR FOR TYPEOUT
2492      TYPDS ;GO TYPE--DECIMAL ASCII WITH SIGN
2493      TYPE  ,MSG4
2494      MOV   PSERR,-(SP) ;;SAVE PSERR FOR TYPEOUT
2495      TYPDS ;GO TYPE--DECIMAL ASCII WITH SIGN
2496      TYPE  ,NULL
2497      MOV   #114,MAXTRK ;NULL CHAR
2498      MOV   #500,MAXSK  ;FOR ALL SUBSEQUENT PASSES
2499
2500      TST   COMP1       ;COMPAT PASS 1?
2501      BEQ  1$          ;BR IF NO
2502      TYPE  ,COMPAT    ;DONE MSG
2503      HALT
2504      CLR  COMP1
2505      INC  COMP2
2506      1$: JMP  LOOP    ;REPEAT
2507
2508      000 NULL: .BYTE  -1,-1,0 ;NULL CHAR STRING
2509      010256 .EVEN
2510
```



```

2511
2512          .SBTTL PROGRAM SUBROUTINES
2513
2514
2515 010256 012737 012004 000004 SIZE:  MOV  #INTSRV,ERRVEC ;SETUP TIMEOUT TRAP ADDR
2516 010264 012737 000200 000006      MOV  #200,ERRVEC+2 ;LOCKOUT INTERR
2517 010272 005037 012012          CLR  INTFLG
2518
2519 010276 012700 017776          MOV  #17776,R0      ;SIZE MEM. START WITH 4K
2520 010302 005001          CLR  R1           ;MEM MAP, 1=4K, 2=8K, ETC
2521 010304 005037 012012          CLR  INTFLG
2522
2523 010310 005710          1$:  TST  (R0)
2524 010312 005737 012012          TST  INTFLG      ;GOT TIMEOUT INTERR?
2525 010316 001007          BNE  2$         ;BR IF YES
2526 010320 005201          INC  R1
2527 010322 020027 157776          CMP  R0,#157776  ;DONE 28K MEM?
2528 010326 001403          BEQ  2$
2529 010330 062700 020000          ADD  #20000,R0   ;GO TO NEXT 4K
2530 010334 000765          BR   1$
2531
2532 010336 010037 010630          2$:  MOV  R0,MAXMEM   ;SAVE
2533 010342 162737 020000 010630  SUB  #20000,MAXMEM ;GO BACK TO LAST VALID BLOCK
2534 010350 006301          ASL  R1           ;MULT BY 2
2535 010352 016137 010610 010362  MOV  MEMTBL-2(R1),3$
2536 010360 104401          TYPE
2537 010362 000000          3$:  .WORD 0         ;MEM SIZE
2538 010364 104401 017160          TYPE  ,MEM
2539
2540 010370 005000          CLR  R0
2541 010372 005037 012012          CLR  INTFLG
2542 010376 012700 176500          MOV  #TK1S,R0    ;SETUP CLUSTER CSR ADDR
2543 010402 005710          4$:  TST  (R0)      ;DARE IT TO TIMEOUT
2544 010404 005737 012012          TST  INTFLG     ;DID IT?
2545 010410 001011          BNE  6$         ;BR IF YES
2546 010412 020027 176520          CMP  R0,#TK3S   ;DID ALL 3?
2547 010416 001403          BEQ  5$         ;BR IF YES
2548 010420 062700 000010          ADD  #10,R0     ;ELSE DO NEXT
2549 010424 000766          BR   4$
2550
2551 010426 005237 010606          5$:  INC  CLPRES    ;OPTION PRESENT
2552 010432 000406          BR   7$         ;ALL 3 MUST RESPOND
2553
2554 010434 005037 010606          6$:  CLR  CLPRES    ;OPTION NOT PRESENT
2555 010440 104401 016746          TYPE ,CLOPT
2556 010444 104401 017200          TYPE ,NOTPRES
2557
2558 010450 012737 000006 000004 7$:  MOV  #ERRVEC+2,ERRVEC ;RESET TMO VECTOR
2559 010456 005037 000006          CLR  ERRVEC+2
    
```

```

2560
2561 010462 005000          CLR    R0
2562 010464 012701 000200  MOV    #BIT7,R1
2563
2564 010470 030137 001246  8$:    BIT    R1,$DEV0      ;DEVICE DROPPED?
2565 010474 001006          BNE    10$              ;BR IF YES
2566 010476 000241          9$:    CLC
2567 010500 006301          ASL    R1
2568 010502 103424          BCS    13$              ;BR IF ALL BITS TESTED
2569 010504 062700 000002  ADD    #2,R0            ;ELSE BUMP INDEX
2570 010510 000767          BR     8$
2571
2572 010512 016037 010532 010522 10$:   MOV    12$(R0),11$
2573 010520 104401          TYPE
2574 010522 000000          11$:   .WORD 0              ;DEVICE TO BE DROPPED
2575 010524 104401 017513  TYPE    ,DROP
2576 010530 000762          BR     9$
2577
2578 010532 017110          12$:   CLOCK
2579 010534 017076          DRVO
2580 010536 017103          DRV1
2581 010540 017021          SCOM
2582 010542 017034          ACOM
2583 010544 016766          CLT1
2584 010546 016777          CLT2
2585 010550 017010          CLT3
2586 010552 017050          PRINT
2587
2588 010554 005737 177514  13$:   TST    PRS              ;SEE IF PRINTER ERROR
2589 010560 100007          BPL    14$              ;BR IF NO
2590 010562 005037 010610  CLR    PRPRES           ;CLR FLAG
2591 010566 104401 017050  TYPE    ,PRINT          ;PRINTER NOT PRESENT
2592 010572 104401 017200  TYPE    ,NOTPRES
2593 010576 000402          BR     15$
2594 010600 005237 010610  14$:   INC    PRPRES
2595
2596 010604 000207          15$:   RTS    PC
2597
2598
2599 010606 000000          CLPRES: 0              ;CLUSTER PRESENT = NON-ZERO
2600 010610 000000          PRPRES: 0              ;PRINTER PRESENT = NON-ZERO
2601
2602 010612 017117          MEMTBL: M4K              ;MSG ADDRESSES
2603 010614 017123          M8K
2604 010616 017127          M12K
2605 010620 017134          M16K
2606 010622 017141          M20K
2607 010624 017146          M24K
2608 010626 017153          M30K
2609
2610 010630 000000          MAXMEM: 0              ;MAX MEMORY
2611
2612
2613
  
```

```

2614
2615
2616
2617
2618 010632 013700 010742
2619 010636 013701 010740
2620 010642 012702 177771
2621 010646 006300
2622 010650 006101
2623 010652 005202
2624 010654 001374
2625 010656 063700 010742
2626 010662 005501
2627 010664 063701 010740
2628 010670 062700 001057
2629 010674 005501
2630 010676 062701 047401
2631 010702 010037 010742
2632 010706 010137 010740
2633
2634 010712 042701 177400
2635 010716 020137 010734
2636 010722 103743
2637 010724 020137 010736
2638 010730 101340
2639 010732 000207
2640
2641 010734 000000
2642 010736 000000
2643
2644 010740 176543
2645 010742 123456
2646
2647
2648 010744 123727 001210 000001
2649 010752 001401
2650 010754 104407
2651 010756 000207

;*****
;      RANDOM NUMBER GENERATOR
;*****
RAND:  MOV    LONUM,R0
      MOV    HINUM,R1
      MOV    #-7,R2
1$:    ASL    R0
      ROL    R1          ;ROTATE CARRY TO R1
      INC    R2          ;DONE?
      BNE    1$         ;BR IN NO
      ADD    LONUM,R0    ;ADD NUMBER TO MAKE X 129
      ADC    R1
      ADD    HINUM,R1    ;ADD NUMBER TO MAKE X 129
      ADD    #1057,R0    ;ADD LO CONSTANT
      ADC    R1
      ADD    #47401,R1   ;ADD HI CONSTANT
      MOV    R0,LONUM
      MOV    R1,HINUM

      ;SCALE TO BE WITHIN LIMITS
      BIC    #177400,R1 ;SAVE LO BYTE ONLY
      CMP    R1,LOLIM
      BLO    RAND       ;BR IF N < LOLIM
      CMP    R1,HILIM
      BHI    RAND       ;BR IF N > HILIM
      RTS    PC         ;ELSE EXIT WITH R1 = N

LOLIM: 0
HILIM: 0

HINUM: .WORD 176543
LONUM: .WORD 123456

EXAMKB: CMPB   $ENV,#1 ;APT?
      BEQ    1$      ;BR IF YES
1$:    CKSWR
      RTS    PC

```

```

2652
2653
2654
2655
2656
2657 010760 104401 020311
2658 010764 013746 001246
2659 010770 104402
2660 010772 104401 025222
2661
2662 010776 005046
2663 011000 005046
2664
2665 011002 105777 170136
2666 011006 100375
2667 011010 117746 170132
2668 011014 042716 177600
2669
2670 011020 021627 000015
2671 011024 001013
2672 011026 005766 000004
2673 011032 001403
2674 011034 016637 000002 001246
2675 011042 062706 000006
2676 011046 104401 001165
2677 011052 000002
2678
2679 011054 004737 024322
2680 011060 021627 000060
2681 011064 002420
2682 011066 021627 000067
2683 011072 003015
2684 011074 042726 000060
2685 011100 005766 000002
2686 011104 001403
2687 011106 006316
2688 011110 006316
2689 011112 006316
2690
2691 011114 005266 000002
2692 011120 056616 177776
2693 011124 000726
2694
2695 011126 104401 001164
2696 011132 000747
2697

```

```

:*****
:ROUTINE TO DISPLAY CURRENT DEVM & ALLOW OPERATOR TO INPUT NEW VALUE.
:*****
GT$DEV: TYPE      ,DEVM      ;ELSE SHOW CURRENT
        MOV      $DEVM,-(SP) ;SAVE $DEVM FOR TYPEOUT
        TYPOC    ;GO TYPE--OCTAL ASCII(ALL DIGITS)
        TYPE     ,SMNEW      ;GET NEW
1$:     CLR      -(SP)       ;CLR CTR
        CLR      -(SP)       ;CLR NEW DEVM
2$:     TSTB    @STKS        ;CHAR THERE?
        BPL     2$          ;BR IF NO
        MOVB    @STKB,-(SP) ;ELSE GET CHAR
        BIC     #'C177,(SP) ;MAKE IT 7 BIT ASCII
3$:     CMP     (SP),#15     ;<CR>?
        BNE     7$          ;BR IF NO
        TST     4(SP)       ;ELSE IS IT 1'ST CHAR?
        BEQ     4$          ;BR IF YES
        MOV     2(SP), $DEVM ;ELSE SAVE NEW DEVM
4$:     ADD     #6,SP        ;RESTORE STACK
5$:     TYPE    , $CRLF
6$:     RTI
7$:     JSR     PC,$TYPEC    ;ECHO CHAR
        CMP     (SP),#60     ;CHAR < 0?
        BLT     9$          ;BR IF YES
        CMP     (SP),#67     ;CHAR >7?
        BGT     9$          ;BR IF YES
        BIC     #60,(SP)+    ;STRIP OFF ASCII
        TST     2(SP)       ;IS IT 1'ST CHAR?
        BEQ     8$          ;BR IF YES
        ASL     (SP)        ;ELSE SHIFT PRESENT CHAR
        ASL     (SP)        ;TO MAKE ROOM
        ASL     (SP)        ;FOR NEW ONE
8$:     INC     2(SP)        ;KEEP CHAR CT
        BIS     -2(SP),(SP) ;SET IN NEW CHAR
        BR     2$          ;GET NEW ONE
9$:     TYPE    , $QUES      ;TYPE ?<CRLF>
        BR     6$          ;DO ALL OVER

```

```

2698
2699
2700
2701
2702
2703
2704
2705
2706
2707 011134 010046
2708 011136 012500
2709 011140 012520
2710 011142 012720 000200
2711 011146 012520
2712 011150 012710 000200
2713 011154 012600
2714 011156 000205
2715
2716
2717
2718
2719
2720
2721
2722
2723
2724
2725
2726 011160 012537 011240
2727 011164 012537 011242
2728
2729 011170 012701 000010
2730 011174 012700 177777
2731 011200 027737 000034 011242
2732 011206 001411
2733 011210 004737 010744
2734 011214 005300
2735 011216 001370
2736 011220 005237 001200
2737 011224 005301
2738 011226 001362
2739 011230 000402
2740
2741 011232 062705 000004
2742 011236 000205
2743
2744 011240 000000
2745 011242 000000
2746

:*****
:* ROUTINE TO SET 2 CONSECUTIVE INTERRUPT VECTORS
:* 3 ARGUMENTS ARE PASSED THRU R5:
:*   VECTOR ADDRESS
:*   INTERRUPT SERVICE ROUTINE ADDRESS FOR RECVR
:*   INTERRUPT SERVICE ROUTINE ADDRESS FOR XMITTER.
:* PRIORITY WILL BE SET TO DISABLE FURTHUR INTERR.
:*****
SETVEC: MOV     R0, -(SP)           ;SAVE
        MOV     (R5)+, R0         ;GET VECTOR ADDR
        MOV     (R5)+, (R0)+      ;PUT SERV ROUTINE ADDR THERE
        MOV     #200, (R0)+      ;DISABLE INTERR
        MOV     (R5)+, (R0)+      ;GET XMIT SERV ADDR NEXT
        MOV     #200, (R0)
        MOV     (SP)+, R0         ;RESTORE
        RTS     R5

:*****
:WATCHDOG TIMER TO PREVENT DEVICES FROM ENTERING AN ENDLESS WAIT LOOP
:RETURN IF TIMED OUT
:RETURN +4 IF NOT TIMED OUT & JUMP OVER ERROR
:*****
TIMER:  MOV     (R5)+, FLGHLD      ;GET FLAG
        MOV     (R5)+, CTHLD      ;GET COUNT THAT FLAG SHOULD GO TO
        MOV     #10, R1
        MOV     #-1, R0
        CMP     @FLGHLD, CTHLD    ;RESPONDING?
        BEQ     2$,              ;BR IF YES
        JSR     PC, EXAMKB
        DEC     R0                ;ELSE DEC COUNTER
        BNE     1$,              ;BR IF NOT TIMED OUT
        INC     $DEVCT           ;FOR APT
        DEC     R1
        BNE     4$,
        BR     .+6                ;ELSE TIMED OUT
        ADD     #4, R5            ;JUMP OVER ERROR ON RETURN
        RTS     R5

FLGHLD: 0 ;FLAG
CTHLD:  0 ;COUNT THAT FLAG MUST REACH
    
```

```

2747 ;*****
2748 ;WATCHDOG TIMER TO PREVENT DX0 FROM ENTERING AN ENDLESS WAIT LOOP
2749 ;RETURN IF TIMED OUT
2750 ;RETURN +4 IF NOT TIMED OUT & JUMP OVER ERROR
2751 ;*****
2752
2753 011244 005037 011336 TIMDO: CLR DOTMO
2754 011250 012737 000010 011334 MOV #10,CTREG1
2755 011256 012737 177777 011332 4$: MOV #-1,CTREG
2756 011264 005737 003122 1$: TST FIN ;FINISHED?
2757 011270 001015 BNE 2$ ;BR IF YES
2758 011272 004737 010744 JSR PC,EXAMKB
2759 011276 005337 011332 DEC CTREG ;ELSE DEC CTR
2760 011302 001370 BNE 1$ ;BR IF NOT TIMED OUT
2761 011304 005237 001200 INC $DEVCT ;FOR APT
2762 011310 005337 011334 DEC CTREG1
2763 011314 001360 BNE 4$
2764 011316 005237 011336 INC DOTMO ;SET FLAG
2765 011322 000402 BR .+6 ;TIMED OUT
2766
2767 011324 062716 000004 2$: ADD #4,(SP) ;BUMP RET
2768 011330 000207 RTS PC
2769
2770 011332 000000 CTREG: 0
2771 011334 000000 CTREG1: 0
2772 011336 000000 DOTMO: 0 ;1 = TIMEOUT OCCURED
2773
2774
2775 ;*****
2776 ;WATCHDOG TIMER TO PREVENT DX1 FROM ENTERING AN ENDLESS WAIT LOOP
2777 ;RETURN IF TIMED OUT
2778 ;RETURN +4 IF NOT TIMED OUT & JUMP OVER ERROR
2779 ;*****
2780
2781 011340 005037 011426 TIMD1: CLR D1TMO
2782 011344 012737 000010 011334 MOV #10,CTREG1
2783 011352 012737 177777 011332 4$: MOV #-1,CTREG
2784 011360 005737 003122 1$: TST FIN ;FINISHED?
2785 011364 001015 BNE 2$ ;BR IF YES
2786 011366 004737 010744 JSR PC,EXAMKB
2787 011372 005337 011332 DEC CTREG ;ELSE DEC CTR
2788 011376 001370 BNE 1$ ;BR IF NOT TIMED OUT
2789 011400 005237 001200 INC $DEVCT ;FOR APT
2790 011404 005337 011334 DEC CTREG1
2791 011410 001360 BNE 4$
2792 011412 005237 011426 INC D1TMO ;SET FLAG
2793 011416 000402 BR .+6 ;TIMED OUT
2794
2795 011420 062716 0000C4 2$: ADD #4,(SP) ;BUMP RET
2796 011424 000207 RTS PC
2797
2798 011426 000000 D1TMO: 0 ;1 = TIMEOUT OCCURED
2799

```

```

2800      ;:*****
2801      ;:      GENERAL TIMER
2802      ;:*****
2803
2804 011430 012700 177777  TIMER1: MOV    #-1,R0
2805 011434 005300          DEC    R0
2806 011436 001376          BNE   .-2
2807 011440 000207          RTS    PC
2808
2809      ;:*****
2810      ;:WATCHDOG TIMER TO PREVENT DEVICES FROM ENTERING AN ENDLESS WAIT LOOP
2811      ;:RETURN    IF TIMED OUT
2812      ;:RETURN +4 IF NOT TIMED OUT & JUMP OVER ERROR
2813      ;:*****
2814
2815 011442 012537 011522  TIMER2: MOV    (R5)+,REGHLD ;GET REG
2816 011446 012537 011524      MOV    (R5)+,BITHLD ;GET BIT TO BE TESTED
2817
2818 011452 012701 000010      MOV    #10,R1
2819 011456 012700 177777  4$:    MOV    #-1,R0
2820 011462 033777 011524 000032 1$:    BIT    BITHLD,@REGHLD ;BIT THERE?
2821 011470 001011          BNE   2$ ;BR IF YES
2822 011472 004737 010744      JSR   PC,EXAMKB
2823 011476 005300          DEC    R0 ;ELSE DEC COUNTER
2824 011500 001370          BNE   1$ ;BR IF NOT TIMED OUT
2825 011502 005237 001200      INC    $DEVCT ;FOR APT
2826 011506 005301          DEC    R1
2827 011510 001362          BNE   4$
2828 011512 000402          BR    .+6 ;ELSE TIMED OUT
2829
2830 011514 062705 000004  2$:    ADD    #4,R5 ;JUMP OVER ERROR ON RETURN
2831 011520 000205          RTS    R5
2832
2833 011522 000000      REGHLD: 0 ;DEVICE REG
2834 011524 000000      BITHLD: 0 ;DEV REG BIT TO BE TESTED
2835
2836
2837      ;:*****
2838      ;:ROUTINE TO CLEAR BOTH BAD TRACK/SECTOR TABLES
2839      ;:*****
2840
2841 011526 012700 002624  CLRBAD: MOV    #DOBAD,R0
2842 011532 005020  1$:    CLR    (R0)+
2843 011534 020027 002650      CMP    R0,#DOBAD+20.
2844 011540 001374          BNE   1$
2845
2846 011542 012700 003074  2$:    MOV    #D1BAD,R0
2847 011546 005020      CLR    (R0)+
2848 011550 020027 003120      CMP    R0,#D1BAD+20.
2849 011554 001374          BNE   2$
2850
2851 011556 000207          RTS    PC
2852

```

```

2853
2854
2855
2856
2857 011560 010046
2858
2859 011562 012700 002624
2860 011566 005710
2861 011570 001003
2862 011572 013710 177174
2863 011576 000405
2864
2865 011600 005720
2866 011602 020027 002650
2867 011606 001367
2868 011610 104104
2869 011612 012600
2870 011614 000207
2871
2872
2873
2874
2875
2876
2877
2878
2879
2880
2881 011616 010046
2882 011620 010146
2883
2884 011622 013701 002406
2885 011626 000301
2886 011630 053701 002410
2887
2888 011634 012700 002624
2889 011640 020027 002650
2890 011644 001405
2891 011646 005710
2892 011650 001403
2893 011652 022001
2894 011654 001371
2895 011656 000402
2896
2897 011660 062705 000002
2898 011664 012601
2899 011666 012600
2900 011670 000205

```

```

;*****
;LOAD DX0 BAD TRACK/SECTOR TABLE
;*****
LDOBAD: MOV     R0,-(SP)           ;SAVE
1$:     MOV     #DOBAD,R0
        TST     (R0)             ;ENTRY PRESENT?
        BNE     2$              ;BR IF YES
        MOV     RXSA,(R0)        ;ELSE STORE BAD RXSA IN TABLE
        BR      3$              ;EXIT
2$:     TST     (R0)+            ;BUMP PTR
        CMP     R0,#DOBAD+20.    ;AT END OF TABLE?
        BNE     1$              ;BR IF NO
        ERROR   104             ;10 BAD SECTORS...REPLACE
3$:     MOV     (SP)+,R0         ;RESTORE
        RTS     PC

```

```

;*****
;CHECK DX0 BAD TRACK/SECTOR TABLE BEFORE DOING RANDOM SEEKS.
;IF TRK/SEC IN TABLE, RETURN TO RE-CALCULATE NEW TRACK & SECTOR.
;*****
CKOBAD: MOV     R0,-(SP)           ;SAVE
        MOV     R1,-(SP)
        MOV     D0TRK,R1
        SWAB   R1
        BIS     D0SEC,R1
1$:     MOV     #DOBAD,R0
        CMP     R0,#DOBAD+20.    ;END OF TABLE?
        BEQ     2$              ;BR IF YES
        TST     (R0)             ;ELSE ANY ENTRY?
        BEQ     2$              ;BR IF NO
        CMP     (R0)+,R1        ;ELSE COMPARE?
        BNE     1$              ;BR IF NO
        BR      3$              ;ELSE DONT BUMP RET ADDR
2$:     ADD     #2,R5            ;BUMP RET ADDR
3$:     MOV     (SP)+,R1         ;RESTORE
        MOV     (SP)+,R0
        RTS     R5

```



```

2901
2902
2903
2904
2905 011672 010046
2906
2907 011674 012700 003074
2908 011700 005710
2909 011702 001003
2910 011704 013710 177174
2911 011710 000405
2912
2913 011712 005720
2914 011714 020027 003120
2915 011720 001367
2916 011722 104105
2917 011724 012600
2918 011726 000207
2919
2920
2921
2922
2923
2924
2925
2926
2927
2928
2929 011730 010046
2930 011732 010146
2931
2932 011734 013701 002656
2933 011740 000301
2934 011742 053701 002660
2935
2936 011746 012700 003074
2937 011752 020027 003120
2938 011756 001405
2939 011760 005710
2940 011762 001403
2941 011764 022001
2942 011766 001371
2943 011770 000402
2944
2945 011772 062705 000002
2946 011776 012601
2947 012000 012600
2948 012002 000205
2949

;*****
;LOAD DX1 BAD TRACK/SECTOR TABLE
;*****
LD1BAD: MOV R0,-(SP) ;SAVE
1$: MOV #D1BAD,R0
TST (R0) ;ENTRY PRESENT?
BNE 2$ ;BR IF YES
MOV RXSA,(R0) ;ELSE STORE BAD RXSA IN TABLE
BR 3$ ;EXIT
2$: TST (R0)+ ;BUMP PTR
CMP R0,#D1BAD+20. ;AT END OF TABLE?
BNE 1$ ;BR IF NO
ERROR 105 ;10 BAD SECTORS...REPLACE
3$: MOV (SP)+,R0 ;RESTORE
RTS PC

;*****
;CHECK DX1 BAD TRACK/SECTOR TABLE BEFORE DOING RANDOM SEEKS.
;IF TRK/SEC IN TABLE, RETURN TO RE-CALCULATE NEW TRACK & SECTOR.
;*****
CK1BAD: MOV R0,-(SP) ;SAVE
MOV R1,-(SP)
MOV D1TRK,R1
SWAB R1
BIS D1SEC,R1
1$: MOV #D1BAD,R0
CMP R0,#D1BAD+20. ;END OF TABLE?
BEQ 2$ ;BR IF YES
TST (R0) ;ELSE ANY ENTRY?
BEQ 2$ ;BR IF NO
CMP (R0)+,R1 ;ELSE COMPARE?
BNE 1$ ;BR IF NO
BR 3$ ;ELSE DONT BUMP RET ADDR
2$: ADD #2,R5 ;BUMP RET ADDR
3$: MOV (SP)+,R1 ;RESTORE
MOV (SP)+,R0
RTS R5

```

```

2950 .SBTTL INTERRUPT HANDLERS
2951 :*****
2952 : GENERAL SERVICE ROUTINE TO BUMP 'INTFLG'
2953 : CALLING ROUTINE WILL KNOW WHAT TO DO
2954 :*****
2955
2956 012004 005237 012012 INTSRV: INC INTFLG
2957 012010 000002 RTI
2958 012012 000000 INTFLG: 0
2959
2960
2961 :*****
2962 :PRINTER INTERRUPT HANDLER: VALUES FROM 40 THRU 176.
2963 :*****
2964
2965 012014 013737 177514 012144 PRSRV: MOV PRS,SPRS ;STORE
2966 012022 005737 012144 TST SPRS ;ERROR?
2967 012026 100002 BPL 1$ ;BR IF NO
2968 012030 104007 ERROR 7 ;PRINTER STATUS ERROR
2969 012032 000002 RTI
2970
2971 012034 013737 012140 177516 1$: MOV PRCHR,PRB ;ELSE PRINT CHAR
2972 012042 023727 012140 000015 CMP PRCHR,#CR ;JUST DID CR?
2973 012050 001413 BEQ 2$ ;BR IF YES
2974 012052 023727 012140 000012 CMP PRCHR,#LF ;JUST DID LF?
2975 012060 001413 BEQ 3$ ;BR IF YES
2976 012062 023727 012140 000176 CMP PRCHR,#176 ;JUST DID LAST CHAR?
2977 012070 001413 BEQ 4$ ;BR IF YES
2978 012072 005237 012140 INC PRCHR ;ELSE BUMP CHAR FOR NEXT INTERRUPT
2979 012076 000117 BR 5$ ;EXIT
2980
2981 012100 012737 000012 012140 2$: MOV #LF,PRCHR ;DO LF NEXT
2982 012106 000413 BR 5$
2983 012110 012737 000040 012140 3$: MOV #40,PRCHR ;DO SPACE NEXT
2984 012116 000407 BR 5$
2985 012120 012737 000015 012140 4$: MOV #CR,PRCHR ;DO CR NEXT & START OVER
2986 012126 005237 012142 INC LINCT
2987 012132 005237 001200 INC $DEVCT ;FOR APT
2988 012136 000002 5$: RTI
2989
2990 012140 000700 PRCHR: 0 ;CHAR TO BE PRINTED
2991 012142 000000 LINCT: 0 ;LINE CTR
2992 012144 000000 SPRS: 0 ;STORE PRINTER CSR
2993 012146 000000 PRPORT: 0 ;SET IF DOING 5 LINES ONLY

```

```
2994
2995
2996
2997
2998
2999 012150 042737 000100 176504 TP1SRV: BIC #IE,TP1S ;DISABLE INTER, RECV'R WILL TURN BACK ON
3000 012156 013737 012244 176506 MOV T1CHR,TP1B ;PRINT CHAR
3001 012164 000002 RTI
3002
3003 012166 013737 176502 012246 TK1SRV: MOV TK1B,T1HLD ;STORE CHAR
3004 012174 023737 012246 012244 CMP T1HLD,T1CHR ;OK?
3005 012202 001401 BEQ 1$ ;BR IF YES
3006 012204 104010 ERROR 10 ;CHAR COMP ERROR
3007
3008 012206 023727 012244 000377 1$: CMP T1CHR,#377 ;LAST CHAR?
3009 012214 001403 BEQ 2$ ;BR IF YES
3010 012216 005237 012244 INC T1CHR ;ELSE BUMP
3011 012222 000404 BR 3$
3012
3013 012224 005037 012244 2$: CLR T1CHR ;START OVER
3014 012230 005237 001200 INC $DEVCT ;FOR APT
3015 012234 052737 000100 176504 3$: BIS #IE,TP1S ;ALLOW PRINTER INTERR
3016 012242 000002 RTI
3017
3018 012244 000000 T1CHR: 0 ;CHAR TO XMITT
3019 012246 000000 T1HLD: 0 ;REC'D CHAR
3020
3021
3022
3023
3024
3025
3026 012250 042737 000100 176514 TP2SRV: BIC #IE,TP2S ;DISABLE INTER, RECV'R WILL TURN BACK ON
3027 012256 013737 012344 176516 MOV T2CHR,TP2B ;PRINT CHAR
3028 012264 000002 RTI
3029
3030 012266 013737 176512 012346 TK2SRV: MOV TK2B,T2HLD ;STORE CHAR
3031 012274 023737 012346 012344 CMP T2HLD,T2CHR ;OK?
3032 012302 001401 BEQ 1$ ;BR IF YES
3033 012304 104011 ERROR 11 ;CHAR COMP ERROR
3034
3035 012306 023727 012344 000377 1$: CMP T2CHR,#377 ;LAST CHAR?
3036 012314 001403 BEQ 2$ ;BR IF YES
3037 012316 005237 012344 INC T2CHR ;ELSE BUMP
3038 012322 000404 BR 3$
3039
3040 012324 005037 012344 2$: CLR T2CHR ;START OVER
3041 012330 005237 001200 INC $DEVCT ;FOR APT
3042 012334 052737 000100 176514 3$: BIS #IE,TP2S ;ALLOW PRINTER INTERR
3043 012342 000002 RTI
3044
3045 012344 000000 T2CHR: 0 ;CHAR TO XMITT
3046 012346 000000 T2HLD: 0 ;REC'D CHAR
```

3047
3048
3049
3050
3051
3052
3053
3054
3055
3056
3057
3058
3059
3060
3061
3062
3063
3064
3065
3066
3067
3068
3069
3070
3071
3072
3073
3074
3075
3076
3077
3078
3079
3080
3081
3082
3083
3084
3085
3086
3087
3088
3089
3090
3091
3092
3093
3094
3095
3096
3097
3098
3099

012350 042737 000100 176524
012356 013737 012444 176526
012364 000002
012366 013737 176522 012446
012374 023737 012446 012444
012402 001401
012404 104012
012406 023727 012444 000377
012414 001403
012416 005237 012444
012422 000404
012424 005037 012444
012430 005237 001200
012434 052737 000100 176524
012442 000002
012444 000000
012446 000000
012450 042737 000100 176614
012456 013737 012544 176616
012464 000002
012466 013737 176612 012546
012474 023737 012546 012544
012502 001401
012504 104013
012506 023727 012544 000377
012514 001403
012516 005237 012544
012522 000404
012524 005037 012544
012530 005237 001200
012534 052737 000100 176614
012542 000002
012544 000000
012546 000000

:CLUSTER TERMINAL #3 INTERRUPT HANDLER: LOOP VALUES 0 THRU 377.

TP3SRV: BIC #IE,TP3S ;DISABLE INTER, RECV'R WILL TURN BACK ON
MOV T3CHR,TP3B ;PRINT CHAR
RTI
TK3SRV: MOV TK3B,T3HLD ;STORE CHAR
CMP T3HLD,T3CHR ;OK?
BEQ 1\$;BR IF YES
ERROR 12 ;CHAR COMP ERROR
1\$: CMP T3CHR,#377 ;LAST CHAR?
BEQ 2\$;BR IF YES
INC T3CHR ;ELSE BUMP
BR 3\$
2\$: CLR T3CHR ;START OVER
INC \$DEVCT ;FOR APT
3\$: BIS #IE,TP3S ;ALLOW PRINTER INTERR
RTI
T3CHR: 0 ;CHAR TO XMITT
T3HLD: 0 ;REC'D CHAR

:ASYN COMM PORT INTERRUPT HANDLER: LOOP VALUES 0 THRU 377.

AMXSRV: BIC #IE,AMXC ;DISABLE INTER, RECV'R WILL TURN BACK ON
MOV COMCHR,AMXB ;XMITT CHAR
RTI
AMRSRV: MOV AMRB,COMHLD ;STORE CHAR
CMP COMHLD,COMCHR ;OK?
BEQ 1\$;BR IF YES
ERROR 13 ;CHAR COMPARE ERROR
1\$: CMP COMCHR,#377 ;LAST CHAR?
BEQ 2\$;BR IF YES
INC COMCHR ;ELSE BUMP
BR 3\$
2\$: CLR COMCHR ;START OVER
INC \$DEVCT ;FOR APT
3\$: BIS #IE,AMXC ;ALLOW XMIT INTERR
RTI
COMCHR: 0 ;XMIT CHAR
COMHLD: 0 ;REC'D CHAR

```
3100
3101
3102      ;*****
3103      ;SYNC COMM PORT INTERRUPT HANDLER: LOOP VALUES 27 THRU 377.
3104      ;*****
3105 012550 042737 000100 176624 SMXSRV: BIC      #IE,SMXC      ;DISABLE INTER, RECV'R WILL TURN BACK ON
3106 012556 013737 012544 176626      MOV      COMCHR,SMXB ;XMITT CHAR
3107 012564 023727 012544 000377      CMP      COMCHR,#377 ;LAST CHAR?
3108 012572 001020      BNE      2$          ;BR IF NO
3109
3110 012574 005737 012636      TST      LSTCHR
3111 012600 001010      BNE      1$
3112 012602 005237 012636      INC      LSTCHR
3113 012606 052737 000100 176624      BIS      #IE,SMXC      ;ALLOW RDY INTR SO CAN SHUT DOWN FAST
3114 012614 005237 001200      INC      $DEVCT        ;FOR APT
3115 012620 000002      RTI
3116
3117 012622 042737 000020 176624 1$:      BIC      #SEND,SMXC    ;NO MORE TO SEND
3118 012630 005237 001200      INC      $DEVCT        ;FOR APT
3119 012634 000002      2$:      RTI
3120
3121 012636 000000      LSTCHR: 0              ;SET WHEN LAST CHAR XMIT
3122
3123
3124
3125
3126
3127 012640 013737 176622 012546 SMRSRV: MOV      SMRB,COMHLD ;STORE CHAR
3128 012646 123727 012546 000026      CMPB     COMHLD,#026  ;IGNORE SYNC CHARS
3129 012654 001430      BEQ      4$
3130 012656 023737 012546 012544      CMP      COMHLD,COMCHR ;OK?
3131 012664 001401      BEQ      1$          ;BR IF YES
3132 012666 104014      ERROR   14          ;CHAR COMPARE ERROR
3133
3134 012670 023727 012544 000377 1$:      CMP      COMCHR,#377  ;LAST CHAR?
3135 012676 001403      BEQ      2$          ;BR IF YES
3136 012700 005237 012544      INC      COMCHR        ;ELSE BUMP
3137 012704 000411      BR       3$
3138
3139 012706 012737 177777 012544 2$:      MOV      #-1,COMCHR   ;INDICATE DONE
3140 012714 042737 000100 176620      BIC      #IE,SMRC      ;ALL DONE
3141 012722 005237 001200      INC      $DEVCT        ;FOR APT
3142 012726 000403      BR       4$
3143
3144 012730 052737 000100 176624 3$:      BIS      #IE,SMXC      ;ALLOW XMIT INTERR
3145 012736 000002      4$:      RTI
3146
```

```

3147
3148
3149
3150
3151
3152
3153
3154
3155
3156
3157 012740 105737 177170      RXSRV:  TSTB   RXCS      ;CONTR RDY?
3158 012744 100413              BMI     1$          ;BR IF YES
3159 012746 013737 177170 013050  MOV     RXCS,SRXCS ;ELSE SAVE FOR ERROR TYPEOUTS
3160 012754 013737 177172 013052  MOV     RXES,SRXES
3161 012762 013737 177174 013054  MOV     RXSA,SRXSA
3162 012770 104015              ERROR   15          ;UNEXPECTED INTERRUPT
3163 012772 000002              RTI                    ;GO BACK
3164
3165 012774 005737 177170      1$:     TST     RXCS      ;ERROR?
3166 013000 100011              BPL     2$          ;BR IF NO
3167
3168 013002 013737 177170 013050  MOV     RXCS,SRXCS ;ELSE SAVE FOR ERROR TYPEOUTS
3169 013010 013737 177172 013052  MOV     RXES,SRXES
3170 013016 013737 177174 013054  MOV     RXSA,SRXSA
3171
3172 013024 032737 000020 177170 2$:     BIT     #USEL,RXCS ;CHECK UNIT SELECT BIT
3173 013032 001003              BNE                    ;BR IF INTER FROM DX1
3174
3175 013034 000177 000000              JMP     @DODISP    ;ELSE DISPATCH TO DX0 SERVICE ROUTINE
3176 013040 000000      DODISP: 0
3177
3178 013042 000177 000000      D1INT:  JMP     @D1DISP ;DX1 INTERRUPT DISPATCH
3179 013046 000000      D1DISP: 0
3180
3181
3182 013050 000000      SRXCS:  0          ;SAVE RXCS
3183 013052 000000      SRXES:  0          ;SAVE RXES
3184 013054 000000      SRXSA:  0          ;SAVE RXSA
3185

```

3186
3187
3188
3189
3190
3191
3192
3193
3194
3195
3196
3197
3198
3199
3200
3201
3202
3203
3204
3205
3206
3207
3208
3209
3210
3211
3212
3213
3214
3215
3216
3217
3218
3219
3220
3221
3222

013056 012703 000100
013062 013701 002406
013066 000301
013070 053701 002410
013074 012737 013122 013040
013102 012737 000101 177170
013110 010137 177172
013114 005303
013116 001374
013120 000002

: HANDLER TO FILL THE DRIVE BUFFER FOR DX0
: & POINT TO THE WRITE SECTOR HANDLER.

```
D0FBUF: MOV #100,R3 ;WORD CT
        MOV D0TRK,R1
        SWAB R1
        BIS D0SEC,R1
        MOV #D0WSEC,D0DISP ;POINT TO WRITE SECTOR AFTER RTI
        MOV #<FBUF!IE>,RXCS ;ISSUE FILL BUFFER CMD
1$:     MOV R1,RXDB ;FILL ENTIRE SECTOR WITH ITS ADDRESS
        DEC R3
        BNE 1$
        RTI
```

: HANDLER TO WRITE THE SECTOR FOR DX0 & POINT TO WRITE CHECK.

013122 013701 002406
013126 000301
013130 053701 002410
013134 010137 177174
013140 012737 013176 013040
013146 023727 003120 000005
013154 001004
013156 012737 000115 177170
013164 000403
013166 012737 000105 177170
013174 000002

```
D0WSEC: MOV D0TRK,R1
        SWAB R1
        BIS D0SEC,R1
        MOV R1,RXSA ;LOAD TRACK & SECTOR ADDR
        MOV #D0WCHK,D0DISP ;POINT TO WRITE CHECK HANDLER
        CMP DXTST,#5
        BNE 1$
        MOV #<WDDSEC!IE>,RXCS ;ONLY FOR WRITE DELETED DATA TEST
        BR 2$
2$:     MOV #<WSEC!IE>,RXCS ;ISSUE WRITE SECTOR COMMAND
        RTI
```

```

3223
3224
3225
3226
3227
3228
3229
3230
3231 013176 005737 177170
3232 013202 100024
3233 013204 023727 002404 000012
3234 013212 001072
3235 013214 004737 011560
3236 013220 032737 000020 013052
3237 013226 001410
3238 013230 104064
3239 013232 012737 013310 013040
3240 013240 012737 000117 177170
3241 013246 000002
3242
3243 013250 104016
3244 013252 000416
3245
3246 013254 005737 002404
3247 013260 001413
3248 013262 005237 026054
3249 013266 005237 026060
3250 013272 032737 000020 013052
3251 013300 001402
3252 013302 104065
3253 013304 000401
3254 013306 104017
3255
3256 013310 005037 002404
3257 013314 023727 002410 000031
3258 013322 001004
3259 013324 012737 000002 002410
3260 013332 000407
3261 013334 023727 002410 000032
3262 013342 001405
3263 013344 062737 000002 002410
3264 013352 000137 013056
3265
3266 013356 012737 000001 002410
3267 013364 005037 002404
3268 013370 005037 002412
3269 013374 000137 013432
3270
3271 013400 005237 002404
3272 013404 032737 000020 013052
3273 013412 001757
3274 013414 012737 013056 013040
3275 013422 012737 000117 177170
3276 013430 000002

```

```

:*****
:HANDLER TO WRITE CHECK ON DX0.
:WILL GO TO FILL BUFFER TO DO SAME SECTOR ON SOFT ERROR
:OR NEXT SECTOR ON NO ERROR OR HARD ERROR.
:WILL GO TO READ SECTOR AFTER ALL SECTORS ON TRACK ARE WRITTEN.
:A RESTORE IS PERFORMED AFTER ANY SEEK ERROR (RNF).
:*****
DOWCHK: TST      RXCS      ;ERROR?
        BPL      1$        ;BR IN NO
        CMP      DOERR,#10. ;10 ERRORS?
        BNE      6$        ;BR IF NO & TRY AGAIN
        JSR      PC,LDOBAD  ;LOAD BAD TRK/SEC TABLE
        BIT      #RNF,SRXES ;SEEK ERROR?
        BEQ      7$        ;BR IF NO
        ERROR    64        ;HARD SEEK ERROR WRITE SECTOR
        MOV      #2$,DODISP ;GO TO 2$ AFTER RESTORE
        MOV      #<RESTOR.IE>,RXCS
        RTI
7$:     ERROR    16        ;HARD ERROR WRITE SECTOR
        BR      2$        ;GO TO NEXT SECTOR
1$:     TST      DOERR      ;ANY PREV ERRORS?
        BEQ      2$        ;BR IF NO
        INC      TSERR     ;TOTAL SOFT ERR CT
        INC      PSERR     ;PASS SOFT ERR CT
        BIT      #RNF,SRXES ;SEEK ERROR?
        BEQ      8$        ;BR IF NO
        ERROR    65        ;SOFT SEEK ERROR WRITE SECTOR
        BR      2$
8$:     ERROR    17        ;SOFT ERROR WRITE SECTOR
2$:     CLR      DOERR     ;LAST ODD SECTOR?
        CMP      DOSEC,#31 ;BR IF NO
        BNE      3$        ;ELSE DO EVEN SECTORS NOW
        MOV      #2,DOSEC
        BR      4$
3$:     CMP      DOSEC,#32 ;LAST EVEN SECTOR?
        BEQ      5$        ;BR IF YES
        ADD      #2,DOSEC  ;ELSE BUMP SECTOR
        JMP      DOFBUF   ;GO TO FILL BUFFER & DO ANOTHER
5$:     MOV      #1,DOSEC  ;ALL SECTORS DONE...CHECK DATA
        CLR      DOERR
        CLR      DDCMER
        JMP      DORSEC   ;GO TO READ SECTOR HANDLER
6$:     INC      DOERR
        BIT      #RNF,SRXES ;SEEK ERROR?
        BEQ      4$        ;BR IN NO
        MOV      #DOFBUF,DODISP ;GOTO FILL BUFF AFTER RESTORE
        MOV      #<RESTOR.IE>,RXCS
        RTI

```



```

3277
3278
3279
3280
3281
3282
3283 013432 013701 002406 DORSEC: MOV DOTRK,R1
3284 013436 000301 SWAB R1
3285 013440 053701 002410 BIS DOSEC,R1
3286 013444 010137 177174 MOV R1,RXSA ;LOAD TRK & SECTOR ADDR
3287
3288 013450 012737 013466 013040 MOV #DORCHK,DODISP ;POINT TO READ CHECK HANDLER
3289 013456 012737 000107 177170 MOV #<RSEC!IE>,RXCS ;ISSUE READ SECTOR COMMAND
3290 013464 000002 RTI
3291
3292
3293
3294
3295
3296
3297
3298
3299
3300
3301 013466 005737 177170 DORCHK: TST RXCS ;ERROR?
3302 013472 100054 BPL 1$ ;BR IF NO
3303 013474 023727 002404 000012 CMP DOERR,#10. ;10 ERRORS?
3304 013502 001415 BEQ 3$ ;BR IF YES
3305 013504 005237 002404 INC DOERR ;ELSE DO AGAIN
3306 013510 032737 000020 013052 BIT #RNF,SRXES ;SEEK ERROR?
3307 013516 001745 BEQ DORSEC ;BR IF NO
3308 013520 012737 013432 013040 MOV #DORSEC,DODISP ;ELSE DO RESTORE & RET TO RD SEC
3309 013526 012737 000117 177170 MOV #<RESTOR!IE>,RXCS
3310 013534 000002 RTI
3311
3312 013536 004737 011560 3$: JSR PC,LDOBAD ;LOAD BAD TRK/SEC TABLE
3313 013542 032737 000020 013052 BIT #RNF,SRXES ;SEEK ERROR?
3314 013550 001410 BEQ 7$ ;BR IF NO
3315 013552 104066 ERROR 66 ;HARD SEEK ERROR READ SECTOR
3316
3317 013554 012737 014454 013040 MOV #DONEXT,DODISP ;DO RESTORE
3318 013562 012737 000117 177170 MOV #<RESTOR!IE>,RXCS
3319 013570 000002 RTI
3320
3321 013572 032737 000010 013052 7$: BIT #CRC,SRXES ;CRC ERROR?
3322 013600 001404 BEQ 10$ ;BR IF NO
3323 013602 005237 002422 INC DOCRC ;ELSE SET FLAG
3324 013606 104106 ERROR 106 ;HARD CRC ERROR
3325 013610 000423 BR 2$ ;& GO TO EMP BUFF TO CHK DATA
3326
3327 013612 005037 002422 10$: CLR DOCRC
3328 013616 104020 ERROR 20 ;HARD READ ERROR
3329 013620 000137 014454 JMP DONEXT
    
```

```

3330
3331 013624 005737 002404 1$: TST DOERR ;ANY PREV ERRORS?
3332 013630 001413 BEQ 2$ ;BR IF NO
3333 013632 005237 026054 INC TSERR ;TOTAL SOFT ERR CT
3334 013636 005237 026060 INL PSERR ;PASS SOFT ERR CT
3335 013642 032737 000020 013052 BIT #RNF,SRXES ;SEEK ERROR?
3336 013650 001402 BEQ 9$ ;BR IF NO
3337 013652 104067 ERROR 67 ;SOFT SEEK ERROR DURING READ
3338 013654 000401 BR 2$
3339 013656 104021 9$: ERROR 21 ;SOFT ERROR READ SECTOR
3340
3341 013660 005037 002404 2$: CLR DOERR
3342 013664 023727 003120 000005 CMP DXTST,#5 ;DOING WRITE DELETED DATA TESTS?
3343 013672 001005 BNE 6$ ;BR IF NO
3344 013674 032737 000040 177172 BIT #DD,RXES ;'DELETED DATA' SET?
3345 013702 001001 BNE 6$ ;BR IF YES
3346 013704 104052 ERROR 52 ;DD NOT SET
3347
3348 013706 000137 013712 6$: JMP DOEBUF ;GO TO EMPTY BUFFER
3349
3350
3351
3352
3353
3354
3355 ;:*****
3356 ;: HANDLER TO EMPTY THE DRIVE BUFFER FOR DX0
3357 ;:& POINT TO THE CHECK DATA HANDLER.
3358 ;:*****
3359
3360 013712 012703 000100 DOEBUF: MOV #100,R3 ;WORD CT
3361 013716 012701 002424 MOV #DOBUF,R1 ;BUFFER ADDRESS
3362 013722 012737 013750 013040 MOV #DOCDAT,DODISP ;POINT TO CHK DATA AFT INTER.
3363 013730 012737 000103 177170 MOV #<EBUF!IE>,RXCS ;ISSUE EMPTY BUFFER CMD
3364 013736 013721 177172 1$: MOV RXDB,(R1)+ ;GET WORD & STORE IT
3365 013742 005303 DEC R3
3366 013744 001374 BNE 1$
3367 013746 000002 RTI
3368
3369
    
```

```
3370 ;:*****
3371 ;:HANDLER TO CHECK DATA AFTER EMPTY BUFFER CMD ON DX0.
3372 ;:WILL GO TO NEXT SECTOR HANDLER WHEN NO ERR OR HARD ERR.
3373 ;:OR TO READ SECTOR (SAME) WHEN SOFT ERR.
3374 ;:THIS ROUTINE IS ALSO ENTERED FROM A CRC ERROR FROM A READ COMMAND.
3375 ;:*****
3376
3377 013750 005037 002412 DOCDAT: CLR DOCMER
3378 013754 012703 000100 MOV #100,R3 ;WORD CTR
3379 013760 013701 002406 MOV DOTRK,R1
3380 013764 000301 SWAB R1
3381 013766 053701 002410 BIS DOSEC,R1 ;R1 NOW HAS EXPECTED DATA
3382 013772 012702 002424 MOV #DOBUF,R2 ;GET BUFFER ADDR
3383 013776 020112 1$: CMP R1,(R2) ;COMPARE OK?
3384 014000 001407 BEQ 2$ ;BR IF YES
3385 014002 010137 002416 MOV R1,DOEXP ;ELSE SAVE
3386 014006 011237 002420 MOV (R2),DOREC
3387 014012 005237 002412 INC DOCMER
3388 014016 000405 BR 3$
3389
3390 014020 005303 2$: DEC R3
3391 014022 001403 BEQ 3$
3392 014024 062702 000002 ADD #2,R2
3393 014030 000762 BR 1$
3394
3395 014032 005737 002412 3$: TST DOCMER ;ANY COMP ERR?
3396 014036 001421 BEQ 4$ ;BR IF NO
3397 014040 005737 002422 TST DOCRC ;HERE FROM CRC ERROR?
3398 014044 001402 BEQ 14$ ;BR IF NO
3399 014046 104107 ERROR 107 ;DATA CRC ERROR
3400 014050 000431 BR 5$
3401
3402 014052 023727 002414 000012 14$: CMP DOCERR,#10. ;ELSE, 10 ERRS YET?
3403 014060 001033 BNE 7$ ;BR IF NO
3404 014062 023727 003120 000003 CMP DXTST,#3 ;DOING INIT TEST?
3405 014070 001002 BNE 8$ ;BR IN NO
3406 014072 104053 ERROR 53 ;INITIALIZE ERROR
3407 014074 000417 BR 5$
3408
3409 014076 104022 8$: ERROR 22 ;HARD ERROR DATA COMPARE
3410 014100 000415 BR 5$
3411
3412 014102 005737 002422 4$: TST DOCRC ;HERE FROM CRC ERROR?
3413 014106 001402 BEQ 15$ ;BR IF NO
3414 014110 104110 ERROR 110 ;CRC ERR WITH DATA OK
3415 014112 000410 BR 5$
3416
3417 014114 005737 002414 15$: TST DOCERR ;ANY PREV ERR?
3418 014120 001405 BEQ 5$ ;BR IF NO
3419 014122 005237 026054 INC TSERR ;TOTAL SOFT ERR CT
3420 014126 005237 026060 INC PSERR ;PASS SOFT ERR CT
3421 014132 104023 ERROR 23 ;SOFT ERROR DATA COMP
```

```

3422
3423 014134 005037 002414 5$: CLR DOCERR
3424 014140 005037 002422 CLR DOCRC
3425 014144 000137 014454 JMP DONEXT
3426
3427 014150 005237 002414 7$: INC DOCERR
3428 014154 023727 003120 000003 CMP DXTST,#3 ;DOING INIT TEST?
3429 014162 001402 BEQ 12$ ;BR IF YES
3430 014164 000137 013432 JMP DORSEC ;ELSE READ SECTOR OVER AGAIN
3431
3432 014170 000137 014612 12$: JMP DODUN ;ALL DONE
3433
3434
3435
3436 ;:*****
3437 ;:HANDLER TO SETUP NEXT SECTOR/TRACK FOR DX0.
3438 ;:WILL GO TO READ SECTOR (NEXT) IF ALL SECTORS ON TRACK NOT DONE.
3439 ;:WILL GO TO FILL BUFFER (NEXT TRACK) IF ALL SECTORS ON TRACK DONE.
3440 ;:WILL SHUT OFF DRV INTERRUPTS IF ALL TRACKS DONE, WITH A MESSAGE.
3441 ;:*****
3442
3443 014174 005037 002404 DONXT1: CLR DOERR
3444 014200 005037 002412 CLR DOCMER
3445 014204 023727 002410 000031 CMP DOSEC,#31 ;LAST ODD SECTOR?
3446 014212 001004 BNE 1$ ;BR IF NO
3447 014214 012737 000002 002410 MOV #2,DOSEC ;ELSE DO EVEN SECTORS NOW
3448 014222 000407 BR 2$
3449 014224 023727 002410 000032 1$: CMP DOSEC,#32 ;LAST EVEN SECTOR?
3450 014232 001405 BEQ 3$ ;BR IF YES
3451 014234 062737 000002 002410 ADD #2,DOSEC ;ELSE BUMP SECTOR
3452 014242 000137 013432 2$: JMP DORSEC ;GO READ SECTOR
3453
3454 014246 032777 010000 164664 3$: BIT #BIT12,@SWR ;SKIP TYPEOUT IF NOT SET
3455 014254 001411 BEQ 6$
3456 014256 104401 017076 TYPE ,DRV0
3457 014262 104401 017551 TYPE ,TRK
3458 014266 013746 002406 MOV DOTRK,-(SP) ;;SAVE DOTRK FOR TYPEOUT
3459 014272 104405 TYPDS ;:GO TYPE--DECIMAL ASCII WITH SIGN
3460 014274 104401 017622 TYPE ,DUN
3461 014300 005237 003122 6$: INC FIN ;SETUP TO GO TO DX1
3462 014304 023737 002406 003124 CMP DOTRK,MAXTRK ;LAST TRACK?
3463 014312 001406 BEQ 4$ ;BR IF YES
3464 014314 005237 002406 INC DOTRK ;BUMP TRACK
3465 014320 012737 000001 002410 MOV #1,DOSEC ;INIT SECTOR
3466 014326 000414 BR 5$
3467
3468 014330 032777 010000 164602 4$: BIT #BIT12,@SWR ;SKIP TYPEOUT IF NOT SET
3469 014336 001406 BEQ 7$
3470 014340 104401 017076 TYPE ,DRV0
3471 014344 104401 017556 TYPE ,PATT
3472 014350 104401 017622 TYPE ,DUN
3473 014354 005237 002400 7$: INC DOPAT ;PATT DONE FLAG
3474 014360 042737 000100 177170 5$: BIC #IE,RXCS ;DISABLE DX0 INTERRUPTS
3475 014366 000002 RTI
  
```

```

3476
3477
3478
3479
3480
3481 014370 005037 002404 DONXT2: CLR DOERR
3482 014374 005037 002412 CLR DOCMER
3483 014400 005237 003122 INC FIN ;SETUP TO GO TO DX1
3484 014404 005237 002402 INC DOCNT
3485 014410 023737 002402 003126 CMP DOCNT,MAXSK ;DID ALL SEEKS?
3486 014416 001012 BNE 1$ ;BR IF NO
3487 014420 032777 010000 164512 BIT #BIT12,@SWR ;SKIP TYPEOUT IF NOT SET
3488 014426 001406 BEQ 1$
3489 014430 104401 017075 TYPE ,DRVO
3490 014434 104401 017571 TYPE ,RANDOM
3491 014440 104401 017622 TYPE ,DUN
3492 014444 042737 000100 177170 1$: BIC #IE,RXCS ;DISABLE DX0 INTERRUPTS
3493 014452 000002 RTI
3494
3495
3496
3497
3498
3499 014454 023727 003120 000001 DONEXT: CMP DXTST,#1
3500 014462 001644 BEQ DONXT1
3501 014464 023727 003120 000002 CMP DXTST,#2
3502 014472 001736 BEQ DONXT2
3503 014474 000446 BR DODUN
3504

```

```
3505
3506
3507      :*****
3508      :HANDLER TO SETUP INITIALIZE COMMAND TEST
3509      :*****
3510 014476 012737 046032 177174 D0INIT: MOV    #46032,RXSA    ;TRY TO FAKE DRIVE TO GO TO TRK 76 SEC 26
3511                                     ;INIT SHOULD GO TO TRK 1, SEC 1.
3512 014504 012737 014612 013040         MOV    #DODUN,DODISP ;POINT TO DONE HANDLER
3513 014512 012737 000111 177170         MOV    #<INITAL!IE>,RXCS ;DO INIT COMMAND
3514 014520 000002
3515
3516
3517      :*****
3518      :HANDLERS TO SETUP RESTORE COMMAND TEST
3519      :*****
3520
3521 014522 012737 014612 013040 DOREST: MOV    #DODUN,DODISP ;POINT TO DONE HANDLER
3522 014530 012737 000117 177170         MOV    #<RESTOR!IE>,RXCS
3523 014536 000002
3524
3525 014540 012737 014626 013046 D1REST: MOV    #D1DUN,D1DISP
3526 014546 012737 000137 177170         MOV    #<RESTOR!IE!DX1>,RXCS
3527 014554 000002
3528
3529
3530      :*****
3531      :HANDLER TO SETUP READ STATUS COMMAND TEST
3532      :*****
3533 014556 012737 014612 013040 DOSTAT: MOV    #DODUN,DODISP ;POINT TO DONE
3534 014564 012737 000113 177170         MOV    #<RSTAT!IE>,RXCS ;DO READ STATUS COMMAND
3535 014572 000002
3536
3537      :*****
3538      :HANDLER TO START INVALID ADDRESS TESTS
3539      :*****
3540
3541 014574 012737 014612 013040 DOINV:  MOV    #DODUN,DODISP ;POINT TO DONE
3542 014602 012737 000105 177170         MOV    #<WSEC!IE>,RXCS ;DO WRITE SECTOR COMMAND
3543 014610 000002
3544
3545
3546      :*****
3547      :HANDLERS TO FINISH INITIALIZE, RESTORE, WRITE DELETED DATA,
3548      :READ STATUS & INVALID ADDRESS TESTS.
3549      :*****
3550
3551 014612 005237 003122          DODUN:  INC    FIN          ;SET DONE FLAG
3552 014616 042737 000100 177170         BIC    #IE,RXCS          ;DISABLE FURTHUR DX0 INTERRUPTS
3553 014624 000002
3554
3555 014626 005237 003122          D1DUN:  INC    FIN
3556 014632 042737 000120 177170         BIC    #<IE!DX1>,RXCS
3557 014640 000002
3558
```



```

3597
3598
3599
3600
3601
3602
3603
3604
3605 014762 005737 177170          D1WCHK: TST      RXCS          ;ERROR?
3606 014766 100024          BPL      1$              ;BR IN NO
3607 014770 023727 002654 000012  CMP      D1ERR,#10.      ;10 ERRORS?
3608 014776 001072          BNE      6$              ;BR IF NO & TRY AGAIN
3609 015000 004737 011672          JSR      PC,LD1BAD       ;LOAD BAD TRK/SEC TABLE
3610 015004 032737 000020 013052  BIT      #RNF,SRXES      ;SEEK ERROR?
3611 015012 001410          BEQ      7$              ;BR IF NO
3612 015014 104070          ERROR    70              ;HARD SEEK ERROR WRITE SECTOR
3613 015016 012737 015074 013046  MOV      #2$,D1DISP      ;RET TO 2$ AFT RESTORE
3614 015024 012737 000137 177170  MOV      #<RESTOR.IE!DX1>,RXCS
3615 015032 000002          RTI
3616
3617 015034 104024          7$:      ERROR    24          ;HARD ERROR WRITE SECTOR
3618 015036 000416          BR       2$              ;GO TO NEXT SECTOR
3619
3620 015040 005737 002654          1$:      TST      D1ERR      ;ANY PREV ERRORS?
3621 015044 001413          BEQ      2$              ;BR IF NO
3622 015046 005237 026054          INC      TSERR          ;TOTAL SOFT ERR CT
3623 015052 005237 026060          INC      PSERR          ;PASS SOFT ERR CT
3624 015056 032737 000020 013052  BIT      #RNF,SRXES      ;SEEK ERROR?
3625 015064 001402          BEQ      8$              ;BR IF NO
3626 015066 104071          ERROR    71              ;SOFT SEEK ERROR WRITE SECTOR
3627 015070 000401          BR       2$
3628 015072 104025          8$:      ERROR    25          ;SOFT ERROR WRITE SECTOR
3629
3630 015074 005037 002654          2$:      CLR      D1ERR
3631 015100 023727 002660 000031  CMP      D1SEC,#31      ;LAST ODD SECTOR?
3632 015106 001004          BNE      3$              ;BR IF NO
3633 015110 012737 000002 002660  MOV      #2,D1SEC        ;ELSE DO EVEN SECTORS NOW
3634 015116 000407          BR       4$
3635 015120 023727 002660 000032  3$:      CMP      D1SEC,#32      ;LAST EVEN SECTOR?
3636 015126 001405          BEQ      5$              ;BR IF YES
3637 015130 062737 000002 002660  ADD      #2,D1SEC        ;ELSE BUMP SECTOR
3638 015136 000137 014642          4$:      JMP      D1FBUF          ;GO TO FILL BUFFER & DO ANOTHER
3639
3640 015142 012737 000001 002660  5$:      MOV      #1,D1SEC        ;ALL SECTORS DONE...CHECK DATA
3641 015150 005037 002654          CLR      D1ERR
3642 015154 005037 002662          CLR      D1CMER
3643 015160 000137 015216          JMP      D1RSEC          ;GO TO READ SECTOR HANDLER
3644
3645 015164 005237 002654          6$:      INC      D1ERR
3646 015170 032737 000020 013052  BIT      #RNF,SRXES      ;SEEK ERROR?
3647 015176 001757          BEQ      4$              ;BR IF NO
3648 015200 012737 014642 013046  MOV      #D1FBUF,D1DISP  ;RET TO FILL BUFF AFT RESTORE
3649 015206 012737 000137 177170  MOV      #<RESTOR!IE!DX1>,RXCS
3650 015214 000002          RTI

```



```
3651
3652
3653
3654
3655
3656 015216 013701 002656 D1RSEC: MOV D1TRK,R1
3657 015222 000301 SWAB R1
3658 015224 053701 002660 BIS D1SEC,R1
3659 015230 010137 177174 MOV R1,RXSA ;LOAD TRK & SECTOR ADDR
3660
3661 015234 012737 015252 013046 MOV #D1RCHK,D1DISP ;POINT TO READ CHECK HANDLER
3662 015242 012737 000127 177170 MOV #<RSEC!IE.DX1>,RXCS ;ISSUE READ SECTOR COMMAND
3663 015250 000002 RTI
3664
3665
3666
3667
3668
3669
3670
3671
3672
3673
3674
3675 015252 005737 177170 D1RCHK: TST RXCS ;ERROR?
3676 015256 100054 BPL 1$ ;BR IF NO
3677 015260 023727 002654 000012 CMP D1ERR,#10. ;10 ERRORS?
3678 015266 001415 BEQ 3$ ;BR IF YES
3679 015270 005237 002654 INC D1ERR ;ELSE TRY AGAIN
3680 015274 032737 000020 013052 BIT #RNF,SRXES ;SEEK ERROR?
3681 015302 001745 BEQ D1RSEC ;BR IF NO
3682 015304 012737 015216 013046 MOV #D1RSEC,D1DISP ;ELSE DO RESTORE & RET TO RD SEC
3683 015312 012737 000137 177170 MOV #<RESTOR!IE!DX1>,RXCS
3684 015320 000002 RTI
3685
3686 015322 004737 011672 3$: JSR PC,LD1BAD ;LOAD BAD TRK/SEC TABLE
3687 015326 032737 000020 013052 BIT #RNF,SRXES ;SEEK ERROR?
3688 015334 001410 BEQ 7$ ;BR IF NO
3689 015336 104072 ERROR 72 ;HARD SEEK ERROR DURING READ
3690
3691 015340 012737 016206 013046 MOV #D1NEXT,D1DISP ;DO RESTORE
3692 015346 012737 000137 177170 MOV #<RESTOR!IE!DX1>,RXCS
3693 015354 000002 RTI
3694
3695 015356 032737 000010 013052 7$: BIT #CRC,SRXES ;CRC ERROR?
3696 015364 001404 BEQ 10$ ;BR IF NO
3697 015366 005237 002672 INC D1CRC ;ELSE SET FLAG
3698 015372 104111 ERROR 111 ;HARD CRC ERROR
3699 015374 000423 BR 2$ ;& GO TO EMP BUFF TO CHK DATA
3700
3701 015376 005037 002672 10$: CLR D1CRC
3702 015402 104026 ERROR 26 ;HARD READ ERROR
3703 015404 000137 016206 JMP D1NEXT
```

```

3704
3705 015410 005737 002654      1$:   TST    D1ERR      ;ANY PREV ERRORS?
3706 015414 001413              BEQ    2$           ;BR IF NO
3707 015416 005237 026054      INC    TSERR       ;TOTAL SOFT ERR CT
3708 015422 005237 026060      INC    PSERR       ;PASS SOFT ERR CT
3709 015426 032737 000020 013052 BIT    #RNF,SRXES  ;SEEK ERROR?
3710 015434 001402              BEQ    9$           ;BR IF NO
3711 015436 104073              ERROR  73          ;SOFT SEEK ERROR DURING READ
3712 015440 000401              BR     2$           ;
3713 015442 104027      9$:   ERROR  27          ;SOFT ERROR READ SECTOR
3714
3715 015444 005037 002654      2$:   CLR    D1ERR
3716 015450 000137 015454      JMP    D1EBUF      ;GO TO EMPTY BUFFER
3717
3718
3719
3720
3721
3722
3723      ::*****
3724      : HANDLER TO EMPTY THE DRIVE BUFFER FOR DX1
3725      : & POINT TO THE CHECK DATA HANDLER.
3726      :*****
3727
3728 015454 012703 000100      D1EBUF: MOV    #100,R3      ;WORD CT
3729 015460 012701 002674      MOV    #D1BUF,R1     ;BUFFER ADDRESS
3730 015464 012737 015512 013046 MOV    #D1DAT,D1DISP ;POINT TO CHK DATA AFT INTER.
3731 015472 012737 000123 177170 MOV    #<EBUF!IE!DX1>,RXCS ;ISSUE EMPTY BUFFER CMD
3732 015500 013721 177172      1$:   MOV    RXDB,(R1)+ ;GET WORD & STORE IT
3733 015504 005303              DEC    R3
3734 015506 001374              BNE    1$
3735 015510 000002              RTI
3736

```

```
3737
3738
3739
3740
3741
3742
3743
3744 015512 005037 002662 D1CDAT: CLR D1CMER
3745 015516 012703 000100 MOV #100,R3 ;WORD CTR
3746 015522 013701 002656 MOV D1TRK,R1
3747 015526 000301 SWAB R1
3748 015530 053701 002660 BIS D1SEC,R1
3749
3750 015534 005737 003132 TST COMP1 ;DONT SET BIT15 IN COMPATABILITY TESTS
3751 015540 001005 BNE 6$
3752 015542 005737 003134 TST COMP2
3753 015546 001002 BNE 6$
3754 015550 052701 100000 BIS #BIT15,R1 ;TO DISTINGUISH DX1 DATA FROM DX0 DATA
3755
3756 015554 012702 002674 6$: MOV #D1BUF,R2 ;GET BUFFER ADDR
3757 015560 020112 1$: CMP R1,(R2) ;COMPARE OK?
3758 015562 001407 BEQ 2$ ;BR IF YES
3759 015564 010137 002666 MOV R1,D1EXP ;ELSE SAVE
3760 015570 011237 002670 MOV (R2),D1REC
3761 015574 005237 002662 INC D1CMER
3762 015600 000405 BR 3$
3763 015602 005303 2$: DEC R3
3764 015604 001403 BEQ 3$
3765 015606 062702 000002 ADD #2,R2
3766 015612 000762 BR 1$
3767
3768 015614 005737 002662 3$: TST D1CMER ;ANY COMP ERR?
3769 015620 001413 BEQ 4$ ;BR IF NO
3770 015622 005737 002672 TST D1CRC ;HERE FROM CRC ERROR?
3771 015626 001402 BEQ 14$ ;BR IF NO
3772 015630 104112 ERROR 112 ;DATA CRC ERROR
3773 015632 000423 BR 5$
3774 015634 023727 002664 000012 14$: CMP D1CERR,#10. ;ELSE, 10 ERRS YET?
3775 015642 001025 BNE 7$ ;BR IF NO
3776 015644 104030 ERROR 30 ;HARD ERROR DATA COMP
3777 015646 000415 BR 5$
3778
3779 015650 005737 002672 4$: TST D1CRC ;HERE FROM CRC ERROR?
3780 015654 001402 BEQ 15$ ;BR IF NO
3781 015656 104113 ERROR 113 ;CRC ERR WITH DATA OK
3782 015660 000410 BR 5$
3783 015662 005737 002664 15$: TST D1CERR ;ANY ERR?
3784 015666 001405 BEQ 5$
3785 015670 005237 026054 INC TSERR ;TOTAL SOFT ERR CT
3786 015674 005237 026060 INC PSERR ;PASS SOFT ERR CT
3787 015700 104031 ERROR 31 ;SOFT ERROR DATA COMP
```

```

3788
3789 015702 005037 002664      5$:   CLR   D1CERR
3790 015706 005037 002672      CLR   D1CRC
3791 015712 000137 016206      JMP   D1NEXT
3792
3793 015716 005237 002664      7$:   INC   D1CERR
3794 015722 000137 015216      JMP   D1RSEC      ;READ SECTOR OVER AGAIN
3795
3796
3797      ;*****
3798      ;HANDLER TO SETUP NEXT SECTOR/TRACK FOR DX1.
3799      ;WILL GO TO READ SECTOR (NEXT) IF ALL SECTORS ON TRACK NOT DONE.
3800      ;WILL GO TO FILL BUFFER (NEXT TRACK) IF ALL SECTORS ON TRACK DONE.
3801      ;WILL SHUT OFF DRV INTERRUPTS IF ALL TRACKS DONE, WITH A MESSAGE.
3802      ;*****
3803
3804 015726 005037 002654      D1NXT1: CLR   D1ERR
3805 015732 005037 002662      CLR   D1CMER
3806 015736 023727 002660 000031      CMP   D1SEC,#31      ;LAST ODD SECTOR?
3807 015744 001004      BNE   1$             ;BR IF NO
3808 015746 012737 000002 002660      MOV   #2,D1SEC      ;ELSE DO EVEN SECTORS NOW
3809 015754 000407      BR    2$
3810 015756 023727 002660 000032 1$:   CMP   D1SEC,#32      ;LAST EVEN SECTOR?
3811 015764 001405      BEQ   3$             ;BR IF YES
3812 015766 062737 000002 002660      ADD   #2,D1SEC      ;ELSE BUMP SECTOR
3813 015774 000137 015216      2$:   JMP   D1RSEC      ;GO READ SECTOR
3814
3815 016000 032777 010000 163132 3$:   BIT   #BIT12,@SWR   ;SKIP TYPEOUT IF NOT SET
3816 016006 001411      BEQ   6$
3817 016010 104401 017103      TYPE  ,DRV1
3818 016014 104401 017551      TYPE  ,TRK
3819 016020 013746 002656      MOV   D1TRK,-(SP)   ;;SAVE D1TRK FOR TYPEOUT
3820 016024 104405      TYPDS ;GO TYPE--DECIMAL ASCII WITH SIGN
3821 016026 104401 017622      TYPE  ,DUN
3822 016032 005237 003122      6$:   INC   FIN           ;SETUP TO GO TO DX0
3823 016036 023737 002656 003124      CMP   D1TRK,MAXTRK ;LAST TRACK?
3824 016044 001406      BEQ   4$             ;BR IF YES
3825 016046 005237 002656      INC   D1TRK         ;BUMP TRACK
3826 016052 012737 000001 002660      MOV   #1,D1SEC     ;INIT SECTOR
3827 016060 000414      BR    5$
3828
3829 016062 032777 010000 163050 4$:   BIT   #BIT12,@SWR   ;SKIP TYPEOUT IF NOT SET
3830 016070 001406      BEQ   7$
3831 016072 104401 017103      TYPE  ,DRV1
3832 016076 104401 017556      TYPE  ,PATT
3833 016102 104401 017622      TYPE  ,DUN
3834 016106 005237 002650      7$:   INC   D1PAT        ;PATT DONE FLAG
3835 016112 042737 000120 177170 5$:   BIC   #<IE.DX1>,RXCS ;DISABLE DX1 INTERRUPTS
3836 016120 000002      RTI
3837

```



```

3864
3865
3866
3867
3868
3869
3870
3871
3872
3873 016220 005037 003130
3874 016224 104401 020021
3875 016230 000000
3876
3877 016232 004737 016540
3878
3879 016236 005037 016532
3880 016242 012737 000001 016534
3881 016250 012737 000001 016536
3882 016256 105737 177170
3883 016262 100375
3884
3885 016264 004537 016610
3886 016270 000007
3887
3888 016272 005737 177170
3889 016276 100011
3890 016300 005237 016532
3891 016304 023727 016532 000012
3892 016312 001364
3893 016314 104100
3894 016316 000000
3895 016320 000776
3896
3897 016322 005037 016532
3898 016326 004537 016650
3899 016332 002424
3900
3901 016334 004537 016610
3902 016340 000025
3903
3904 016342 005737 177170
3905 016346 100011
3906 016350 005237 016532
3907 016354 023727 016532 000012
3908 016362 001364
3909 016364 104101
3910 016366 000000
3911 016370 000776

```

```

:*****
:THE FOLLOWING CODE, UP TO THE PROGRAM MESSAGES, IS A UTILITY TO COPY
:THE DISKETTE CONTAINING ONLY THE BOOT & THE EXERCISER FROM DX0 TO DX1.
:THE OPERATOR CAN TYPE 'P' TO DO ANOTHER COPY OR
:TYPE '240G' TO ENTER NORMAL TESTING.
:ENTIRE CODE, INCLUDING TEXT & ERROR TABLE, IS 400(10) WORDS.
:*****

```

```

COPY:  CLR      COPFLG      ;DONT NEED FLAG ANY MORE
      TYPE     ,COPMSG      ;INSERT SCRATCH IN DX1 & TYPE P
      HALT                               ;WAIT FOR PROCEED
      JSR      PC,GETSEC      ;CALCULATE MAX SECTORS TO BE COPIED
      CLR      ERFLG          ;INITIALIZE
      MOV      #1,TRACK
      MOV      #1,SECT
      TSTB     RXCS           ;DONE? (CTL RDY)
      BPL      .-4           ;BR IF NO & TRY AGAIN
1$:    JSR      R5,RDWR      ;READ DX0
      RSEC
      TST      RXCS          ;ERROR?
      BPL      2$          ;BR IF NO
      INC      ERFLG
      CMP      ERFLG,#10.   ;10 ERRORS?
      BNE      1$          ;BR IF NO & TRY AGAIN
      ERROR    100         ;HARD ERR READ SECTOR DX0
      HALT
      BR       .-2         ;FORCE A RESTART FROM 250
2$:    CLR      ERFLG
      JSR      R5,EBUFF     ;EMPTY BUFFER INTO DOBUF
      DOBUF
3$:    JSR      R5,RDWR      ;WRITE DX1
      <WSEC!DX1>
      TST      RXCS          ;ERROR?
      BPL      4$          ;BR IF NO
      INC      ERFLG
      CMP      ERFLG,#10.   ;10 ERRORS?
      BNE      3$          ;BR IF NO & TRY AGAIN
      ERROR    101         ;HARD ERR WRITE SECTOR DX1
      HALT
      BR       .-2         ;FORCE RESTART AT 250

```

```

3912
3913 016372 005037 016532      4$: CLR      ERFLG
3914 016376 004537 016610      JSR      R5,RDWR      ;READ DX1
3915 016402 000027      <RSEC!DX1>
3916
3917 016404 005737 177170      TST      RXCS      ;ERROR?
3918 016410 100011      BPL      5$      ;BR IF NO
3919 016412 005237 016532      INC      ERFLG
3920 016416 023727 016532 000012      CMP      ERFLG,#10.  ;10 ERRORS?
3921 016424 001362      BNE      4$      ;BR IF NO & TRY AGAIN
3922 016426 104103      ERROR   103      ;HARD ERR READ SECTOR DX1
3923 016430 000000      HALT
3924 016432 000776      BR      .-2      ;FORCE 250 RESTART
3925
3926 016434 004537 016650      5$: JSR      R5,EBUFF  ;EMPTY BUFFER INTO D1BUF
3927 016440 002674      D1BUF
3928
3929 016442 004737 016712      JSR      PC,DATCHK  ;COMPARE DOBUF WITH D1BUF
3930
3931 016446 005237 016530      INC      SECCNT      ;TOTAL DONE
3932
3933 016452 023737 016530 016526      CMP      SECCNT,MAXSEC ;DID TOTAL # SECTORS?
3934 016460 001415      BEQ      8$      ;BR IF YES
3935
3936 016462 023727 016536 000032      CMP      SECT,#32   ;ELSE DID WE DO LAST SECTOR ON TRACK?
3937 016470 001403      BEQ      6$      ;BR IF YES
3938 016472 005237 016536      INC      SECT
3939 016476 000672      BR      1$      ;ELSE BUMP & DO ANOTHER
3940
3941 016500 005237 016534      6$: INC      TRACK      ;BUMP TRK
3942 016504 012737 000001 016536      MOV      #1,SECT   ;INIT SECTOR
3943 016512 000664      BR      1$      ;& DO ANOTHER
3944
3945 016514 104401 020122      8$: TYPE      ,COPDUN  ;DONE MSG
3946 016520 000000      HALT
3947 016522 000137 016220      JMP      COPY      ;DO AGAIN IF 'P' TYPED
3948
3949 016526 000000      MAXSEC: 0      ;TOTAL # SECTORS TO READ FROM DX0
3950 016530 000000      SECCNT: 0      ;TOTAL # SECTORS WRITTEN TO DX1
3951 016532 000000      ERFLG: 0
3952 016534 000000      TRACK: 0
3953 016536 000000      SECT: 0

```

3954
3955
3956
3957
3958
3959
3960
3961
3962
3963
3964
3965
3966
3967
3968
3969
3970
3971
3972
3973
3974
3975
3976
3977
3978
3979
3980
3981
3982
3983
3984
3985
3986
3987
3988
3989
3990
3991
3992
3993
3994
3995

016540 012700 026616
016544 006200
016546 005500
016550 062700 000377
016554 042700 000377
016560 000300
016562 006300
016564 006300
016566 062700 000074
016572 062700 000010
016576 010037 016526
016602 005037 016530
016606 000207
016610 013700 016534
016614 000300
016616 053700 016536
016622 010037 177174
016626 012537 177170
016632 004537 011442
016636 177170
016640 000200
016642 104075
016644 000000
016646 000205

:ROUTINE TO CALCULATE THE MAX # OF SECTORS TO READ FROM DX0

```
GETSEC: MOV    #LSTAD,RO      ;GET MAX ADDR
        ASR    RO            ;GET WORD CT
        ADC    RO            ;DONT LOOSE ANY
        ADD    #377,RO       ;ROUND OFF TO NEAREST WHOLE WORD
        BIC    #377,RO
        SWAB   RO
        ASL    RO
        ASL    RO            ;DIVIDE BY 100(8) FOR SECTOR CT.
        ADD    #<15.*4>,RO   ;ADD SECTOR CT FOR BLOCKS 0 - 14 OF DX0
        ADD    #8.,RO       ;2 MORE BLOCKS (NECESSARY FUDGE FACTOR)
        MOV    RO,MAXSEC    ;SAVE
                                ;THIS IS THE MAX SECTORS WE WILL READ OFF DX0
                                ;& WRITE TO DX1 STARTING AT TRK 1, SECT 1
        CLR    SECCNT       ;COUNT OF TOTAL SECTORS WRITTEN TO DX1
        RTS    PC
```

:ROUTINE TO READ OF WRITE DX0 OR DX1
:R5 POINTS TO THE COMMAND & DISK #

```
RDWR:  MOV    TRACK,RO
        SWAB   RO
        BIS    SECT,RO
        MOV    RO,RXSA      ;LOAD TRK/SEC
        MOV    (R5)+,RXCS   ;ISSUE COMMAND
        JSR    R5,TIMER2    ;WAIT FOR DONE
        RXCS  DONE
        ERROR 75           ;NO DONE
        HALT
        RTS    R5
```



```
3996  
3997  
3998  
3999  
4000  
4001  
4002 016650 012700 000100          EBUFF:  MOV    #100,R0          ;WD CT  
4003 016654 012501                MOV    (R5)+,R1          ;BUFFER ADDR  
4004 016656 012737 000003 177170    MOV    #EBUF,RXCS       ;ISSUE EMPTY BUFFER COMMAND  
4005 016664 013721 177172          1$:    MOV    RXDB,(R1)+     ;PUT IT IN TABLE  
4006 016670 005300                DEC    R0  
4007 016672 001374                BNE    1$  
4008 016674 004537 011442          JSR    R5,TIMER2        ;WAIT FOR DONE  
4009 016700 177170                RXCS  
4010 016702 000200                DONE  
4011 016704 104075                ERROR  75              ;NO DONE  
4012 016706 000000                HALT  
4013 016710 000205                RTS    R5  
4014  
4015  
4016  
4017  
4018  
4019  
4020 016712 012700 000100          DATCHK: MOV    #100,R0          ;WD CT  
4021 016716 012701 002424          MOV    #DOBUF,R1  
4022 016722 012702 002674          MOV    #D1BUF,R2  
4023 016726 022122          1$:    CMP    (R1)+,(R2)+     ;COMPARE?  
4024 016730 001403                BEQ    2$                ;BR IF YES  
4025 016732 104102                ERROR  102             ;MISCOMPARE  
4026 016734 000000                HALT  
4027 016736 000776                BR     .-2  
4028  
4029 016740 005300          2$:    DEC    R0  
4030 016742 001371          BNE    1$  
4031 016744 000207          RTS    PC  
4032
```

4033
4034
4035
4036
4037
4038
4039
4040
4041
4042
4043
4044
4045
4046
4047
4048
4049
4050
4051
4052
4053
4054
4055
4056
4057
4058
4059
4060
4061
4062
4063
4064
4065
4066
4067
4068
4069
4070
4071
4072
4073
4074
4075
4076
4077
4078
4079
4080
4081
4082
4083
4084
4085
4086

016746 041600 052514 052123
016754 051105 047440 052120
016762 047511 000116
016766 052200 051105 020115
016774 030443 000
016777 200 042524 046522
017004 021440 000062
017010 052200 051105 020115
017016 031443 000
017021 200 054523 041516
017026 041440 046517 000115
017034 040600 054523 041516
017042 041440 046517 000115
017050 050200 044522 052116
017056 051105 000
017061 040 047520 052122
017066 052040 051505 042524
017074 000104
017076 042200 030130 000
017103 200 054104 000061
017110 041600 047514 045503
017116 000
017117 200 045464 000
017123 200 045470 000
017127 200 031061 000113
017134 030600 045466 000
017141 200 030062 000113
017146 031200 045464 000
017153 200 030063 000113
017160 046440 046505 051117
017166 020131 051120 051505
017174 047105 000124

017200 047040 052117 050040
017206 042522 042523 052116
017214 000
017215 200 044412 051516
017222 051105 020124 041523
017230 040522 041524 020110
017236 044504 045523 026123
017244 052040 050131 020105
017252 050047 020047 047506
017260 020122 047516 046522
017266 046101 052040 051505
017274 044524 043516
017300 023600 032062 043460

.SBTTL PROGRAM MESSAGES
:*****
: MESSAGES & ERROR FORMATS
:*****
CLOPT: .ASCIZ <CRLF>/CLUSTER OPTION/
CLT1: .ASCIZ <CRLF>/TERM #1/
CLT2: .ASCIZ <CRLF>/TERM #2/
CLT3: .ASCIZ <CRLF>/TERM #3/
SCOM: .ASCIZ <CRLF>/SYNC COMM/
ACOM: .ASCIZ <CRLF>/ASYNC COMM/
PRINT: .ASCIZ <CRLF>/PRINTER/
PORT: .ASCIZ / PORT TESTED/
DRV0: .ASCIZ <CRLF>/DX0/
DRV1: .ASCIZ <CRLF>/DX1/
CLOCK: .ASCIZ <CRLF>/CLOCK/
M4K: .ASCIZ <CRLF>/4K/
M8K: .ASCIZ <CRLF>/8K/
M12K: .ASCIZ <CRLF>/12K/
M16K: .ASCIZ <CRLF>/16K/
M20K: .ASCIZ <CRLF>/20K/
M24K: .ASCIZ <CRLF>/24K/
M30K: .ASCIZ <CRLF>/30K/
MEM: .ASCIZ / MEMORY PRESENT/

NOTPRES: .ASCIZ / NOT PRESENT/
WARNING: .ASCII <CRLF><LF>/INSERT SCRATCH DISKS, TYPE 'P' FOR NORMAL TESTING/

.ASCII <CRLF>/'240G' FOR NORMAL RESTARTS/

| | | | | | | |
|------|--------|--------|--------|--------|---------|--|
| 4087 | 017306 | 020047 | 047506 | 020122 | | |
| 4088 | 017314 | 047516 | 046522 | 046101 | | |
| 4089 | 017322 | 051040 | 051505 | 040524 | | |
| 4090 | 017330 | 052122 | 123 | | | |
| 4091 | 017333 | 200 | 031047 | 030065 | .ASCII | <CRLF>/'250G' TO COPY SYS EXERCISER DISK/ |
| 4092 | 017340 | 023507 | 052040 | 020117 | | |
| 4093 | 017346 | 047503 | 054520 | 051440 | | |
| 4094 | 017354 | 051531 | 042440 | 042530 | | |
| 4095 | 017362 | 041522 | 051511 | 051105 | | |
| 4096 | 017370 | 042040 | 051511 | 113 | | |
| 4097 | 017375 | 200 | 031047 | 030066 | .ASCII | <CRLF>/'260G' FOR COMPATABILITY PASS 1: WRITE/ |
| 4098 | 017402 | 023507 | 043040 | 051117 | | |
| 4099 | 017410 | 041440 | 046517 | 040520 | | |
| 4100 | 017416 | 040524 | 044502 | 044514 | | |
| 4101 | 017424 | 054524 | 050040 | 051501 | | |
| 4102 | 017432 | 020123 | 035061 | 053440 | | |
| 4103 | 017440 | 044522 | 042524 | | | |
| 4104 | 017444 | 023600 | 033462 | 043460 | .ASCIZ | <CRLF>/'270G' FOR COMPATABILITY PASS 2: READ/ |
| 4105 | 017452 | 020047 | 047506 | 020122 | | |
| 4106 | 017460 | 047503 | 050115 | 052101 | | |
| 4107 | 017466 | 041101 | 046111 | 052111 | | |
| 4108 | 017474 | 020131 | 040520 | 051523 | | |
| 4109 | 017502 | 031040 | 020072 | 042522 | | |
| 4110 | 017510 | 042101 | 000 | | | |
| 4111 | 017513 | 040 | 042524 | 052123 | DROP: | .ASCIZ / TESTING DROPPED/ |
| 4112 | 017520 | 047111 | 020107 | 051104 | | |
| 4113 | 017526 | 050117 | 042520 | 000104 | | |
| 4114 | 017534 | 051040 | 047125 | 044516 | RUN: | .ASCIZ / RUNNING/ |
| 4115 | 017542 | 043516 | 000 | | | |
| 4116 | 017545 | 040 | 045517 | 000 | OK: | .ASCIZ / OK/ |
| 4117 | 017551 | 040 | 051124 | 000113 | TRK: | .ASCIZ / TRK/ |
| 4118 | 017556 | 042040 | 052101 | 020101 | PATT: | .ASCIZ / DATA PATT/ |
| 4119 | 017564 | 040520 | 052124 | 000 | | |
| 4120 | 017571 | 040 | 040522 | 042116 | RANDOM: | .ASCIZ / RANDOM SEEKS/ |
| 4121 | 017576 | 046517 | 051440 | 042505 | | |
| 4122 | 017604 | 051513 | 000 | | | |
| 4123 | 017607 | 200 | 042515 | 020115 | MEMORY: | .ASCIZ <CRLF>/MEM TESTS/ |
| 4124 | 017614 | 042524 | 052123 | 000123 | | |
| 4125 | 017622 | 042040 | 047117 | 000105 | DUN: | .ASCIZ / DONE/ |
| 4126 | 017630 | 005200 | 047105 | 020104 | ENDPAS: | .ASCIZ <CRLF><LF>/END PASS #/ |
| 4127 | 017636 | 040520 | 051523 | 021440 | | |
| 4128 | 017644 | 000 | | | | |
| 4129 | 017645 | 011 | 047524 | 040524 | MSG1: | .ASCIZ / TOTAL ERRORS: / |
| 4130 | 017652 | 020114 | 051105 | 047522 | | |
| 4131 | 017660 | 051522 | 004472 | 020040 | | |
| 4132 | 017666 | 020040 | 020040 | 000040 | | |
| 4133 | 017674 | 004600 | 004411 | 047524 | MSG2: | .ASCIZ <CRLF>/ TOTAL SOFT ERRORS: / |
| 4134 | 017702 | 040524 | 020114 | 047523 | | |
| 4135 | 017710 | 052106 | 042440 | 051122 | | |
| 4136 | 017716 | 051117 | 035123 | 020040 | | |
| 4137 | 017724 | 020040 | 000040 | | | |
| 4138 | 017730 | 005200 | 004411 | 052011 | MSG3: | .ASCIZ <CRLF><LF>/ TOTAL ERRORS THIS PASS: / |
| 4139 | 017736 | 052117 | 046101 | 042440 | | |
| 4140 | 017744 | 051122 | 051117 | 020123 | | |

| | | | | |
|------|--------|--------|--------|--------|
| 4141 | 017752 | 044124 | 051511 | 050040 |
| 4142 | 017760 | 051501 | 035123 | 000 |
| 4143 | 017765 | 200 | 004411 | 051411 |
| 4144 | 017772 | 043117 | 020124 | 042440 |
| 4145 | 020000 | 051122 | 051117 | 020123 |
| 4146 | 020006 | 044124 | 051511 | 050040 |
| 4147 | 020014 | 051501 | 035123 | 000 |
| 4148 | 020021 | 200 | 047503 | 054520 |
| 4149 | 020026 | 042040 | 030130 | 052040 |
| 4150 | 020034 | 020117 | 054104 | 061 |
| 4151 | 020041 | 200 | 047111 | 042523 |
| 4152 | 020046 | 052122 | 051440 | 051103 |
| 4153 | 020054 | 052101 | 044103 | 042040 |
| 4154 | 020062 | 051511 | 020113 | 047111 |
| 4155 | 020070 | 042040 | 030530 | 020054 |
| 4156 | 020076 | 054524 | 042520 | 023440 |
| 4157 | 020104 | 023520 | 052040 | 020117 |
| 4158 | 020112 | 051120 | 041517 | 042505 |
| 4159 | 020120 | 000104 | | |
| 4160 | 020122 | 042200 | 047117 | 027105 |
| 4161 | 020130 | 027056 | 054524 | 042520 |
| 4162 | 020136 | 023440 | 023520 | 052040 |
| 4163 | 020144 | 020117 | 047504 | 040440 |
| 4164 | 020152 | 040507 | 047111 | 047440 |
| 4165 | 020160 | 020122 | 031047 | 030064 |
| 4166 | 020166 | 023507 | 043040 | 051117 |
| 4167 | 020174 | 047040 | 051117 | 040515 |
| 4168 | 020202 | 020114 | 042524 | 052123 |
| 4169 | 020210 | 047111 | 000107 | |
| 4170 | 020214 | 005200 | 047503 | 050115 |
| 4171 | 020222 | 052101 | 041101 | 046111 |
| 4172 | 020230 | 052111 | 020131 | 040520 |
| 4173 | 020236 | 051523 | 030440 | 024040 |
| 4174 | 020244 | 051127 | 052111 | 024505 |
| 4175 | 020252 | 042040 | 047117 | 105 |
| 4176 | 020257 | 200 | 047504 | 023440 |
| 4177 | 020264 | 023520 | 043040 | 051117 |
| 4178 | 020272 | 050040 | 051501 | 020123 |
| 4179 | 020300 | 020062 | 051050 | 040505 |
| 4180 | 020306 | 024504 | 000 | |
| 4181 | 020311 | 200 | 042504 | 046526 |
| 4182 | 020316 | 036440 | 000040 | |
| 4183 | 020322 | 042515 | 047515 | 054522 |
| 4184 | 020330 | 052040 | 046511 | 047505 |
| 4185 | 020336 | 052125 | 000 | |
| 4186 | 020341 | 115 | 046505 | 042040 |
| 4187 | 020346 | 052101 | 020101 | 047503 |
| 4188 | 020354 | 050115 | 042440 | 051122 |
| 4189 | 020362 | 000 | | |
| 4190 | 020363 | 123 | 047131 | 020103 |
| 4191 | 020370 | 047503 | 046515 | 042040 |
| 4192 | 020376 | 042111 | 047040 | 052117 |
| 4193 | 020404 | 051040 | 041505 | 044505 |
| 4194 | 020412 | 042526 | 051440 | 047131 |

MSG4: .ASCIZ <CRLF>/ SOFT ERRORS THIS PASS:/

COPMSG: .ASCII <CRLF>/COPY DX0 TO DX1/

.ASCIZ <CRLF>/INSERT SCRATCH DISK IN DX1, TYPE 'P' TO PROCEED/

COPDUN: .ASCIZ <CRLF>/DONE...TYPE 'P' TO DO AGAIN OR '240G' FOR NORMAL TESTING/

COMPAT: .ASCII <CRLF><LF>/COMPATABILITY PASS 1 (WRITE) DONE/

.ASCIZ <CRLF>/DO 'P' FOR PASS 2 (READ)/

DEVMM: .ASCIZ <CRLF>/DEVMM /

EM1: .ASCIZ /MEMORY TIMEOUT/

EM2: .ASCIZ /MEM DATA COMP ERR/

EM3: .ASCIZ /SYNC COMM DID NOT RECEIVE SYNC CHAR/

| | | | | | | |
|------|--------|--------|--------|--------|-------|-----------------------------------|
| 4195 | 020420 | 020103 | 044103 | 051101 | | |
| 4196 | 020426 | 000 | | | | |
| 4197 | 020427 | 120 | 044522 | 052116 | EM4: | .ASCIZ /PRINTER STATUS ERR/ |
| 4198 | 020434 | 051105 | 051440 | 040524 | | |
| 4199 | 020442 | 052524 | 020123 | 051105 | | |
| 4200 | 020450 | 000122 | | | | |
| 4201 | 020452 | 042524 | 046522 | 021440 | EM5: | .ASCIZ /TERM #1 DATA COMP ERR/ |
| 4202 | 020460 | 020061 | 040504 | 040524 | | |
| 4203 | 020466 | 041440 | 046517 | 020120 | | |
| 4204 | 020474 | 051105 | 000122 | | | |
| 4205 | 020500 | 042524 | 046522 | 021440 | EM6: | .ASCIZ /TERM #2 DATA COMP ERR/ |
| 4206 | 020506 | 020062 | 040504 | 040524 | | |
| 4207 | 020514 | 041440 | 046517 | 020120 | | |
| 4208 | 020522 | 051105 | 000122 | | | |
| 4209 | 020526 | 042524 | 046522 | 021440 | EM7: | .ASCIZ /TERM #3 DATA COMP ERR/ |
| 4210 | 020534 | 020063 | 040504 | 040524 | | |
| 4211 | 020542 | 041440 | 046517 | 020120 | | |
| 4212 | 020550 | 051105 | 000122 | | | |
| 4213 | 020554 | 051501 | 047131 | 020103 | EM8: | .ASCIZ /ASYNC COMM DATA COMP ERR/ |
| 4214 | 020562 | 047503 | 046515 | 042040 | | |
| 4215 | 020570 | 052101 | 020101 | 047503 | | |
| 4216 | 020576 | 050115 | 042440 | 051122 | | |
| 4217 | 020604 | 000 | | | | |
| 4218 | 020605 | 123 | 047131 | 020103 | EM9: | .ASCIZ /SYNC COMM DATA COMP ERR/ |
| 4219 | 020612 | 047503 | 046515 | 042040 | | |
| 4220 | 020620 | 052101 | 020101 | 047503 | | |
| 4221 | 020626 | 050115 | 042440 | 051122 | | |
| 4222 | 020634 | 000 | | | | |
| 4223 | 020635 | 125 | 042516 | 050130 | EM10: | .ASCIZ /UNEXP DISK INTERRUPT/ |
| 4224 | 020642 | 042040 | 051511 | 020113 | | |
| 4225 | 020650 | 047111 | 042524 | 051122 | | |
| 4226 | 020656 | 050125 | 000124 | | | |
| 4227 | 020662 | 054104 | 020060 | 051127 | EM11: | .ASCIZ /DXO WRITE ERR/ |
| 4228 | 020670 | 052111 | 020105 | 051105 | | |
| 4229 | 020676 | 000122 | | | | |
| 4230 | 020700 | 054104 | 020060 | 040510 | EM13: | .ASCIZ /DXO HARD ERR - READ SECT/ |
| 4231 | 020706 | 042122 | 042440 | 051122 | | |
| 4232 | 020714 | 026440 | 051040 | 040505 | | |
| 4233 | 020722 | 020104 | 042523 | 052103 | | |
| 4234 | 020730 | 000 | | | | |
| 4235 | 020731 | 104 | 030130 | 051440 | EM14: | .ASCIZ /DXO SOFT ERR - READ SECT/ |
| 4236 | 020736 | 043117 | 020124 | 051105 | | |
| 4237 | 020744 | 020122 | 020055 | 042522 | | |
| 4238 | 020752 | 042101 | 051440 | 041505 | | |
| 4239 | 020760 | 000124 | | | | |
| 4240 | 020762 | 054104 | 020060 | 040510 | EM15: | .ASCIZ /DXO HARD ERR - DATA COMP/ |
| 4241 | 020770 | 042122 | 042440 | 051122 | | |
| 4242 | 020776 | 026440 | 042040 | 052101 | | |
| 4243 | 021004 | 020101 | 047503 | 050115 | | |
| 4244 | 021012 | 000 | | | | |
| 4245 | 021013 | 104 | 030130 | 051440 | EM16: | .ASCIZ /DXO SOFT ERR - DATA COMP/ |
| 4246 | 021020 | 043117 | 020124 | 051105 | | |
| 4247 | 021026 | 020122 | 020055 | 040504 | | |
| 4248 | 021034 | 040524 | 041440 | 046517 | | |

| | | | | | | | | | |
|------|--------|--------|--------|--------|-------|--------|-----------------------------|--|--|
| 4249 | 021042 | 000120 | | | | | | | |
| 4250 | 021044 | 054104 | 020061 | 040510 | EM17: | .ASCIZ | /DX1 HARD ERR - WRITE SECT/ | | |
| 4251 | 021052 | 042122 | 042440 | 051122 | | | | | |
| 4252 | 021060 | 026440 | 053440 | 044522 | | | | | |
| 4253 | 021066 | 042524 | 051440 | 041505 | | | | | |
| 4254 | 021074 | 000124 | | | | | | | |
| 4255 | 021076 | 054104 | 020061 | 047523 | EM18: | .ASCIZ | /DX1 SOFT ERR - WRITE SECT/ | | |
| 4256 | 021104 | 052106 | 042440 | 051122 | | | | | |
| 4257 | 021112 | 026440 | 053440 | 044522 | | | | | |
| 4258 | 021120 | 042524 | 051440 | 041505 | | | | | |
| 4259 | 021126 | 000124 | | | | | | | |
| 4260 | 021130 | 054104 | 020061 | 040510 | EM19: | .ASCIZ | /DX1 HARD ERR - READ SECT/ | | |
| 4261 | 021136 | 042122 | 042440 | 051122 | | | | | |
| 4262 | 021144 | 026440 | 051040 | 040505 | | | | | |
| 4263 | 021152 | 020104 | 042523 | 052103 | | | | | |
| 4264 | 021160 | 000 | | | | | | | |
| 4265 | 021161 | 104 | 030530 | 051440 | EM20: | .ASCIZ | /DX1 SOFT ERR - READ SECT/ | | |
| 4266 | 021166 | 043117 | 020124 | 051105 | | | | | |
| 4267 | 021174 | 020122 | 020055 | 042522 | | | | | |
| 4268 | 021202 | 042101 | 051440 | 041505 | | | | | |
| 4269 | 021210 | 000124 | | | | | | | |
| 4270 | 021212 | 054104 | 020061 | 040510 | EM21: | .ASCIZ | /DX1 HARD ERR - DATA COMP/ | | |
| 4271 | 021220 | 042122 | 042440 | 051122 | | | | | |
| 4272 | 021226 | 026440 | 042040 | 052101 | | | | | |
| 4273 | 021234 | 020101 | 047503 | 050115 | | | | | |
| 4274 | 021242 | 000 | | | | | | | |
| 4275 | 021243 | 104 | 030530 | 051440 | EM22: | .ASCIZ | /DX1 SOFT ERR - DATA COMP/ | | |
| 4276 | 021250 | 043117 | 020124 | 051105 | | | | | |
| 4277 | 021256 | 020122 | 020055 | 040504 | | | | | |
| 4278 | 021264 | 040524 | 041440 | 046517 | | | | | |
| 4279 | 021272 | 000120 | | | | | | | |
| 4280 | 021274 | 046103 | 020113 | 052510 | EM23: | .ASCIZ | /CLK HUNG/ | | |
| 4281 | 021302 | 043516 | 000 | | | | | | |
| 4282 | 021305 | 120 | 044522 | 052116 | EM24: | .ASCIZ | /PRINTER HUNG/ | | |
| 4283 | 021312 | 051105 | 044040 | 047125 | | | | | |
| 4284 | 021320 | 000107 | | | | | | | |
| 4285 | 021322 | 042524 | 046522 | 021440 | EM25: | .ASCIZ | /TERM #1 HUNG/ | | |
| 4286 | 021330 | 020061 | 052510 | 043516 | | | | | |
| 4287 | 021336 | 000 | | | | | | | |
| 4288 | 021337 | 124 | 051105 | 020115 | EM26: | .ASCIZ | /TERM #2 HUNG/ | | |
| 4289 | 021344 | 031043 | 044040 | 047125 | | | | | |
| 4290 | 021352 | 000107 | | | | | | | |
| 4291 | 021354 | 042524 | 046522 | 021440 | EM27: | .ASCIZ | /TERM #3 HUNG/ | | |
| 4292 | 021362 | 020063 | 052510 | 043516 | | | | | |
| 4293 | 021370 | 000 | | | | | | | |
| 4294 | 021371 | 123 | 047131 | 020103 | EM28: | .ASCIZ | /SYNC COMM HUNG/ | | |
| 4295 | 021376 | 047503 | 046515 | 044040 | | | | | |
| 4296 | 021404 | 047125 | 000107 | | | | | | |
| 4297 | 021410 | 051501 | 047131 | 020103 | EM29: | .ASCIZ | /ASYNC COMM HUNG/ | | |
| 4298 | 021416 | 047503 | 046515 | 044040 | | | | | |
| 4299 | 021424 | 047125 | 000107 | | | | | | |
| 4300 | 021430 | 054104 | 020060 | 052510 | EM30: | .ASCIZ | /DX0 HUNG/ | | |
| 4301 | 021436 | 043516 | 000 | | | | | | |
| 4302 | 021441 | 104 | 030530 | 044040 | EM31: | .ASCIZ | /DX1 HUNG/ | | |

| | | | | | |
|------|--------|--------|--------|--------|---|
| 4303 | 021446 | 047125 | 000107 | | |
| 4304 | 021452 | 054523 | 041516 | 041440 | EM32: .ASCIZ /SYNC COMM - DATA NOT AVAIL NOT SET/ |
| 4305 | 021460 | 046517 | 020115 | 020055 | |
| 4306 | 021466 | 040504 | 040524 | 047040 | |
| 4307 | 021474 | 052117 | 040440 | 040526 | |
| 4308 | 021502 | 046111 | 047040 | 052117 | |
| 4309 | 021510 | 051440 | 052105 | 000 | |
| 4310 | 021515 | 123 | 047131 | 020103 | EM33: .ASCIZ /SYNC COMM - RECVR ACTIVE NOT SET/ |
| 4311 | 021522 | 047503 | 046515 | 026440 | |
| 4312 | 021530 | 051040 | 041505 | 051126 | |
| 4313 | 021536 | 040440 | 052103 | 053111 | |
| 4314 | 021544 | 020105 | 047516 | 020124 | |
| 4315 | 021552 | 042523 | 000124 | | |
| 4316 | 021556 | 054523 | 041516 | 041440 | EM34: .ASCIZ /SYNC COMM - RECVR DONE NOT SET/ |
| 4317 | 021564 | 046517 | 020115 | 020055 | |
| 4318 | 021572 | 042522 | 053103 | 020122 | |
| 4319 | 021600 | 047504 | 042516 | 047040 | |
| 4320 | 021606 | 052117 | 051440 | 052105 | |
| 4321 | 021614 | 000 | | | |
| 4322 | 021615 | 104 | 030130 | 023440 | EM35: .ASCIZ /DXO 'DEL DATA' BIT 5 NOT SET IN RXES/ |
| 4323 | 021622 | 042504 | 020114 | 040504 | |
| 4324 | 021630 | 040524 | 020047 | 044502 | |
| 4325 | 021636 | 020124 | 020065 | 047516 | |
| 4326 | 021644 | 020124 | 042523 | 020124 | |
| 4327 | 021652 | 047111 | 051040 | 042530 | |
| 4328 | 021660 | 000123 | | | |
| 4329 | 021662 | 054104 | 020060 | 047111 | EM36: .ASCIZ /DXO INIT CMD DID NOT READ TRK 1, SEC 1/ |
| 4330 | 021670 | 052111 | 041440 | 042115 | |
| 4331 | 021676 | 042040 | 042111 | 047040 | |
| 4332 | 021704 | 052117 | 051040 | 040505 | |
| 4333 | 021712 | 020104 | 051124 | 020113 | |
| 4334 | 021720 | 026061 | 051440 | 041505 | |
| 4335 | 021726 | 030440 | 000 | | |
| 4336 | 021731 | 104 | 030130 | 051040 | EM37: .ASCIZ /DXO RESTORE CMD ERR/ |
| 4337 | 021736 | 051505 | 047524 | 042522 | |
| 4338 | 021744 | 041440 | 042115 | 042440 | |
| 4339 | 021752 | 051122 | 000 | | |
| 4340 | 021755 | 104 | 030130 | 023440 | EM38: .ASCIZ /DXO 'INIT DONE' BIT 0 NOT SET IN RXES/ |
| 4341 | 021762 | 047111 | 052111 | 042040 | |
| 4342 | 021770 | 047117 | 023505 | 041040 | |
| 4343 | 021776 | 052111 | 030040 | 047040 | |
| 4344 | 022004 | 052117 | 051440 | 052105 | |
| 4345 | 022012 | 044440 | 020116 | 054122 | |
| 4346 | 022020 | 051505 | 000 | | |
| 4347 | 022023 | 104 | 030130 | 051040 | EM39: .ASCIZ /DXO READ STATUS CMD ERR/ |
| 4348 | 022030 | 040505 | 020104 | 052123 | |
| 4349 | 022036 | 052101 | 051525 | 041440 | |
| 4350 | 022044 | 042115 | 042440 | 051122 | |
| 4351 | 022052 | 000 | | | |
| 4352 | 022053 | 104 | 030130 | 044440 | EM40: .ASCIZ /DXO INV ADDR BIT 1 NOT SET IN RXES/ |
| 4353 | 022060 | 053116 | 040440 | 042104 | |
| 4354 | 022066 | 020122 | 044502 | 020124 | |
| 4355 | 022074 | 020061 | 047516 | 020124 | |
| 4356 | 022102 | 042523 | 020124 | 047111 | |

| | | | | | |
|------|--------|--------|--------|--------|---|
| 4357 | 022110 | 051040 | 042530 | 000123 | |
| 4358 | 022116 | 054104 | 020060 | 051105 | EM41: .ASCIZ /DX0 ERR BIT NOT SET IN RXCS/ |
| 4359 | 022124 | 020122 | 044502 | 020124 | |
| 4360 | 022132 | 047516 | 020124 | 042523 | |
| 4361 | 022140 | 020124 | 047111 | 051040 | |
| 4362 | 022146 | 041530 | 000123 | | |
| 4363 | 022152 | 054104 | 020060 | 047111 | EM42: .ASCIZ /DX0 INIT CMD ERR/ |
| 4364 | 022160 | 052111 | 041440 | 042115 | |
| 4365 | 022166 | 042440 | 051122 | 000 | |
| 4366 | 022173 | 104 | 030130 | 051440 | EM43: .ASCIZ /DX0 SEEK ERR - WRITE SECT/ |
| 4367 | 022200 | 042505 | 020113 | 051105 | |
| 4368 | 022206 | 020122 | 020055 | 051127 | |
| 4369 | 022214 | 052111 | 020105 | 042523 | |
| 4370 | 022222 | 052103 | 000 | | |
| 4371 | 022225 | 104 | 030130 | 044040 | EM45: .ASCIZ /DX0 HARD SEEK ERR - READ SECT/ |
| 4372 | 022232 | 051101 | 020104 | 042523 | |
| 4373 | 022240 | 045505 | 042440 | 051122 | |
| 4374 | 022246 | 026440 | 051040 | 040505 | |
| 4375 | 022254 | 020104 | 042523 | 052103 | |
| 4376 | 022262 | 000 | | | |
| 4377 | 022263 | 104 | 030130 | 051440 | EM46: .ASCIZ /DX0 SOFT SEEK ERR - READ SECT/ |
| 4378 | 022270 | 043117 | 020124 | 042523 | |
| 4379 | 022276 | 045505 | 042440 | 051122 | |
| 4380 | 022304 | 026440 | 051040 | 040505 | |
| 4381 | 022312 | 020104 | 042523 | 052103 | |
| 4382 | 022320 | 000 | | | |
| 4383 | 022321 | 104 | 030530 | 044040 | EM47: .ASCIZ /DX1 HARD SEEK ERR - WRITE SECT/ |
| 4384 | 022326 | 051101 | 020104 | 042523 | |
| 4385 | 022334 | 045505 | 042440 | 051122 | |
| 4386 | 022342 | 026440 | 053440 | 044522 | |
| 4387 | 022350 | 042524 | 051440 | 041505 | |
| 4388 | 022356 | 000124 | | | |
| 4389 | 022360 | 054104 | 020061 | 047523 | EM48: .ASCIZ /DX1 SOFT SEEK ERR - WRITE SECT/ |
| 4390 | 022366 | 052106 | 051440 | 042505 | |
| 4391 | 022374 | 020113 | 051105 | 020122 | |
| 4392 | 022402 | 020055 | 051127 | 052111 | |
| 4393 | 022410 | 020105 | 042523 | 052103 | |
| 4394 | 022416 | 000 | | | |
| 4395 | 022417 | 104 | 030530 | 044040 | EM49: .ASCIZ /DX1 HARD SEEK ERR - READ SECT/ |
| 4396 | 022424 | 051101 | 020104 | 042523 | |
| 4397 | 022432 | 045505 | 042440 | 051122 | |
| 4398 | 022440 | 026440 | 051040 | 040505 | |
| 4399 | 022446 | 020104 | 042523 | 052103 | |
| 4400 | 022454 | 000 | | | |
| 4401 | 022455 | 104 | 030530 | 051440 | EM50: .ASCIZ /DX1 SOFT SEEK ERR - READ SECT/ |
| 4402 | 022462 | 043117 | 020124 | 042523 | |
| 4403 | 022470 | 045505 | 042440 | 051122 | |
| 4404 | 022476 | 026440 | 051040 | 040505 | |
| 4405 | 022504 | 020104 | 042523 | 052103 | |
| 4406 | 022512 | 000 | | | |
| 4407 | 022513 | 104 | 030530 | 051040 | EM51: .ASCIZ /DX1 RESTORE CMD ERR/ |
| 4408 | 022520 | 051505 | 047524 | 042522 | |
| 4409 | 022526 | 041440 | 042115 | 042440 | |
| 4410 | 022534 | 051122 | 000 | | |

| | | | | | | |
|------|--------|--------|--------|--------|-------|---|
| 4411 | 022537 | 104 | 051511 | 020113 | EM52: | .ASCIZ /DISK 'DONE' NOT SET/ |
| 4412 | 022544 | 042047 | 047117 | 023505 | | |
| 4413 | 022552 | 047040 | 052117 | 051440 | | |
| 4414 | 022560 | 052105 | 000 | | | |
| 4415 | 022563 | 106 | 046111 | 020114 | EM53: | .ASCIZ /FILL & EMPTY BUFF TEST... DATA MISCOMP/ |
| 4416 | 022570 | 020046 | 046505 | 052120 | | |
| 4417 | 022576 | 020131 | 052502 | 043106 | | |
| 4418 | 022604 | 052040 | 051505 | 027124 | | |
| 4419 | 022612 | 027056 | 042040 | 052101 | | |
| 4420 | 022620 | 020101 | 044515 | 041523 | | |
| 4421 | 022626 | 046517 | 000120 | | | |
| 4422 | 022632 | 040504 | 040524 | 046440 | EM54: | .ASCIZ /DATA MISCOMP/ |
| 4423 | 022640 | 051511 | 047503 | 050115 | | |
| 4424 | 022646 | 000 | | | | |
| 4425 | 022647 | 122 | 050105 | 040514 | EM55: | .ASCIZ /REPLACE DX0...10 BAD SECTORS/ |
| 4426 | 022654 | 042503 | 042040 | 030130 | | |
| 4427 | 022662 | 027056 | 030456 | 020060 | | |
| 4428 | 022670 | 040502 | 020104 | 042523 | | |
| 4429 | 022676 | 052103 | 051117 | 000123 | | |
| 4430 | 022704 | 042522 | 046120 | 041501 | EM56: | .ASCIZ /REPLACE DX1...10 BAD SECTORS/ |
| 4431 | 022712 | 020105 | 054104 | 027061 | | |
| 4432 | 022720 | 027056 | 030061 | 041040 | | |
| 4433 | 022726 | 042101 | 051440 | 041505 | | |
| 4434 | 022734 | 047524 | 051522 | 000 | | |
| 4435 | 022741 | 104 | 030130 | 044040 | EM57: | .ASCIZ /DX0 HARD CRC ERR/ |
| 4436 | 022746 | 051101 | 020104 | 051103 | | |
| 4437 | 022754 | 020103 | 051105 | 000122 | | |
| 4438 | 022762 | 054104 | 020060 | 040504 | EM58: | .ASCIZ /DX0 DATA CRC ERR/ |
| 4439 | 022770 | 040524 | 041440 | 041522 | | |
| 4440 | 022776 | 042440 | 051122 | 000 | | |
| 4441 | 023003 | 104 | 030130 | 050040 | EM59: | .ASCIZ /DX0 PREV CRC ERR WITH DATA OK/ |
| 4442 | 023010 | 042522 | 020126 | 051103 | | |
| 4443 | 023016 | 020103 | 051105 | 020122 | | |
| 4444 | 023024 | 044527 | 044124 | 042040 | | |
| 4445 | 023032 | 052101 | 020101 | 045517 | | |
| 4446 | 023040 | 000 | | | | |
| 4447 | 023041 | 104 | 030530 | 044040 | EM60: | .ASCIZ /DX1 HARD CRC ERR/ |
| 4448 | 023046 | 051101 | 020104 | 051103 | | |
| 4449 | 023054 | 020103 | 051105 | 000122 | | |
| 4450 | 023062 | 054104 | 020061 | 040504 | EM61: | .ASCIZ /DX1 DATA CRC ERR/ |
| 4451 | 023070 | 040524 | 041440 | 041522 | | |
| 4452 | 023076 | 042440 | 051122 | 000 | | |
| 4453 | 023103 | 104 | 030530 | 050040 | EM62: | .ASCIZ /DX1 PREV CRC ERR WITH DATA OK/ |
| 4454 | 023110 | 042522 | 020126 | 051103 | | |
| 4455 | 023116 | 020103 | 051105 | 020122 | | |
| 4456 | 023124 | 044527 | 044124 | 042040 | | |
| 4457 | 023132 | 052101 | 020101 | 045517 | | |
| 4458 | 023140 | 000 | | | | |
| 4459 | | | | | | |


```

4512
4513 023560 026050 001116 000000 DT1: .WORD ERRNUM,$ERRPC,0
4514 023566 026050 001116 004436 DT2: .WORD ERRNUM,$ERRPC,ADDR,0
4515 023574 000000
4516 023576 026050 001116 004436 DT3: .WORD ERRNUM,$ERRPC,ADDR,PAT,MEMHLD,0
4517 023604 004434 004432 000000
4518 023612 026050 001116 012144 DT4: .WORD ERRNUM,$ERRPC,SPRS,0
4519 023620 000000
4520 023622 026050 001116 012244 DT5: .WORD ERRNUM,$ERRPC,T1CHR,T1HLD,0
4521 023630 012246 000000
4522 023634 026050 001116 012344 DT6: .WORD ERRNUM,$ERRPC,T2CHR,T2HLD,0
4523 023642 012346 000000
4524 023646 026050 001116 012444 DT7: .WORD ERRNUM,$ERRPC,T3CHR,T3HLD,0
4525 023654 012446 000000
4526 023660 026050 001116 012544 DT8: .WORD ERRNUM,$ERRPC,COMCHR,COMHLD,0
4527 023666 012546 000000
4528 023672 026050 003120 001116 DT9: .WORD ERRNUM,DXTST,$ERRPC,SRXCS,SRXES,SRXSA,0
4529 023700 013050 013052 013054
4530 023706 000000
4531 023710 026050 003120 001116 DT10: .WORD ERRNUM,DXTST,$ERRPC,SRXCS,SRXES,DOTRK,DOSEC,DOERR,0
4532 023716 013050 013052 002406
4533 023724 002410 002404 000000
4534 023732 003120 001116 177170 DT11: .WORD DXTST,$ERRPC,RXCS,RXES,RXSA,DOEXP,DOREC,DOCERR,0
4535 023740 177172 177174 002416
4536 023746 002420 002414 000000
4537 023754 026050 003120 001116 DT12: .WORD ERRNUM,DXTST,$ERRPC,SRXCS,SRXES,D1TRK,D1SEC,D1ERR,0
4538 023762 013050 013052 002656
4539 023770 002660 002654 000000
4540 023776 003120 001116 177170 DT13: .WORD DXTST,$ERRPC,RXCS,RXES,RXSA,D1EXP,D1REC,D1CERR,0
4541 024004 177172 177174 002666
4542 024012 002670 002664 000000
4543 024020 026050 003120 001116 DT14: .WORD ERRNUM,DXTST,$ERRPC,RXCS,RXES,RXSA,0
4544 024026 177170 177172 177174
4545 024034 000000
4546 024036 026050 001116 006274 DT15: .WORD ERRNUM,$ERRPC,EXPO,EBHLD,0
4547 024044 006272 000000
4548 024050 026050 001116 006276 DT16: .WORD ERRNUM,$ERRPC,EXP1,EBHLD,0
4549 024056 006272 000000
4550
4551 024062 000000 DF: .WORD 0 ;OCTAL FORMAT FOR ALL
4552 024064 000000 .WORD 0
4553 024066 000000 .WORD 0
4554
4555 024070 000000 DF1: .WORD 0 ;OCTAL,OCTAL
4556 024072 000000 .WORD 0 ;OCTAL,OCTAL
4557 024074 000400 .WORD 400 ;DECIMAL,OCTAL
4558 024076 000401 .WORD 401 ;DECIMAL,DECIMAL
4559
4560 024100 000000 DF2: .WORD 0 ;OCTAL,OCTAL
4561 024102 000000 .WORD 0 ;OCTAL,OCTAL
4562 024104 000000 .WORD 0 ;OCTAL,OCTAL
4563 024106 000400 .WORD 400 ;DECIMAL,OCTAL
4564

```

```

4565          .SBTTL  TYPE ROUTINE
4566
4567          ;*****
4568          ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
4569          ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
4570          ;*NOTE1:          $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHAPACTR.
4571          ;*NOTE2:          $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
4572          ;*NOTE3:          $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
4573          ;*
4574          ;*CALL:
4575          ;*1) USING A TRAP INSTRUCTION
4576          ;*      TYPE      ,MESADR          ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
4577          ;*OR
4578          ;*      TYPE
4579          ;*      MESADR
4580          ;*
4581
4582 024110 105737 001157          $TYPE:  TSTB      $TPFLG          ;; IS THERE A TERMINAL?
4583 024114 100002                BPL          1$          ;; BR IF YES
4584 024116 000000                HALT                ;; HALT HERE IF NO TERMINAL
4585 024120 000430                BR          3$          ;; LEAVE
4586 024122 010046          1$:  MOV          R0,-(SP)        ;; SAVE R0
4587 024124 017600 000002        MOV          @2(SP),R0      ;; GET ADDRESS OF ASCIZ STRING
4588 024130 122737 000001 001210  CMPB      #APTENV,$ENV  ;; RUNNING IN APT MODE
4589 024136 001011                BNE          62$        ;; NO,GO CHECK FOR APT CONSOLE
4590 024140 132737 000100 001211  BITB      #APTPOOL,$ENVM ;; SPOOL MESSAGE TO APT
4591 024146 001405                BEQ          62$        ;; NO,GO CHECK FOR CONSOLE
4592 024150 010037 024160        MOV          R0,61$      ;; SETUP MESSAGE ADDRESS FOR APT
4593 024154 004737 025242        JSR          PC,$ATY3  ;; SPOOL MESSAGE TO APT
4594 024160 000000          61$:  .WORD          0          ;; MESSAGE ADDRESS
4595 024162 132737 000040 001211  62$:  BITB      #APTCSUP,$ENVM ;; APT CONSOLE SUPPRESSED
4596 024170 001003                BNE          60$        ;; YES,SKIP TYPE OUT
4597 024172 112046          2$:  MOVB      (R0)+,-(SP)    ;; PUSH CHARACTER TO BE TYPED ONTO STACK
4598 024174 001005                BNE          4$          ;; BR IF IT ISN'T THE TERMINATOR
4599 024176 005726                TST          (SP)+      ;; IF TERMINATOR POP IT OFF THE STACK
4600 024200 012600          60$:  MOV          (SP)+,R0      ;; RESTORE R0
4601 024202 062716 000002        3$:  ADD          #2,(SP)    ;; ADJUST RETURN PC
4602 024206 000002                RTI                    ;; RETURN
4603 024210 122716 000011        4$:  CMPB      #HT,(SP)    ;; BRANCH IF <HT>
4604 024214 001430                BEQ          8$          ;;
4605 024216 122716 000200        CMPB      #CRLF,(SP)  ;; BRANCH IF NOT <CRLF>
4606 024222 001006                BNE          5$          ;;
4607 024224 005726                TST          (SP)+      ;; POP <CR><LF> EQUIV
4608 024226 104401                TYPE                ;; TYPE A CR AND LF
4609 024230 001165                $CRLF
4610 024232 105037 024366        CLRB      $CHARCNT    ;; CLEAR CHARACTER COUNT
4611 024236 000755                BR          2$          ;; GET NEXT CHARACTER
4612 024240 004737 024322        5$:  JSR          PC,$TYPEC  ;; GO TYPE THIS CHARACTER
4613 024244 123726 001156        6$:  CMPB      $FILLC,(SP)+ ;; IS IT TIME FOR FILLER CHARS.?
4614 024250 001350                BNE          2$        ;; IF NO GO GET NEXT CHAR.
4615 024252 013746 001154        MOV          $NULL,-(SP) ;; GET # OF FILLER CHARS. NEEDED
4616                                ;; AND THE NULL CHAR.
4617 024256 105366 000001        7$:  DECB      1(SP)      ;; DOES A NULL NEED TO BE TYPED?
4618 024262 002770                BLT          6$        ;; BR IF NO--GO POP THE NULL OFF OF STACK

```

```

4619 024264 004737 024322          JSR    PC,$TYPEC      ;;GO TYPE A NULL
4620 024270 105337 024366          DECB  $CHARCNT       ;;DO NOT COUNT AS A COUNT
4621 024274 000770                   BR     7$            ;;LOOP
4622
4623                               ;HORIZONTAL TAB PROCESSOR
4624
4625 024276 112716 000040          8$:   MOVB  #' ,(SP)      ;;REPLACE TAB WITH SPACE
4626 024302 004737 024322          9$:   JSR    PC,$TYPEC      ;;TYPE A SPACE
4627 024306 132737 000007 024366  BITB  #7,$CHARCNT     ;;BRANCH IF NOT AT
4628 024314 001372                   BNE   9$            ;;TAB STOP
4629 024316 005726                   TST   (SP)+         ;;POP SPACE OFF STACK
4630 024320 000724                   BR    2$            ;;GET NEXT CHARACTER
4631 024322 105777 154622          $TYPEC: TSTB @ $TPS     ;;WAIT UNTIL PRINTER IS READY
4632 024326 100375                   BPL  $TYPEC
4633 024330 116677 000002 154614  MOVB  2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
4634 024336 122766 000015 000002  CMPB  #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
4635 024344 001003                   BNE  1$            ;;BRANCH IF NO
4636 024346 105037 024366          CLRB  $CHARCNT       ;;YES--CLEAR CHARACTER COUNT
4637 024352 000406                   BR    $TYPEX        ;;EXIT
4638 024354 122766 000012 000002  1$:   CMPB  #LF,2(SP)   ;;IS CHARACTER A LINE FEED?
4639 024362 001402                   BEQ  $TYPEX        ;;BRANCH IF YES
4640 024364 105227                   INCB (PC)+         ;;COUNT THE CHARACTER
4641 024366 000000          $CHARCNT: .WORD 0    ;;CHARACTER COUNT STORAGE
4642 024370 000207          $TYPEX: RTS    PC
4643

```

```

4644      .SBTTL  TTY INPUT ROUTINE
4645
4646      ;:*****
4647      .ENABL  LSB
4648
4649      ;:*****
4650      ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
4651      ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
4652      ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
4653      ;*WHEN OPERATING IN TTY FLAG MODE.
4654      024372 022737 000176 001140 $CKSWR: CMP      #SWREG,SWR      ;; IS THE SOFT-SWR SELECTED?
4655      024400 001114          BNE      15$          ;; BRANCH IF NO
4656      024402 105777 154536          TSTB   @STKS          ;; CHAR THERE?
4657      024406 100111          BPL     15$          ;; IF NO, DON'T WAIT AROUND
4658      024410 117746 154532          MOVB  @STKB,-(SP)     ;; SAVE THE CHAR
4659      024414 042716 177600          BIC   #^C177,(SP)   ;; STRIP-OFF THE ASCII
4660      024420 022726 000007          CMP   #7,(SP)+      ;; IS IT A CONTROL G?
4661      024424 001102          BNE   15$          ;; NO, RETURN TO USER
4662      024426 123727 001134 000001  CMPB  $AUTOB,#1     ;; ARE WE RUNNING IN AUTO-MODE?
4663      024434 001476          BEQ   15$          ;; BRANCH IF YES
4664
4665      024436 104401 025204          $GTSWR: TYPE   , $CNTLG      ;; ECHO THE CONTROL-G (^G)
4666      024442 104401 025211          TYPE   , $MSWR        ;; TYPE CURRENT CONTENTS
4667      024446 013746 000176          MOV   SWREG,-(SP)    ;; SAVE SWREG FOR TYPEOUT
4668      024452 104402          TYPOC          ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
4669      024454 104401 025222          TYPE   , $MNEW       ;; PROMPT FOR NEW SWR
4670      024460 005046          19$:  CLR   -(SP)      ;; CLEAR COUNTER
4671      024462 005046          CLR   -(SP)      ;; THE NEW SWR
4672      024464 105777 154454          7$:  TSTB  @STKS     ;; CHAR THERE?
4673      024470 100375          BPL   7$         ;; IF NOT TRY AGAIN
4674
4675      024472 117746 154450          MOVB  @STKB,-(SP)   ;; PICK UP CHAR
4676      024476 042716 177600          BIC   #^C177,(SP)  ;; MAKE IT 7-BIT ASCII
4677
4678      024502 021627 000003          CMP   (SP),#3      ;; IS IT A CONTROL-C?
4679      024506 001015          BNE   9$         ;; BRANCH IF NOT
4680      024510 104401 025172          TYPE   , $CNTLC     ;; YES, ECHO CONTROL-C (^C)
4681      024514 062706 000006          ADD   #6,SP        ;; CLEAN UP STACK
4682      024520 123727 001135 000001  CMPB  $INTAG,#1     ;; REENABLE TTY KEYBOARD INTERRUPTS?
4683      024526 001003          BNE   8$         ;; BRANCH IF NO
4684      024530 012777 000100 154406  MOV   #100,@STKS   ;; ALLOW TTY KEYBOARD INTERRUPTS
4685      024536 000137 010760          8$:  JMP   GT$DEV    ;; CONTROL-C RESTART
4686
4687
4688      024542 021627 000025          9$:  CMP   (SP),#25   ;; IS IT A CONTROL-U?
4689      024546 001005          BNE   10$        ;; BRANCH IF NOT
4690      024550 104401 025177          TYPE   , $CNTLU     ;; YES, ECHO CONTROL-U (^U)
4691      024554 062706 000006          20$: ADD   #6,SP        ;; IGNORE PREVIOUS INPUT
4692      024560 000737          BR    19$        ;; LET'S TRY IT AGAIN
4693
4694
4695      024562 021627 000015          10$: CMP   (SP),#15   ;; IS IT A <CR>?
4696      024566 001022          BNE   16$        ;; BRANCH IF NO
4697      024570 005766 000004          TST   4(SP)       ;; YES, IS IT THE FIRST CHAR?

```

```

4698 024574 001403      BEQ      11$      ;;BRANCH IF YES
4699 024576 016677 000002 154334    MOV      2(SP),@SWR  ;;SAVE NEW SWR
4700 024604 062706 000006      11$: ADD      #6,SP      ;;CLEAR UP STACK
4701 024610 104401 001165      14$: TYPE     $CRLF   ;;ECHO <CR> AND <LF>
4702 024614 123727 001135 000001    CMPB    $INTAG,#1   ;;RE-ENABLE TTY KBD INTERRUPTS?
4703 024622 001003      BNE      15$      ;;BRANCH IF NOT
4704 024624 012777 000100 154312    MOV      #100,@$TKS ;;RE-ENABLE TTY KBD INTERRUPTS
4705 024632 000002      RTI      15$:      ;;RETURN
4706 024634 004737 024322      16$: JSR      PC,$TYPEC ;;ECHO CHAR
4707 024640 021627 000060      CMP      (SP),#60   ;;CHAR < 0?
4708 024644 002420      BLT      18$      ;;BRANCH IF YES
4709 024646 021627 000067      CMP      (SP),#67   ;;CHAR > 7?
4710 024652 003015      BGT      18$      ;;BRANCH IF YES
4711 024654 042726 000060      BIC      #60,(SP)+  ;;STRIP-OFF ASCII
4712 024660 005766 000002      TST      2(SP)      ;;IS THIS THE FIRST CHAR
4713 024664 001403      BEQ      17$      ;;BRANCH IF YES
4714 024666 006316      ASL      (SP)      ;;NO, SHIFT PRESENT
4715 024670 006316      ASL      (SP)      ;;CHAR OVER TO MAKE
4716 024672 006316      ASL      (SP)      ;;ROOM FOR NEW ONE.
4717 024674 005266 000002      17$: INC      2(SP)   ;;KEEP COUNT OF CHAR
4718 024700 056616 177776      BIS      -2(SP),(SP) ;;SET IN NEW CHAR
4719 024704 000667      BR       7$        ;;GET THE NEXT ONE
4720 024706 104401 001164      18$: TYPE     $QUES   ;;TYPE ?<CR><LF>
4721 024712 000720      BR       20$      ;;SIMULATE CONTROL-U
4722      .DSABL  LSB
4723
4724
4725      ;:*****
4726      ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
4727      ;*CALL:
4728      ;*      RDCHR      ;;INPUT A SINGLE CHARACTER FROM THE TTY
4729      ;*      RETURN HERE ;;CHARACTER IS ON THE STACK
4730      ;*                ;;WITH PARITY BIT STRIPPED OFF
4731      ;*
4732      ;*
4733 024714 011646      $RDCHR: MOV      (SP),-(SP) ;;PUSH DOWN THE PC
4734 024716 016666 000004 000002    MOV      4(SP),2(SP) ;;SAVE THE PS
4735 024724 105777 154214      1$: TSTB    @$TKS     ;;WAIT FOR
4736 024730 100375      BPL      1$        ;;A CHARACTER
4737 024732 117766 154210 000004    MOVB    @$TKB,4(SP) ;;READ THE TTY
4738 024740 042766 177600 000004    BIC     #^C<177>,4(SP) ;;GET RID OF JUNK IF ANY
4739 024746 026627 000004 000023    CMP     4(SP),#23   ;;IS IT A CONTROL-S?
4740 024754 001013      BNE      3$        ;;BRANCH IF NO
4741 024756 105777 154162      2$: TSTB    @$TKS     ;;WAIT FOR A CHARACTER
4742 024762 100375      BPL      2$        ;;LOOP UNTIL ITS THERE
4743 024764 117746 154156      MOVB    @$TKB,-(SP) ;;GET CHARACTER
4744 024770 042716 177600      BIC     #^C177,(SP) ;;MAKE IT 7-BIT ASCII!
4745 024774 022627 000021      CMP     (SP)+,#21   ;;IS IT A CONTROL-Q?
4746 025000 001366      BNE      2$        ;;IF NOT DISCARD IT
4747 025002 000750      BR      1$        ;;YES, RESUME
4748 025004 026627 000004 000140      3$: CMP     4(SP),#140 ;;IS IT UPPER CASE?
4749 025012 002407      BLT     4$        ;;BRANCH IF YES
4750 025014 026627 000004 000175    CMP     4(SP),#175  ;;IS IT A SPECIAL CHAR?
4751 025022 003003      BGT     4$        ;;BRANCH IF YES

```

```

4752 025024 042766 000040 000004      BIC      #40,4(SP)      ;;MAKE IT UPPER CASE
4753 025032 000002                    4$:      RTI              ;;GO BACK TO USER
4754                                     ;;*****
4755                                     ;;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
4756                                     ;;*CALL:
4757                                     ;;*      RDLIN              ;;INPUT A STRING FROM THE TTY
4758                                     ;;*      RETURN HERE      ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
4759                                     ;;*                          ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
4760
4761 025034 010346                    $RDLIN: MOV      R3,-(SP)      ;;SAVE R3
4762 025036 012703 025162              1$:      MOV      #$TTYIN,R3    ;;GET ADDRESS
4763 025042 022703 025172              2$:      CMP      #$TTYIN+8.,R3    ;;BUFFER FULL?
4764 025046 101415                    BLOS     4$              ;;BR IF YES
4765 025050 104410                    RDCHR   ;;GO READ ONE CHARACTER FROM THE TTY
4766 025052 112613                    MOVB    (SP)+,(R3)        ;;GET CHARACTER
4767 025054 122713 000003              CMPB    #3,(R3)          ;;IS IT A CONTROL-C?
4768 025060 001005                    BNE     10$              ;;BRANCH IF NO
4769 025062 104401 025172              TYPE   , $CNTLC         ;;TYPE A CONTROL-C (^C)
4770 025066 012603                    MOV     (SP)+,R3         ;;RESTORE R3
4771 025070 000137 010760              JMP     GT$DEV           ;;GOTO CONTROL-C RESTART
4772 025074 122713 000177              10$:    CMPB    #177,(R3)     ;;IS IT A RUBOUT
4773 025100 001003                    BNE     3$              ;;SKIP IF NOT
4774 025102 104401 001164              4$:     TYPE   , $QUES     ;;TYPE A '?'
4775 025106 000753                    BR      1$              ;;CLEAR THE BUFFER AND LOOP
4776 025110 111337 025160              3$:     MOVB    (R3),9$     ;;ECHO THE CHARACTER
4777 025114 104401 025160              TYPE   ,9$
4778 025120 122723 000015              CMPB    #15,(R3)+      ;;CHECK FOR RETURN
4779 025124 001346                    BNE     2$              ;;LOOP IF NOT RETURN
4780 025126 105063 177777              CLRB   -1(R3)          ;;CLEAR RETURN (THE 15)
4781 025132 104401 001166              TYPE   , $LF           ;;TYPE A LINE FEED
4782 025136 012603                    MOV     (SP)+,R3         ;;RESTORE R3
4783 025140 011646                    MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
4784 025142 016666 000004 000002      MOV     4(SP),2(SP)    ;;FIRST ASCII CHARACTER ON IT
4785 025150 012766 025162 000004      MOV     #$TTYIN,4(SP)
4786 025156 000002                    RTI                    ;;RETURN
4787 025160 000                    9$:     .BYTE   0           ;;STORAGE FOR ASCII CHAR. TO TYPE
4788 025161 000                    .BYTE   0           ;;TERMINATOR
4789 025162 000010                    $TTYIN: .BLKB  8.     ;;RESERVE 8 BYTES FOR TTY INPUT
4790 025172 041536 005015 000          $CNTLC: .ASCIZ  /^C/<15><12>  ;;CONTROL 'C'
4791 025177 0136 006525 000012        $CNTLU: .ASCIZ  /^U/<15><12>  ;;CONTROL 'U'
4792 025204 043536 005015 000          $CNTLG: .ASCIZ  /^G/<15><12>  ;;CONTROL 'G'
4793 025211 015 051412 051127        $MSWR:  .ASCIZ  <15><12>/SWR = /
4794 025216 036440 000040
4795 025222 020040 042516 020127        $MNEW:  .ASCIZ  / NEW = /
4796 025230 020075 000
4797 025234 .EVEN

```



```

4798          .SBTTL  APT COMMUNICATIONS ROUTINE
4799
4800          /:*****
4801 025234 112737 000001 025500 $ATY1:  MOVB  #1,$FFLG      ;;TO REPORT FATAL ERROR
4802 025242 112737 000001 025476 $ATY3:  MOVB  #1,$MFLG      ;;TO TYPE A MESSAGE
4803 025250 000403                BR      $ATYC
4804 025252 112737 000001 025500 $ATY4:  MOVB  #1,$FFLG      ;;TO ONLY REPORT FATAL ERROR
4805 025260                $ATYC:
4806 025260 010046                MOV    R0,-(SP)      ;;PUSH R0 ON STACK
4807 025262 010146                MOV    R1,-(SP)      ;;PUSH R1 ON STACK
4808 025264 105737 025476                TSTB  $MFLG          ;;SHOULD TYPE A MESSAGE?
4809 025270 001450                BEQ    5$            ;;IF NOT: BR
4810 025272 122737 000001 001210        CMPB  #APTENV,$ENV   ;;OPERATING UNDER APT?
4811 025300 001031                BNE    3$            ;;IF NOT: BR
4812 025302 132737 000100 001211        BITB  #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
4813 025310 001425                BEQ    3$            ;;IF NOT: BR
4814 025312 017600 000004                MOV    @4(SP),R0     ;;GET MESSAGE ADDR.
4815 025316 062766 000002 000004        ADD    #2,4(SP)      ;;BUMP RETURN ADDR.
4816 025324 005737 001170                1$:  TST    $MSGTYPE   ;;SEE IF DONE W/ LAST XMISSION?
4817 025330 001375                BNE    1$            ;;IF NOT: WAIT
4818 025332 010037 001204                MOV    R0,$MSGAD     ;;PUT ADDR IN MAILBOX
4819 025336 105720                2$:  TSTB  (R0)+        ;;FIND END OF MESSAGE
4820 025340 001376                BNE    2$
4821 025342 163700 001204                SUB    $MSGAD,R0     ;;SUB START OF MESSAGE
4822 025346 006200                ASR    R0            ;;GET MESSAGE LNTH IN WORDS
4823 025350 010037 001206                MOV    R0,$MSGGLT    ;;PUT LENGTH IN MAILBOX
4824 025354 012737 000004 001170        MOV    #4,$MSGTYPE   ;;TELL APT TO TAKE MSG.
4825 025362 000413                BR     5$
4826 025364 017637 000004 025410        3$:  MOV    @4(SP),4$    ;;PUT MSG ADDR IN JSR LINKAGE
4827 025372 062766 000002 000004        ADD    #2,4(SP)      ;;BUMP RETURN ADDRESS
4828 025400 013746 177776                MOV    177776,-(SP)  ;;PUSH 177776 ON STACK
4829 025404 004737 024110                JSR    PC,$TYPE      ;;CALL TYPE MACRO
4830 025410 000000                4$:  .WORD  0
4831 025412                5$:
4832 025412 105737 025500                10$: TSTB  $FFLG          ;;SHOULD REPORT FATAL ERROR?
4833 025416 001416                BEQ    12$          ;;IF NOT: BR
4834 025420 005737 001210                TST    $ENV          ;;RUNNING UNDER APT?
4835 025424 001413                BEQ    12$          ;;IF NOT: BR
4836 025426 005737 001170                11$: TST    $MSGTYPE     ;;FINISHED LAST MESSAGE?
4837 025432 001375                BNE    11$          ;;IF NOT: WAIT
4838 025434 017637 000004 001172        MOV    @4(SP),$FATAL ;;GET ERROR #
4839 025442 062766 000002 000004        ADD    #2,4(SP)      ;;BUMP RETURN ADDR.
4840 025450 005237 001170                INC    $MSGTYPE     ;;TELL APT TO TAKE ERROR
4841 025454 105037 025500                12$: CLRB  $FFLG          ;;CLEAR FATAL FLAG
4842 025460 105037 025477                CLRB  $LFLG          ;;CLEAR LOG FLAG
4843 025464 105037 025476                CLRB  $MFLG          ;;CLEAR MESSAGE FLAG
4844 025470 012601                MOV    (SP)+,R1     ;;POP STACK INTO R1
4845 025472 012600                MOV    (SP)+,R0     ;;POP STACK INTO R0
4846 025474 000207                RTS    PC            ;;RETURN
4847 025476          000                $MFLG: .BYTE 0      ;;MESSG. FLAG
4848 025477          000                $LFLG: .BYTE 0      ;;LOG FLAG
4849 025500          000                $FFLG: .BYTE 0      ;;FATAL FLAG
4850                025502                .EVEN
4851                000200                APTSIZE=200

```

| | | |
|------|--------|--------------|
| 4852 | 000001 | APTENV=001 |
| 4853 | 000100 | APTSPool=100 |
| 4854 | 000040 | APTCSUP=040 |

```

4855      .SBTTL  ERROR HANDLER ROUTINE
4856
4857      ;*****
4858      ;THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
4859      ;SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
4860      ;AND GO TO MYTYPE ON ERROR
4861      ;THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
4862      ;*SW15=1      HALT ON ERROR
4863      ;*SW13=1      INHIBIT ERROR TYPEOUTS
4864      ;*SW10=1     BELL ON ERROR
4865      ;*CALL
4866      ;*      ERROR  N      ;;ERROR=EMT AND N=ERROR ITEM NUMBER
4867
4868      $ERROR:
4869      025502 104407
4870      025504 105237 001103
4871      025510 001775
4872      025512 013777 001102 153422
4873      025520 032777 002000 153412
4874      025526 001402
4875      025530 104401 001160
4876      025534 005237 001112
4877      025540 011637 001116
4878      025544 162737 000002 001116
4879      025552 117737 153340 001114
4880      025560 032777 020000 153352
4881      025566 001004
4882      025570 004737 026016
4883      025574 104401 001165
4884      025600
4885      025600 122737 000001 001210
4886      025606 001007
4887      025610 113737 001114 025622
4888      025616 004737 025252
4889      025622 000
4890      025623 000
4891      025624 000777
4892      025626 005777 153306
4893      025632 100002
4894      025634 000000
4895      025636 104407
4896      025640
4897      025640 000002

      7$:      CKSWR      ;;TEST FOR CHANGE IN SOFT-SWR
      INCB      $ERFLG    ;;SET THE ERROR FLAG
      BEQ      7$        ;;DON'T LET THE FLAG GO TO ZERO
      MOV      $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
      BIT      #BIT10,@SWR ;;BELL ON ERROR?
      BEQ      1$        ;;NO - SKIP
      TYPE     ,$BELL    ;;RING BELL
      1$:      INC      $ERTTL ;;COUNT THE NUMBER OF ERRORS
      MOV      (SP),$ERRPC ;;GET ADDRESS OF ERROR INSTRUCTION
      SUB      #2,$ERRPC
      MOVB     @ $ERRPC,$ITEMB ;;STRIP AND SAVE THE ERROR ITEM CODE
      BIT      #BIT13,@SWR ;;SKIP TYPEOUT IF SET
      BNE      20$      ;;SKIP TYPEOUTS
      JSR      PC,MYTYPE ;;GO TO USER ERROR ROUTINE
      TYPE     ,$CRLF

      20$:     CMPB     #APTENV,$ENV ;;RUNNING IN APT MODE
      BNE      2$      ;;NO,SKIP APT ERROR REPORT
      MOVB     $ITEMB,21$ ;;SET ITEM NUMBER AS ERROR NUMBER
      JSR      PC,$ATY4 ;;REPORT FATAL ERROR TO APT

      21$:     .BYTE   0
      .BYTE   0

      22$:     BR      22$ ;;APT ERROR LOOP
      2$:      TST     @SWR ;;HALT ON ERROR
      BPL     3$      ;;SKIP IF CONTINUE
      HALT    ;;HALT ON ERROR!
      CKSWR   ;;TEST FOR CHANGE IN SOFT-SWR

      3$:      RTI     ;;RETURN

```

```

4898          .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE
4899
4900          ::*****
4901          ::*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
4902          ::*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
4903          ::*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
4904
4905          $ERRTYP:
4906          025642 104401 001165          TYPE      , $CRLF          ::'CARRIAGE RETURN' & 'LINE FEED'
4907          025646 010046          MOV      R0,-(SP)          ::SAVE R0
4908          025650 005000          CLR      R0              ::PICKUP THE ITEM INDEX
4909          025652 153700 001114          BISB    @($ITEMB),R0
4910          025656 001004          BNE     1$              ::IF ITEM NUMBER IS ZERO, JUST
4911                                     ::TYPE THE PC OF THE ERROR
4912          025660 013746 001116          MOV     $ERRPC,-(SP)    ::SAVE $ERRPC FOR TYPEOUT
4913                                     ::ERROR ADDRESS
4914          025664 104402          TYPDC   10$           ::GO TYPE--OCTAL ASCII(ALL DIGITS)
4915          025666 000445          BR      10$           ::GET OUT
4916          025670 005300          1$:    DEC     R0              ::ADJUST THE INDEX SO THAT IT WILL
4917          025672 006300          ASL     R0              ::
4918          025674 006300          ASL     R0              ::
4919          025676 006300          ASL     R0              ::
4920          025700 062700 001250          ADD     #$ERRTB,R0      ::FORM TABLE POINTER
4921          025704 012037 025714          MOV     (R0)+,2$       ::PICKUP "ERROR MESSAGE" POINTER
4922          025710 001404          BEQ     3$              ::SKIP TYPEOUT IF NO POINTER
4923          025712 104401          TYPE   'ERROR MESSAGE'
4924          025714 000000          2$:    .WORD   0          ::"ERROR MESSAGE" POINTER GOES HERE
4925          025716 104401 001165          TYPE   , $CRLF          ::'CARRIAGE RETURN' & 'LINE FEED'
4926          025722 012037 025732          3$:    MOV     (R0)+,4$       ::PICKUP "DATA HEADER" POINTER
4927          025726 001404          BEQ     5$              ::SKIP TYPEOUT IF 0
4928          025730 104401          TYPE   'DATA HEADER'
4929          025732 000000          4$:    .WORD   0          ::"DATA HEADER" POINTER GOES HERE
4930          025734 104401 001165          TYPE   , $CRLF          ::'CARRIAGE RETURN' & 'LINE FEED'
4931          025740 010146          5$:    MOV     R1,-(SP)       ::SAVE R1
4932          025742 012001          MOV     (R0)+,R1       ::PICKUP "DATA TABLE" POINTER
4933          025744 001415          BEQ     9$              ::BR IF NO DATA TO BE TYPED
4934          025746 012000          MOV     (R0)+,R0       ::PICKUP "DATA FORMAT" POINTER
4935          025750 105720          6$:    TSTB   (R0)+         ::'OCTAL' OR 'DECIMAL'
4936          025752 001003          BNE     7$              ::BR IF DECIMAL
4937          025754 013146          MOV     @ (R1)+,-(SP)  ::SAVE @ (R1)+ FOR TYPEOUT
4938          025756 104402          TYPDC   8$           ::GO TYPE--OCTAL ASCII(ALL DIGITS)
4939          025760 000402          BR      8$
4940          025762          7$:
4941          025762 013146          MOV     @ (R1)+,-(SP)  ::SAVE @ (R1)+ FOR TYPEOUT
4942          025764 104405          TYPDC   9$           ::GO TYPE--DECIMAL ASCII WITH SIGN
4943          025766 005711          8$:    TST     (R1)          ::IS THERE ANOTHER NUMBER?
4944          025770 001403          BEQ     9$              ::BR IF NO
4945          025772 104401 026012          TYPE   ,11$           ::TYPE TWO(2) SPACES
4946          025776 000764          BR      6$              ::LOOP
4947
4948          026000 012601          9$:    MOV     (SP)+,R1     ::RESTORE R1
4949          026002 012600          10$:   MOV     (SP)+,R0     ::RESTORE R0
4950          026004 104401 001165          TYPE   , $CRLF          ::'CARRIAGE RETURN' & 'LINE FEED'
4951          026010 000207          RTS     PC              ::RETURN

```

```

4952 026012 020040 000 11$: .ASCIZ / / ;;TWO(2) SPACES
4953 026016 .EVEN
4954
4955
4956
4957 026016 113737 001114 026050 MYTYPE: MOVB $ITEMB,ERRNUM ;FOR ERROR TYPEOUT
4958 026024 013737 013054 001174 MOV SRXSA,$TESTN ;FOR APT
4959 ;APT PRINTS $TESTN & $FATAL IN THAT ORDER.
4960 ;$TESTN WILL BE FLOPPY RXSA
4961 ;$FATAL IS ERR #
4962 026032 004737 025642 JSR PC,$ERRTYP ;TYPE ERR MSG
4963 026036 005237 026052 INC TERR
4964 026042 005237 026056 INC PERR
4965 026046 000207 RTS PC
4966
4967 026050 000000 ERRNUM: 0 ;FOR ERR TYPEOUT
4968
4969 026052 000000 TERR: 0 ;TOTAL ERROR COUNT
4970 026054 000000 TSERR: 0 ;TOTAL SOFT ERR COUNT
4971 026056 000000 PERR: 0 ;PASS ERR COUNT
4972 026060 000000 PSERR: 0 ;PASS SOFT ERR COUNT... ALL FOR EOP TYPEOUT
  
```

```

4973          .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
4974
4975          ::*****
4976          ::*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
4977          ::*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
4978          ::*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
4979          ::*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
4980          ::*REPLACED WITH SPACES.
4981          ::*CALL:
4982          ::*      MOV      NUM,-(SP)          ;;PUT THE BINARY NUMBER ON THE STACK
4983          ::*      TYPDS          ;;GO TO THE ROUTINE
4984
4985          $TYPDS:
4986          026062 010046          MOV      R0,-(SP)          ;;PUSH R0 ON STACK
4987          026064 010146          MOV      R1,-(SP)          ;;PUSH R1 ON STACK
4988          026066 010246          MOV      R2,-(SP)          ;;PUSH R2 ON STACK
4989          026070 010346          MOV      R3,-(SP)          ;;PUSH R3 ON STACK
4990          026072 010546          MOV      R5,-(SP)          ;;PUSH R5 ON STACK
4991          026074 012746 020200  MOV      #20200,-(SP)      ;;SET BLANK SWITCH AND SIGN
4992          026100 016605 000020  MOV      20(SP),R5        ;;GET THE INPUT NUMBER
4993          026104 100004          BPL      1$                ;;BR IF INPUT IS POS.
4994          026106 005405          NEG      R5                ;;MAKE THE BINARY NUMBER POS.
4995          026110 112766 000055 000001  MOVVB   #'-,1(SP)        ;;MAKE THE ASCII NUMBER NEG.
4996          026116 005000          1$:     CLR      R0                ;;ZERO THE CONSTANTS INDEX
4997          026120 012703 026276  MOV      #$DBLK,R3        ;;SETUP THE OUTPUT POINTER
4998          026124 112723 000040  MOVVB   #' ,(R3)+        ;;SET THE FIRST CHARACTER TO A BLANK
4999          026130 005002          2$:     CLR      R2                ;;CLEAR THE BCD NUMBER
5000          026132 016001 026266  MOV      $DTBL(R0),R1     ;;GET THE CONSTANT
5001          026136 160105          3$:     SUB      R1,R5                ;;FORM THIS BCD DIGIT
5002          026140 002402          BLT     4$                ;;BR IF DONE
5003          026142 005202          INC     R2                ;;INCREASE THE BCD DIGIT BY 1
5004          026144 000774          BR     3$
5005          026146 060105          4$:     ADD     R1,R5                ;;ADD BACK THE CONSTANT
5006          026150 005702          TST    R2                ;;CHECK IF BCD DIGIT=0
5007          026152 001002          BNE    5$                ;;FALL THROUGH IF 0
5008          026154 105716          TSTB   (SP)              ;;STILL DOING LEADING 0'S?
5009          026156 100407          BMI    7$                ;;BR IF YES
5010          026160 106316          5$:     ASLB   (SP)              ;;MSD?
5011          026162 103003          BCC    6$                ;;BR IF NO
5012          026164 116663 000001 177777  MOVVB   1(SP),-1(R3)      ;;YES--SET THE SIGN
5013          026172 052702 000060  6$:     BIS    #'0,R2          ;;MAKE THE BCD DIGIT ASCII
5014          026176 052702 000040  7$:     BIS    #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
5015          026202 110223          MOVVB   R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
5016          026204 005720          TST    (R0)+            ;;JUST INCREMENTING
5017          026206 020027 000010  CMP     R0,#10          ;;CHECK THE TABLE INDEX
5018          026212 002746          BLT    2$                ;;GO DO THE NEXT DIGIT
5019          026214 003002          BGT    8$                ;;GO TO EXIT
5020          026216 010502          MOV    R5,R2            ;;GET THE LSD
5021          026220 000764          BR     6$                ;;GO CHANGE TO ASCII
5022          026222 105726          8$:     TSTB   (SP)+        ;;WAS THE LSD THE FIRST NON-ZERO?
5023          026224 100003          BPL    9$                ;;BR IF NO
5024          026226 116663 177777 177776  MOVVB   -1(SP),-2(R3)    ;;YES--SET THE SIGN FOR TYPING
5025          026234 105013          9$:     CLRB   (R3)          ;;SET THE TERMINATOR
5026          026236 012605          MOV    (SP)+,R5        ;;POP STACK INTO R5

```

```
5027 026240 012603      MOV      (SP)+,R3      ;;POP STACK INTO R3
5028 026242 012602      MOV      (SP)+,R2      ;;POP STACK INTO R2
5029 026244 012601      MOV      (SP)+,R1      ;;POP STACK INTO R1
5030 026246 012600      MOV      (SP)+,R0      ;;POP STACK INTO R0
5031 026250 104401 026276  TYPE      ,SDBLK      ;;NOW TYPE THE NUMBER
5032 026254 016666 000002 000004  MOV      2(SP),4(SP)    ;;ADJUST THE STACK
5033 026262 012616      MOV      (SP)+,(SP)
5034 026264 000002      RTI                          ;;RETURN TO USER
5035 026266 023420      $DTBL: 10000.
5036 026270 001750      1000.
5037 026272 000144      100.
5038 026274 000012      10.
5039 026276 000004      $DBLK: .BLKW 4
```

```

5040 .SBTTL BINARY TO OCTAL (ASCII) AND TYPE
5041
5042 ::*****
5043 ::*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
5044 ::*OCTAL (ASCII) NUMBER AND TYPE IT.
5045 ::*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
5046 ::*CALL:
5047 ::*      MOV      NUM,-(SP)      ::NUMBER TO BE TYPED
5048 ::*      TYPOS      ::CALL FOR TYPEOUT
5049 ::*      .BYTE  N      ::N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
5050 ::*      .BYTE  M      ::M=1 OR 0
5051 ::*                               ::1=TYPE LEADING ZEROS
5052 ::*                               ::0=SUPPRESS LEADING ZEROS
5053
5054 ::*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
5055 ::*$TYPOS OR $TYPOC
5056 ::*CALL:
5057 ::*      MOV      NUM,-(SP)      ::NUMBER TO BE TYPED
5058 ::*      TYPON      ::CALL FOR TYPEOUT
5059
5060 ::*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
5061 ::*CALL:
5062 ::*      MOV      NUM,-(SP)      ::NUMBER TO BE TYPED
5063 ::*      TYPOC      ::CALL FOR TYPEOUT
5064
5065 026306 017646 000000 $TYPOS: MOV @ (SP),-(SP) ::PICKUP THE MODE
5066 026312 116637 000001 026531 MOVB 1(SP), $OFILL ::LOAD ZERO FILL SWITCH
5067 026320 112637 026533 MOVB (SP)+, $OMODE+1 ::NUMBER OF DIGITS TO TYPE
5068 026324 062716 000002 ADD #2, (SP) ::ADJUST RETURN ADDRESS
5069 026330 000406 BR $TYPON
5070 026332 112737 000001 026531 $TYPOC: MOVB #1, $OFILL ::SET THE ZERO FILL SWITCH
5071 026340 112737 000006 026533 MOVB #6, $OMODE+1 ::SET FOR SIX(6) DIGITS
5072 026346 112737 000005 026530 $TYPON: MOVB #5, $OCNT ::SET THE ITERATION COUNT
5073 026354 010346 MOV R3, -(SP) ::SAVE R3
5074 026356 010446 MOV R4, -(SP) ::SAVE R4
5075 026360 010546 MOV R5, -(SP) ::SAVE R5
5076 026362 113704 026533 MOVB $OMODE+1, R4 ::GET THE NUMBER OF DIGITS TO TYPE
5077 026366 005404 NEG R4
5078 026370 062704 000006 ADD #6, R4 ::SUBTRACT IT FOR MAX. ALLOWED
5079 026374 110437 026532 MOVB R4, $OMODE ::SAVE IT FOR USE
5080 026400 113704 026531 MOVB $OFILL, R4 ::GET THE ZERO FILL SWITCH
5081 026404 016605 000012 MOV 12(SP), R5 ::PICKUP THE INPUT NUMBER
5082 026410 005003 CLR R3 ::CLEAR THE OUTPUT WORD
5083 026412 006105 1$: ROL R5 ::ROTATE MSB INTO 'C'
5084 026414 000404 BR 3$ ::GO DO MSB
5085 026416 006105 2$: ROL R5 ::FORM THIS DIGIT
5086 026420 006105 ROL R5
5087 026422 006105 ROL R5
5088 026424 010503 MOV R5, R3
5089 026426 006103 3$: ROL R3 ::GET LSB OF THIS DIGIT
5090 026430 105337 026532 DECB $OMODE ::TYPE THIS DIGIT?
5091 026434 100016 BPL 7$ ::BR IF NO
5092 026436 042703 177770 BIC #177770, R3 ::GET RID OF JUNK
5093 026442 001002 BNE 4$ ::TEST FOR 0

```


| | | | | | | |
|------|--------|--------|---------------|----------------|-------------|-----------------------------------|
| 5094 | 026444 | 005704 | | TST | R4 | ::SUPPRESS THIS 0? |
| 5095 | 026446 | 001403 | | BEQ | 5\$ | ::BR IF YES |
| 5096 | 026450 | 005204 | | 4\$: INC | R4 | ::DON'T SUPPRESS ANYMORE 0'S |
| 5097 | 026452 | 052703 | 000060 | BIS | #'0,R3 | ::MAKE THIS DIGIT ASCII |
| 5098 | 026456 | 052703 | 000040 | 5\$: BIS | #',R3 | ::MAKE ASCII IF NOT ALREADY |
| 5099 | 026462 | 110337 | 026526 | MOVB | R3,8\$ | ::SAVE FOR TYPING |
| 5100 | 026466 | 104401 | 026526 | TYPE | ,8\$ | ::GO TYPE THIS DIGIT |
| 5101 | 026472 | 105337 | 026530 | 7\$: DECB | \$OCNT | ::COUNT BY 1 |
| 5102 | 026476 | 003347 | | BGT | 2\$ | ::BR IF MORE TO DO |
| 5103 | 026500 | 002402 | | BLT | 6\$ | ::BR IF DONE |
| 5104 | 026502 | 005204 | | INC | R4 | ::INSURE LAST DIGIT ISN'T A BLANK |
| 5105 | 026504 | 000744 | | BR | 2\$ | ::GO DO THE LAST DIGIT |
| 5106 | 026506 | 012605 | | 6\$: MOV | (SP)+,R5 | ::RESTORE R5 |
| 5107 | 026510 | 012604 | | MOV | (SP)+,R4 | ::RESTORE R4 |
| 5108 | 026512 | 012603 | | MOV | (SP)+,R3 | ::RESTORE R3 |
| 5109 | 026514 | 016666 | 000002 000004 | MOV | 2(SP),4(SP) | ::SET THE STACK FOR RETURNING |
| 5110 | 026522 | 012616 | | MOV | (SP)+,(SP) | |
| 5111 | 026524 | 000002 | | RTI | | ::RETURN |
| 5112 | 026526 | 000 | | 8\$: .BYTE | 0 | ::STORAGE FOR ASCII DIGIT |
| 5113 | 026527 | 000 | | .BYTE | 0 | ::TERMINATOR FOR TYPE ROUTINE |
| 5114 | 026530 | 000 | | \$OCNT .BYTE | 0 | ::OCTAL DIGIT COUNTER |
| 5115 | 026531 | 000 | | \$OFILL: .BYTE | 0 | ::ZERO FILL SWITCH |
| 5116 | 026532 | 000000 | | \$OMODE: .WORD | 0 | ::NUMBER OF DIGITS TO TYPE |

5117
5118
5119
5120
5121
5122
5123
5124
5125 026534 010046
5126 026536 016600 000002
5127 026542 005740
5128 026544 111000
5129 026546 006300
5130 026550 016000 026570
5131 026554 000200
5132
5133
5134
5135
5136 026556 011646
5137 026560 016666 000004 000002
5138 026566 000002
5139
5140
5141
5142
5143
5144
5145
5146
5147 026570 026556
5148 026572 024110
5149 026574 026332
5150 026576 026306
5151 026600 026346
5152 026602 026062
5153
5154 026604 024442
5155
5156 026606 024372
5157 026610 024714
5158 026612 025034
5159 026614 010760
5160
5161 026616
5162
5163
5164 003174

.SBTTL TRAP DECODER

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

\$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
ASL R0 ;;POSITION FOR INDEXING
MOV \$TRPAD(R0),R0 ;;INDEX TO TABLE
RTS R0 ;;GO TO ROUTINE

;;THIS IS USE TO HANDLE THE 'GETPRI' MACRO

\$TRAP2: MOV (SP),-(SP) ;;MOVE THE PC DOWN
MOV 4(SP),2(SP) ;;MOVE THE PSW DOWN
RTI ;;RESTORE THE PSW

.SBTTL TRAP TABLE

*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
*BY THE 'TRAP' INSTRUCTION.

| | | | |
|----------|---------------|---------------|---|
| : | ROUTINE | | |
| : | ----- | | |
| \$TRPAD: | .WORD \$TRAP2 | | |
| | \$TYPE | ::CALL=TYPE | TRAP+1(104401) TTY TYPEOUT ROUTINE |
| | \$TYPOC | ::CALL=TYPOC | TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS) |
| | \$TYPOS | ::CALL=TYPOS | TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS) |
| | \$TYPON | ::CALL=TYPON | TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL) |
| | \$TYPDS | ::CALL=TYPDS | TRAP+5(104405) TYPE DECIMAL NUMBER (WITH SIGN) |
| | \$GTSWR | ::CALL=GTSWR | TRAP+6(104406) GET SOFT-SWR SETTING |
| | \$CKSWR | ::CALL=CKSWR | TRAP+7(104407) TEST FOR CHANGE IN SOFT-SWR |
| | \$RDCHR | ::CALL=RDCHR | TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE |
| | \$RDLIN | ::CALL=RDLIN | TRAP+11(104411) TTY TYPEIN STRING ROUTINE |
| | GT\$DEV | ::CALL=GTDEVM | TRAP+12(104412) ^C WILL OPEN UP \$DEV |

LSTAD: ;MAX ADDR FOR 8K WITH APT = 37400
;MAX ADDR FOR 8K NO APT BUT TO SAVE ABS LOADER = 37500
.END START

| | | | | | | | | | | | | | | | | | | | | | |
|----------------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|--|--|--|--|--|--|--|--|
| ORERR = 040000 | 943# | 979# | | | | | | | | | | | | | | | | | | | |
| PARAM = 177420 | 829# | 1787* | 1892* | 1894* | 1936* | 1938* | 1970* | 1972* | 2004* | 2006* | | | | | | | | | | | |
| PARERR= 010000 | 945# | 980# | | | | | | | | | | | | | | | | | | | |
| PAT 004434 | 1672* | 1676 | 1682 | 1700* | 1706* | 1709 | 1715 | 1726* | 1744# | 4516 | | | | | | | | | | | |
| PATT 017556 | 3471 | 3832 | 4118# | | | | | | | | | | | | | | | | | | |
| PC =%000007 | 717# | 1597* | 1610* | 1622* | 1627* | 1652* | 2149* | 2168* | 2218* | 2223* | 2234* | 2243* | 2248* | | | | | | | | |
| | 2259* | 2299* | 2321* | 2340* | 2357* | 2368* | 2384* | 2410* | 2419* | 2449* | 2596* | 2639* | 2651* | | | | | | | | |
| | 2679* | 2733* | 2758* | 2768* | 2786* | 2796* | 2807* | 2822* | 2851* | 2870* | 2918* | 3235* | 3312* | | | | | | | | |
| | 3609* | 3686* | 3877* | 3929* | 3976* | 4031* | 4593* | 4612* | 4619* | 4626* | 4640* | 4642* | 4706* | | | | | | | | |
| | 4829* | 4846* | 4882* | 4888* | 4951* | 4962* | 4965* | | | | | | | | | | | | | | |
| PERR 026056 | 1624* | 2491 | 4964* | 4971# | | | | | | | | | | | | | | | | | |
| PIRQ = 177772 | 703# | | | | | | | | | | | | | | | | | | | | |
| PIRQVE= 000240 | 797# | | | | | | | | | | | | | | | | | | | | |
| PORT 017061 | 1812 | 4055# | | | | | | | | | | | | | | | | | | | |
| PRB = 177516 | 834# | 2971* | | | | | | | | | | | | | | | | | | | |
| PRCHR 012140 | 1792* | 2971 | 2972 | 2974 | 2976 | 2978* | 2981* | 2983* | 2985* | 2990# | | | | | | | | | | | |
| PRINT 017050 | 1809 | 2586 | 2591 | 4053# | | | | | | | | | | | | | | | | | |
| PRPORT 012146 | 1783* | 1786* | 1803 | 1810 | 2993# | | | | | | | | | | | | | | | | |
| PRPRES 010610 | 1776 | 2590* | 2594* | 2600# | | | | | | | | | | | | | | | | | |
| PRS = 177514 | 833# | 1781 | 1794* | 1805* | 2588 | 2965 | | | | | | | | | | | | | | | |
| PRSRV 012014 | 1788 | 2965# | | | | | | | | | | | | | | | | | | | |
| PRT = 020000 | 877# | 1787 | | | | | | | | | | | | | | | | | | | |
| PRTST 004534 | 1753 | 1763 | 1766 | 1774# | | | | | | | | | | | | | | | | | |
| PRVEC = 000200 | 806# | 1788* | 1789* | | | | | | | | | | | | | | | | | | |
| PRO = 000000 | 720# | 1617 | | | | | | | | | | | | | | | | | | | |
| PR1 = 000040 | 721# | | | | | | | | | | | | | | | | | | | | |
| PR2 = 000100 | 722# | | | | | | | | | | | | | | | | | | | | |
| PR3 = 000140 | 723# | | | | | | | | | | | | | | | | | | | | |
| PR4 = 000200 | 724# | 1516 | | | | | | | | | | | | | | | | | | | |
| PR5 = 000240 | 725# | | | | | | | | | | | | | | | | | | | | |
| PR6 = 000300 | 726# | | | | | | | | | | | | | | | | | | | | |
| PR7 = 000340 | 727# | | | | | | | | | | | | | | | | | | | | |
| PS = 177776 | 700# | 701 | | | | | | | | | | | | | | | | | | | |
| PSERR 026060 | 1625* | 2494 | 3249* | 3334* | 3420* | 3623* | 3708* | 3786* | 4972# | | | | | | | | | | | | |
| PSW = 177776 | 701# | | | | | | | | | | | | | | | | | | | | |
| PWRVEC= 000024 | 792# | | | | | | | | | | | | | | | | | | | | |
| RACT = 004000 | 964# | | | | | | | | | | | | | | | | | | | | |
| RAND 010632 | 2218 | 2223 | 2243 | 2248 | 2618# | 2636 | 2638 | | | | | | | | | | | | | | |
| RANDOM 017571 | 3490 | 3852 | 4120# | | | | | | | | | | | | | | | | | | |
| RDCHR - 104410 | 4765 | 5157# | | | | | | | | | | | | | | | | | | | |
| RDLIN = 104411 | 5158# | | | | | | | | | | | | | | | | | | | | |
| RDWR 016610 | 3885 | 3901 | 3914 | 3984# | | | | | | | | | | | | | | | | | |
| RDY - 000200 | 949# | 1007# | 1041# | | | | | | | | | | | | | | | | | | |
| REGHLD 011522 | 2815* | 2820 | 2833# | | | | | | | | | | | | | | | | | | |
| RESTOR- 000017 | 1037# | 3240 | 3275 | 3309 | 3318 | 3522 | 3526 | 3614 | 3649 | 3683 | 3692 | | | | | | | | | | |
| RESVEC= 000010 | 787# | | | | | | | | | | | | | | | | | | | | |
| RING = 040000 | 928# | 961# | | | | | | | | | | | | | | | | | | | |
| RNF = 000020 | 1043# | 3236 | 3250 | 3272 | 3306 | 3313 | 3335 | 3610 | 3624 | 3646 | 3680 | 3687 | 3709 | | | | | | | | |
| RSEC = 000007 | 1033# | 3289 | 3662 | 3886 | 3915 | | | | | | | | | | | | | | | | |
| RSTAT - 000013 | 1035# | 3534 | | | | | | | | | | | | | | | | | | | |
| RTS = 000004 | 937# | 973# | 1588 | 1589 | 1619 | 1779 | 1780 | 1834 | 1895 | 1928 | 1929 | | | | | | | | | | |
| RUN 017534 | 1768 | 1814 | 1916 | 1959 | 1993 | 2027 | 4114# | | | | | | | | | | | | | | |
| RXCS - 177170 | 847# | 2046 | 2051* | 2054 | 2061* | 2068 | 2076* | 2082 | 2147* | 2166* | 2232* | 2257* | 2297* | | | | | | | | |
| | 2303 | 2319* | 2338* | 2344 | 2355* | 2366* | 2372 | 2382* | 2408* | 2417* | 2423 | 2447* | 2458 | | | | | | | | |

| | | | | | | | | | |
|----------|---------|-------|-------|-------|-------|-------|-------|------|--|
| \$ERTY | 025642 | 4905# | 4962 | | | | | | |
| \$ERTL | 001112 | 1115# | 4876* | 4898 | | | | | |
| \$ETABL | 001210 | 1156# | | | | | | | |
| \$ETEND | 001250 | 1100 | 1189# | | | | | | |
| \$FATAL | 001172 | 1149# | 4838* | | | | | | |
| \$FFLG | 025500 | 4801* | 4804* | 4832 | 4841* | 4849# | | | |
| \$FILLC | 001156 | 1136# | 4613 | 4644 | | | | | |
| \$FILLS | 001155 | 1135# | 4644 | | | | | | |
| \$GDADR | 001120 | 1119# | | | | | | | |
| \$GDDAT | 001124 | 1121# | | | | | | | |
| \$GTSWR | 024442 | 4666# | 5154 | | | | | | |
| \$HD = | 000000 | 675 | | | | | | | |
| \$HIBTS | 001000 | 1095# | | | | | | | |
| \$ICNT | 001104 | 1112# | | | | | | | |
| \$INTAG | 001135 | 1126# | 4682 | 4702 | 4797 | | | | |
| \$ITEMB | 001114 | 1116# | 4879* | 4887 | 4898 | 4909 | 4957 | | |
| \$LF | 001166 | 1141# | 4644 | 4781 | 4790 | 4898 | | | |
| \$LFLG | 025477 | 4842* | 4848# | | | | | | |
| \$LPADR | 001106 | 1113# | | | | | | | |
| \$LPERR | 001110 | 1114# | | | | | | | |
| \$LSTCN= | 177777 | 664# | | | | | | | |
| \$LSTIN= | 000000 | 664# | | | | | | | |
| \$LSTST= | 177777 | 664# | | | | | | | |
| \$LSTTA= | 000000 | 664# | | | | | | | |
| \$MADR1 | 001222 | 1174# | | | | | | | |
| \$MADR2 | 001226 | 1178# | | | | | | | |
| \$MADR3 | 001232 | 1181# | | | | | | | |
| \$MADR4 | 001236 | 1184# | | | | | | | |
| \$MAIL | 001170 | 1096 | 1100 | 1147# | 1550 | 1563 | 4588 | 4885 | |
| \$MAMS1 | 001220 | 1168# | | | | | | | |
| \$MAMS2 | 001224 | 1176# | | | | | | | |
| \$MAMS3 | 001230 | 1179# | | | | | | | |
| \$MAMS4 | 001234 | 1182# | | | | | | | |
| \$MBADR | 001002 | 1096# | | | | | | | |
| \$MFLG | 025476 | 4802* | 4808 | 4843* | 4847# | | | | |
| \$MNEW | 025222 | 2660 | 4669 | 4795# | | | | | |
| \$MSGAD | 001204 | 1154# | 4818* | 4821 | | | | | |
| \$MSGLG | 001206 | 1155# | 4823* | | | | | | |
| \$MSGTY | 001170 | 1148# | 4816 | 4824* | 4836 | 4840* | | | |
| \$MSWR | 025211 | 4666 | 4793# | | | | | | |
| \$MTYP1 | 001221 | 1169# | | | | | | | |
| \$MTYP2 | 001225 | 1177# | | | | | | | |
| \$MTYP3 | 001231 | 1180# | | | | | | | |
| \$MTYP4 | 001235 | 1183# | | | | | | | |
| \$NULL | 001154 | 1134# | 4615 | 4644 | | | | | |
| \$SOCNT | 026530 | 5072* | 5101* | 5114# | | | | | |
| \$SOMODE | 026532 | 5067* | 5071* | 5076 | 5079* | 5090* | 5116# | | |
| \$SPASS | 001176 | 1151# | 1550* | 2480* | 2482 | | | | |
| \$PASTM | 001006 | 1098# | | | | | | | |
| \$QUES | 001164 | 1139# | 2695 | 4644 | 4720 | 4774 | 4790 | 4898 | |
| \$RDCHR | 024714 | 4733# | 5157 | | | | | | |
| \$RDDEC= | ***** U | 5159 | | | | | | | |
| \$RDLIN | 025034 | 4761# | 5158 | | | | | | |
| \$RDOCT= | ***** U | 5159 | | | | | | | |

| | | |
|--------|------|-------|
| .SWRHI | 664# | |
| .SACT1 | 664# | |
| .SAPT8 | 664# | 1143# |
| .SAPTH | 664# | 1079 |
| .SAPTY | 664# | 4798 |
| .SCATC | 664# | |
| .SCMTA | 664# | 1101 |
| .SEOP | 664# | |
| .SERRO | 664# | 4855 |
| .SERRT | 664# | 4898 |
| .SREAD | 664# | 4644 |
| .SSCOP | 664# | |
| .STRAP | 664# | 5117 |
| .STYPD | 664# | 4973 |
| .STYPE | 664# | 4565 |
| .STYPO | 664# | 5040 |

| | | | | | | | | | | | | | | | |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| ADC | 2626 | 2629 | 3961 | | | | | | | | | | | | |
| ADD | 1667 | 1687 | 1719 | 2529 | 2548 | 2569 | 2625 | 2627 | 2628 | 2630 | 2675 | 2741 | 2767 | 2795 | 2830 |
| | 2897 | 2945 | 3263 | 3392 | 3451 | 3637 | 3765 | 3812 | 3962 | 3969 | 3970 | 4601 | 4681 | 4691 | 4700 |
| | 4815 | 4827 | 4839 | 4920 | 5005 | 5068 | 5078 | | | | | | | | |
| ASL | 2534 | 2567 | 2621 | 2687 | 2688 | 2689 | 3966 | 3967 | 4714 | 4715 | 4716 | 4917 | 4918 | 4919 | 5129 |
| ASLB | 5010 | | | | | | | | | | | | | | |
| ASR | 3960 | 4822 | | | | | | | | | | | | | |
| BCC | 5011 | | | | | | | | | | | | | | |
| BCS | 2568 | | | | | | | | | | | | | | |
| BEQ | 1519 | 1552 | 1564 | 1594 | 1599 | 1601 | 1639 | 1664 | 1678 | 1683 | 1689 | 1711 | 1716 | 1725 | 1733 |
| | 1766 | 1804 | 1808 | 1811 | 1821 | 1831 | 1847 | 1868 | 1874 | 1891 | 1914 | 1927 | 1935 | 1957 | 1969 |
| | 1991 | 2003 | 2025 | 2041 | 2090 | 2095 | 2141 | 2160 | 2177 | 2185 | 2189 | 2274 | 2501 | 2528 | 2547 |
| | 2649 | 2673 | 2686 | 2732 | 2890 | 2892 | 2938 | 2940 | 2973 | 2975 | 2977 | 3005 | 3009 | 3032 | 3036 |
| | 3058 | 3062 | 3085 | 3089 | 3129 | 3131 | 3135 | 3237 | 3247 | 3251 | 3262 | 3273 | 3304 | 3307 | 3314 |
| | 3322 | 3332 | 3336 | 3384 | 3391 | 3396 | 3398 | 3413 | 3418 | 3429 | 3450 | 3455 | 3463 | 3469 | 3488 |
| | 3500 | 3502 | 3611 | 3621 | 3625 | 3636 | 3647 | 3678 | 3681 | 3688 | 3696 | 3706 | 3710 | 3758 | 3764 |
| | 3769 | 3771 | 3780 | 3784 | 3811 | 3816 | 3824 | 3830 | 3850 | 3861 | 3934 | 3937 | 4024 | 4591 | 4604 |
| | 4639 | 4663 | 4698 | 4713 | 4809 | 4813 | 4833 | 4835 | 4871 | 4874 | 4922 | 4927 | 4933 | 4944 | 5095 |
| BGT | 2683 | 4710 | 4751 | 5019 | 5102 | | | | | | | | | | |
| BHI | 2638 | | | | | | | | | | | | | | |
| BIC | 1512 | 1589 | 1613 | 1779 | 1805 | 1928 | 2634 | 2668 | 2684 | 2999 | 3026 | 3052 | 3079 | 3105 | 3117 |
| | 3140 | 3474 | 3492 | 3552 | 3556 | 3835 | 3854 | 3963 | 4659 | 4676 | 4711 | 4738 | 4744 | 4752 | 5092 |
| BIS | 1588 | 1619 | 1757 | 1780 | 1794 | 1827 | 1828 | 1832 | 1834 | 1836 | 1857 | 1858 | 1895 | 1904 | 1905 |
| | 1929 | 1947 | 1948 | 1981 | 1982 | 2015 | 2016 | 2147 | 2166 | 2232 | 2257 | 2297 | 2319 | 2338 | 2355 |
| | 2366 | 2382 | 2408 | 2417 | 2447 | 2692 | 2886 | 2934 | 3015 | 3042 | 3068 | 3095 | 3113 | 3144 | 3195 |
| | 3213 | 3285 | 3381 | 3569 | 3575 | 3591 | 3658 | 3748 | 3754 | 3986 | 4718 | 5013 | 5014 | 5097 | 5098 |
| BISB | 4909 | | | | | | | | | | | | | | |
| BIT | 1600 | 1602 | 1732 | 1752 | 1765 | 1774 | 1807 | 1820 | 1830 | 1867 | 1887 | 1890 | 1913 | 1926 | 1931 |
| | 1934 | 1956 | 1965 | 1968 | 1990 | 1999 | 2002 | 2024 | 2040 | 2042 | 2134 | 2153 | 2172 | 2180 | 2213 |
| | 2238 | 2263 | 2268 | 2287 | 2307 | 2332 | 2361 | 2398 | 2435 | 2453 | 2564 | 2820 | 3172 | 3236 | 3250 |
| | 3272 | 3306 | 3313 | 3321 | 3335 | 3344 | 3454 | 3468 | 3487 | 3610 | 3624 | 3646 | 3680 | 3687 | 3695 |
| | 3709 | 3815 | 3829 | 3849 | 4873 | 4880 | | | | | | | | | |
| BITB | 1551 | 4590 | 4595 | 4627 | 4812 | | | | | | | | | | |
| BLO | 2636 | | | | | | | | | | | | | | |
| BLOS | 4764 | | | | | | | | | | | | | | |
| BLT | 2681 | 4618 | 4708 | 4749 | 5002 | 5018 | 5103 | | | | | | | | |
| BMI | 1782 | 2459 | 3158 | 5009 | | | | | | | | | | | |
| BNE | 1527 | 1541 | 1558 | 1562 | 1566 | 1575 | 1578 | 1580 | 1592 | 1603 | 1630 | 1632 | 1669 | 1692 | 1696 |
| | 1698 | 1704 | 1721 | 1753 | 1775 | 1777 | 1888 | 1923 | 1932 | 1966 | 2000 | 2043 | 2065 | 2079 | 2099 |
| | 2135 | 2154 | 2173 | 2175 | 2181 | 2183 | 2214 | 2239 | 2264 | 2266 | 2269 | 2271 | 2288 | 2308 | 2333 |
| | 2362 | 2399 | 2436 | 2454 | 2463 | 2525 | 2545 | 2565 | 2624 | 2671 | 2735 | 2738 | 2757 | 2760 | 2763 |
| | 2785 | 2788 | 2791 | 2806 | 2821 | 2824 | 2827 | 2844 | 2849 | 2861 | 2867 | 2894 | 2909 | 2915 | 2942 |
| | 3108 | 3111 | 3173 | 3200 | 3217 | 3234 | 3258 | 3343 | 3345 | 3366 | 3403 | 3405 | 3446 | 3486 | 3572 |
| | 3574 | 3581 | 3608 | 3632 | 3734 | 3751 | 3753 | 3775 | 3807 | 3848 | 3892 | 3908 | 3921 | 4007 | 4030 |
| | 4589 | 4596 | 4598 | 4606 | 4614 | 4628 | 4635 | 4655 | 4661 | 4679 | 4683 | 4689 | 4696 | 4703 | 4740 |
| | 4746 | 4768 | 4773 | 4779 | 4811 | 4817 | 4820 | 4837 | 4881 | 4886 | 4910 | 4936 | 5007 | 5093 | |
| BPL | 2304 | 2345 | 2373 | 2424 | 2589 | 2666 | 2967 | 3166 | 3232 | 3302 | 3606 | 3676 | 3883 | 3889 | 3905 |
| | 3918 | 4583 | 4632 | 4657 | 4673 | 4736 | 4742 | 4893 | 4993 | 5023 | 5091 | | | | |
| BR | 1496 | 1500 | 1505 | 1543 | 1568 | 1571 | 1633 | 1693 | 1701 | 1727 | 1736 | 1763 | 1784 | 1801 | 1813 |
| | 1843 | 1849 | 1863 | 1871 | 1876 | 1879 | 1893 | 1911 | 1937 | 1954 | 1971 | 1988 | 2005 | 2022 | 2049 |
| | 2057 | 2071 | 2085 | 2100 | 2143 | 2162 | 2178 | 2186 | 2227 | 2252 | 2301 | 2342 | 2370 | 2412 | 2421 |
| | 2451 | 2456 | 2469 | 2471 | 2530 | 2549 | 2552 | 2570 | 2576 | 2593 | 2693 | 2696 | 2739 | 2765 | 2793 |
| | 2828 | 2863 | 2895 | 2911 | 2943 | 2979 | 2982 | 2984 | 3011 | 3038 | 3064 | 3091 | 3137 | 3142 | 3219 |
| | 3244 | 3253 | 3260 | 3325 | 3338 | 3388 | 3393 | 3400 | 3407 | 3410 | 3415 | 3448 | 3466 | 3503 | 3618 |

| | | | | | | | | | | | | | | | |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| | 3627 | 3634 | 3699 | 3712 | 3762 | 3766 | 3773 | 3777 | 3782 | 3809 | 3827 | 3862 | 3895 | 3911 | 3924 |
| | 3939 | 3943 | 4027 | 4585 | 4611 | 4621 | 4630 | 4637 | 4692 | 4719 | 4721 | 4747 | 4775 | 4803 | 4825 |
| | 4891 | 4915 | 4939 | 4946 | 5004 | 5021 | 5069 | 5084 | 5105 | | | | | | |
| CLC | 2566 | | | | | | | | | | | | | | |
| CLR | 1499 | 1503 | 1504 | 1507 | 1508 | 1509 | 1520 | 1525 | 1550 | 1585 | 1586 | 1624 | 1625 | 1660 | 1661 |
| | 1671 | 1675 | 1708 | 1730 | 1756 | 1786 | 1791 | 1824 | 1826 | 1902 | 1945 | 1979 | 2013 | 2038 | 2060 |
| | 2122 | 2123 | 2124 | 2125 | 2128 | 2129 | 2130 | 2131 | 2146 | 2165 | 2205 | 2206 | 2207 | 2209 | 2210 |
| | 2211 | 2216 | 2231 | 2241 | 2256 | 2291 | 2292 | 2296 | 2311 | 2312 | 2318 | 2337 | 2348 | 2351 | 2352 |
| | 2353 | 2365 | 2376 | 2378 | 2379 | 2380 | 2402 | 2403 | 2407 | 2416 | 2438 | 2446 | 2466 | 2504 | 2517 |
| | 2520 | 2521 | 2540 | 2541 | 2554 | 2559 | 2561 | 2590 | 2662 | 2663 | 2753 | 2781 | 2842 | 2847 | 3013 |
| | 3040 | 3066 | 3093 | 3256 | 3267 | 3268 | 3327 | 3341 | 3377 | 3423 | 3424 | 3443 | 3444 | 3481 | 3482 |
| | 3630 | 3641 | 3642 | 3701 | 3715 | 3744 | 3789 | 3790 | 3804 | 3805 | 3843 | 3844 | 3873 | 3879 | 3897 |
| | 3913 | 3975 | 4670 | 4671 | 4908 | 4996 | 4999 | 5082 | | | | | | | |
| CLRB | 4610 | 4636 | 4780 | 4841 | 4842 | 4843 | 5025 | | | | | | | | |
| CMP | 1526 | 1540 | 1565 | 1668 | 1682 | 1688 | 1691 | 1695 | 1703 | 1715 | 1720 | 1724 | 1846 | 2094 | 2098 |
| | 2265 | 2270 | 2527 | 2546 | 2635 | 2637 | 2670 | 2680 | 2682 | 2731 | 2843 | 2848 | 2866 | 2889 | 2893 |
| | 2914 | 2937 | 2941 | 2972 | 2974 | 2976 | 3004 | 3008 | 3031 | 3035 | 3057 | 3061 | 3084 | 3088 | 3107 |
| | 3130 | 3134 | 3216 | 3233 | 3257 | 3261 | 3303 | 3342 | 3383 | 3402 | 3404 | 3428 | 3445 | 3449 | 3462 |
| | 3485 | 3499 | 3501 | 3607 | 3631 | 3635 | 3677 | 3757 | 3774 | 3806 | 3810 | 3823 | 3847 | 3860 | 3891 |
| | 3907 | 3920 | 3933 | 3936 | 4023 | 4654 | 4660 | 4678 | 4688 | 4695 | 4707 | 4709 | 4739 | 4745 | 4748 |
| | 4750 | 4763 | 5017 | | | | | | | | | | | | |
| CMPB | 1518 | 1563 | 1593 | 1598 | 2648 | 3128 | 4588 | 4603 | 4605 | 4613 | 4634 | 4638 | 4662 | 4682 | 4702 |
| | 4767 | 4772 | 4778 | 4810 | 4885 | | | | | | | | | | |
| COM | 2063 | | | | | | | | | | | | | | |
| DEC | 2064 | 2078 | 2734 | 2737 | 2759 | 2762 | 2787 | 2790 | 2805 | 2823 | 2826 | 3199 | 3365 | 3390 | 3580 |
| | 3733 | 3763 | 4006 | 4029 | 4916 | | | | | | | | | | |
| DECB | 4617 | 4620 | 5090 | 5101 | | | | | | | | | | | |
| EMT | 692 | | | | | | | | | | | | | | |
| HALT | 1605 | 2503 | 3875 | 3894 | 3910 | 3923 | 3946 | 3993 | 4012 | 4026 | 4584 | 4894 | | | |
| INC | 1495 | 1498 | 1502 | 1557 | 1591 | 1699 | 1723 | 1783 | 1878 | 2221 | 2246 | 2465 | 2480 | 2505 | 2526 |
| | 2551 | 2594 | 2623 | 2691 | 2736 | 2761 | 2764 | 2789 | 2792 | 2825 | 2956 | 2978 | 2986 | 2987 | 3010 |
| | 3014 | 3037 | 3041 | 3063 | 3067 | 3090 | 3094 | 3112 | 3114 | 3118 | 3136 | 3141 | 3248 | 3249 | 3271 |
| | 3305 | 3323 | 3333 | 3334 | 3387 | 3419 | 3420 | 3427 | 3461 | 3464 | 3473 | 3483 | 3484 | 3551 | 3555 |
| | 3622 | 3623 | 3645 | 3679 | 3697 | 3707 | 3708 | 3761 | 3785 | 3786 | 3793 | 3822 | 3825 | 3834 | 3845 |
| | 3846 | 3890 | 3906 | 3919 | 3931 | 3938 | 3941 | 4717 | 4840 | 4876 | 4963 | 4964 | 5003 | 5096 | 5104 |
| INCB | 4640 | 4870 | | | | | | | | | | | | | |
| IOT | 693 | | | | | | | | | | | | | | |
| JMP | 1064 | 1067 | 1070 | 1072 | 1074 | 1636 | 1640 | 1822 | 1924 | 2102 | 2190 | 2275 | 2506 | 3175 | 3178 |
| | 3264 | 3269 | 3329 | 3348 | 3425 | 3430 | 3432 | 3452 | 3638 | 3643 | 3703 | 3716 | 3791 | 3794 | 3813 |
| | 3947 | 4685 | 4771 | | | | | | | | | | | | |
| JSR | 1597 | 1610 | 1622 | 1627 | 1652 | 1759 | 1797 | 1839 | 1851 | 1860 | 1897 | 1907 | 1940 | 1950 | 1974 |
| | 1984 | 2008 | 2018 | 2045 | 2053 | 2067 | 2081 | 2149 | 2168 | 2218 | 2223 | 2226 | 2234 | 2243 | 2248 |
| | 2251 | 2259 | 2299 | 2321 | 2340 | 2357 | 2368 | 2384 | 2410 | 2419 | 2449 | 2679 | 2733 | 2758 | 2786 |
| | 2822 | 3235 | 3312 | 3609 | 3686 | 3877 | 3885 | 3898 | 3901 | 3914 | 3926 | 3929 | 3989 | 4008 | 4593 |
| | 4612 | 4619 | 4626 | 4706 | 4829 | 4882 | 4888 | 4962 | | | | | | | |
| MOV | 1511 | 1517 | 1524 | 1528 | 1530 | 1531 | 1532 | 1533 | 1536 | 1537 | 1538 | 1539 | 1544 | 1546 | 1547 |
| | 1548 | 1553 | 1582 | 1583 | 1608 | 1635 | 1654 | 1655 | 1657 | 1665 | 1672 | 1673 | 1676 | 1679 | 1681 |
| | 1684 | 1700 | 1706 | 1707 | 1709 | 1712 | 1714 | 1717 | 1726 | 1729 | 1754 | 1755 | 1787 | 1788 | 1789 |
| | 1792 | 1833 | 1837 | 1845 | 1856 | 1866 | 1892 | 1894 | 1903 | 1936 | 1938 | 1946 | 1970 | 1972 | 1980 |
| | 2004 | 2006 | 2014 | 2051 | 2059 | 2061 | 2062 | 2074 | 2075 | 2076 | 2077 | 2088 | 2089 | 2093 | 2120 |
| | 2126 | 2132 | 2137 | 2138 | 2142 | 2145 | 2156 | 2157 | 2161 | 2164 | 2203 | 2217 | 2219 | 2222 | 2224 |
| | 2229 | 2242 | 2244 | 2247 | 2249 | 2254 | 2290 | 2294 | 2313 | 2314 | 2316 | 2335 | 2336 | 2349 | 2354 |
| | 2364 | 2377 | 2381 | 2401 | 2405 | 2414 | 2439 | 2440 | 2441 | 2442 | 2444 | 2467 | 2468 | 2482 | 2485 |
| | 2488 | 2491 | 2494 | 2497 | 2498 | 2515 | 2516 | 2519 | 2532 | 2535 | 2542 | 2558 | 2562 | 2572 | 2618 |

| | | | | | | | | | | | | | | | | | | |
|--------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|--|--|--|
| .DSABL | 4722 | | | | | | | | | | | | | | | | | |
| .ENABL | 664 | 4647 | | | | | | | | | | | | | | | | |
| .END | 5164 | | | | | | | | | | | | | | | | | |
| .ENDC | 670 | 692 | 784 | 798 | 803 | 805 | 826 | 828 | 868 | 870 | 922 | 924 | 955 | 957 | 1017 | | | |
| | 1019 | 1052 | 1054 | 1082 | 1084 | 1091 | 1104 | 1108 | 1110 | 1138 | 1139 | 1143 | 1146 | 1168 | 1176 | | | |
| | 1179 | 1182 | 1185 | 1186 | 1187 | 1188 | 1191 | 1442 | 1444 | 1460 | 1462 | 1493 | 1528 | 1529 | 1530 | | | |
| | 1532 | 1534 | 1555 | 1559 | 1565 | 1571 | 1573 | 1644 | 1651 | 1749 | 1751 | 1771 | 1773 | 1817 | 1819 | | | |
| | 1884 | 1886 | 1919 | 1921 | 1962 | 1964 | 1996 | 1998 | 2030 | 2037 | 2109 | 2119 | 2194 | 2202 | 2279 | | | |
| | 2286 | 2326 | 2331 | 2391 | 2397 | 2428 | 2434 | 2477 | 2479 | 2615 | 2617 | 2654 | 2656 | 2699 | 2706 | | | |
| | 2721 | 2725 | 2748 | 2752 | 2776 | 2780 | 2801 | 2803 | 2810 | 2814 | 2838 | 2840 | 2854 | 2856 | 2877 | | | |
| | 2880 | 2902 | 2904 | 2925 | 2928 | 2952 | 2955 | 2962 | 2964 | 2996 | 2998 | 3023 | 3025 | 3049 | 3051 | | | |
| | 3076 | 3078 | 3102 | 3104 | 3149 | 3156 | 3188 | 3191 | 3208 | 3210 | 3224 | 3230 | 3280 | 3282 | 3294 | | | |
| | 3300 | 3356 | 3359 | 3371 | 3376 | 3437 | 3442 | 3478 | 3480 | 3507 | 3509 | 3518 | 3520 | 3530 | 3532 | | | |
| | 3538 | 3540 | 3547 | 3550 | 3562 | 3565 | 3586 | 3588 | 3598 | 3604 | 3653 | 3655 | 3668 | 3674 | 3724 | | | |
| | 3727 | 3738 | 3743 | 3798 | 3803 | 3840 | 3842 | 3866 | 3872 | 3956 | 3958 | 3980 | 3983 | 3998 | 4001 | | | |
| | 4017 | 4019 | 4037 | 4039 | 4568 | 4597 | 4647 | 4648 | 4650 | 4686 | 4722 | 4726 | 4754 | 4755 | 4762 | | | |
| | 4764 | 4772 | 4774 | 4790 | 4791 | 4797 | 4801 | 4802 | 4805 | 4832 | 4847 | 4858 | 4861 | 4870 | 4877 | | | |
| | 4882 | 4883 | 4884 | 4892 | 4897 | 4898 | 4901 | 4916 | 4954 | 4976 | 5043 | 5120 | 5126 | 5129 | 5148 | | | |
| | 5149 | 5150 | 5151 | 5152 | 5153 | 5154 | 5155 | 5156 | 5157 | 5158 | 5159 | | | | | | | |
| .EQUIV | 692 | 693 | 701 | 746 | 747 | 748 | 749 | 750 | 751 | 752 | 753 | 754 | 755 | 774 | 775 | | | |
| | 776 | 777 | 778 | 779 | 780 | 781 | 782 | 783 | | | | | | | | | | |
| .EVEN | 1146 | 1573 | 2509 | 4511 | 4797 | 4850 | 4953 | | | | | | | | | | | |
| .IF | 666 | 690 | 756 | 784 | 802 | 804 | 825 | 827 | 867 | 869 | 921 | 923 | 954 | 956 | 1016 | | | |
| | 1018 | 1051 | 1053 | 1081 | 1083 | 1090 | 1103 | 1107 | 1109 | 1138 | 1142 | 1143 | 1145 | 1168 | 1176 | | | |
| | 1179 | 1182 | 1185 | 1186 | 1187 | 1188 | 1189 | 1191 | 1441 | 1443 | 1459 | 1461 | 1493 | 1523 | 1528 | | | |
| | 1530 | 1532 | 1534 | 1550 | 1558 | 1559 | 1560 | 1563 | 1572 | 1643 | 1650 | 1748 | 1750 | 1770 | 1772 | | | |
| | 1816 | 1818 | 1883 | 1885 | 1918 | 1920 | 1961 | 1963 | 1995 | 1997 | 2029 | 2036 | 2108 | 2118 | 2193 | | | |
| | 2201 | 2278 | 2285 | 2325 | 2330 | 2390 | 2396 | 2427 | 2433 | 2476 | 2478 | 2614 | 2616 | 2653 | 2655 | | | |
| | 2698 | 2705 | 2720 | 2724 | 2747 | 2751 | 2775 | 2779 | 2800 | 2802 | 2809 | 2813 | 2837 | 2839 | 2853 | | | |
| | 2855 | 2876 | 2879 | 2901 | 2903 | 2924 | 2927 | 2951 | 2954 | 2961 | 2963 | 2995 | 2997 | 3022 | 3024 | | | |
| | 3048 | 3050 | 3075 | 3077 | 3101 | 3103 | 3148 | 3155 | 3187 | 3190 | 3207 | 3209 | 3223 | 3229 | 3279 | | | |
| | 3281 | 3293 | 3299 | 3355 | 3358 | 3370 | 3375 | 3436 | 3441 | 3477 | 3479 | 3506 | 3508 | 3517 | 3519 | | | |
| | 3529 | 3531 | 3537 | 3539 | 3546 | 3549 | 3561 | 3564 | 3585 | 3587 | 3597 | 3603 | 3652 | 3654 | 3667 | | | |
| | 3673 | 3723 | 3726 | 3737 | 3742 | 3797 | 3802 | 3839 | 3841 | 3865 | 3871 | 3955 | 3957 | 3979 | 3982 | | | |
| | 3997 | 4000 | 4016 | 4018 | 4036 | 4038 | 4567 | 4588 | 4646 | 4648 | 4649 | 4650 | 4678 | 4725 | 4726 | | | |
| | 4754 | 4762 | 4763 | 4767 | 4773 | 4789 | 4790 | 4797 | 4800 | 4802 | 4805 | 4832 | 4847 | 4857 | 4860 | | | |
| | 4870 | 4873 | 4880 | 4882 | 4883 | 4885 | 4892 | 4896 | 4897 | 4898 | 4900 | 4915 | 4931 | 4975 | 5042 | | | |
| | 5119 | 5125 | 5129 | 5140 | 5149 | 5150 | 5151 | 5152 | 5153 | 5154 | 5156 | 5157 | 5158 | 5159 | | | | |
| .IFF | 690 | 803 | 805 | 826 | 828 | 868 | 870 | 922 | 924 | 955 | 957 | 1017 | 1019 | 1052 | 1054 | | | |
| | 1082 | 1084 | 1091 | 1104 | 1107 | 1110 | 1138 | 1143 | 1146 | 1442 | 1444 | 1460 | 1462 | 1528 | 1558 | | | |
| | 1559 | 1644 | 1651 | 1749 | 1751 | 1771 | 1773 | 1817 | 1819 | 1884 | 1886 | 1919 | 1921 | 1962 | 1964 | | | |
| | 1996 | 1998 | 2030 | 2037 | 2109 | 2119 | 2194 | 2202 | 2279 | 2286 | 2326 | 2331 | 2391 | 2397 | 2428 | | | |
| | 2434 | 2477 | 2479 | 2615 | 2617 | 2654 | 2656 | 2699 | 2706 | 2721 | 2725 | 2748 | 2752 | 2776 | 2780 | | | |
| | 2801 | 2803 | 2810 | 2814 | 2838 | 2840 | 2854 | 2856 | 2877 | 2880 | 2902 | 2904 | 2925 | 2928 | 2952 | | | |
| | 2955 | 2962 | 2964 | 2996 | 2998 | 3023 | 3025 | 3049 | 3051 | 3076 | 3078 | 3102 | 3104 | 3149 | 3156 | | | |
| | 3188 | 3191 | 3208 | 3210 | 3224 | 3230 | 3280 | 3282 | 3294 | 3300 | 3356 | 3359 | 3371 | 3376 | 3437 | | | |
| | 3442 | 3478 | 3480 | 3507 | 3509 | 3518 | 3520 | 3530 | 3532 | 3538 | 3540 | 3547 | 3550 | 3562 | 3565 | | | |
| | 3586 | 3588 | 3598 | 3604 | 3653 | 3655 | 3668 | 3674 | 3724 | 3727 | 3738 | 3743 | 3798 | 3803 | 3840 | | | |
| | 3842 | 3866 | 3872 | 3956 | 3958 | 3980 | 3983 | 3998 | 4001 | 4017 | 4019 | 4037 | 4039 | 4568 | 4647 | | | |
| | 4650 | 4726 | 4728 | 4733 | 4754 | 4755 | 4764 | 4773 | 4790 | 4801 | 4858 | 4860 | 4873 | 4896 | 4897 | | | |
| | 4898 | 4901 | 4915 | 4931 | 4976 | 5043 | 5120 | 5126 | | | | | | | | | | |
| .IFT | 1573 | 4728 | 4733 | 4883 | | | | | | | | | | | | | | |
| .IFTF | 1573 | 4665 | 4726 | 4729 | 4882 | | | | | | | | | | | | | |
| .IIF | 665 | 670 | 675 | 1142 | 1146 | 1529 | 1534 | 1559 | 2483 | 2486 | 2489 | 2492 | 2495 | 2659 | 3459 | | | |

| | | | | | | | | | | | | | | | |
|--------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| | 3820 | 4644 | 4647 | 4668 | 4770 | 4782 | 4790 | 4797 | 4861 | 4862 | 4863 | 4864 | 4865 | 4869 | 4895 |
| | 4897 | 4898 | 4913 | 4938 | 4942 | 5148 | 5149 | 5150 | 5151 | 5152 | 5154 | 5156 | 5157 | 5158 | 5159 |
| .IRP | 1493 | 4806 | 4807 | 4828 | 4844 | 4845 | 4986 | 5026 | | | | | | | |
| .LIST | 1 | 664 | 798 | 1138 | 1143 | 1146 | 1493 | 1534 | 1559 | 1560 | 1573 | 4754 | 4897 | 5140 | 5148 |
| | 5149 | 5150 | 5151 | 5152 | 5153 | 5154 | 5155 | 5156 | 5157 | 5158 | 5159 | 5160 | | | |
| .MACRO | 1101 | 1550 | 5140 | | | | | | | | | | | | |
| .MCALL | 664 | 798 | 1143 | 1534 | 1560 | | | | | | | | | | |
| .MEXIT | 1190 | | | | | | | | | | | | | | |
| .NLIST | 1 | 664 | 798 | 1138 | 1143 | 1146 | 1493 | 1534 | 1559 | 1560 | 1573 | 4754 | 4897 | 5140 | 5148 |
| | 5149 | 5150 | 5151 | 5152 | 5153 | 5154 | 5155 | 5156 | 5157 | 5158 | 5159 | 5160 | | | |
| .PAGE | 1101 | 1191 | 4565 | 4644 | 4798 | 4855 | 4898 | 4973 | 5040 | 5117 | | | | | |
| .REM | 1 | | | | | | | | | | | | | | |
| .SBTTL | 676 | 688 | 1079 | 1101 | 1143 | 1191 | 1493 | 1522 | 1555 | 1560 | 1642 | 2512 | 2950 | 4034 | 4565 |
| | 4644 | 4798 | 4855 | 4898 | 4973 | 5040 | 5117 | 5140 | | | | | | | |
| .TITLE | 665 | | | | | | | | | | | | | | |
| .WORD | 1056 | 1057 | 1095 | 1096 | 1097 | 1098 | 1099 | 1100 | 1109 | 1112 | 1113 | 1114 | 1115 | 1118 | 1119 |
| | 1120 | 1121 | 1122 | 1123 | 1124 | 1127 | 1128 | 1129 | 1148 | 1149 | 1150 | 1151 | 1152 | 1153 | 1154 |
| | 1155 | 1159 | 1160 | 1161 | 1174 | 1178 | 1181 | 1184 | 1185 | 1186 | 1187 | 1188 | 1515 | 1616 | 2537 |
| | 2574 | 2644 | 2645 | 4513 | 4514 | 4516 | 4518 | 4520 | 4522 | 4524 | 4526 | 4528 | 4531 | 4534 | 4537 |
| | 4540 | 4543 | 4546 | 4548 | 4551 | 4552 | 4553 | 4555 | 4556 | 4557 | 4558 | 4560 | 4561 | 4562 | 4563 |
| | 4594 | 4641 | 4830 | 4924 | 4929 | 5116 | 5147 | | | | | | | | |

ERRORS DETECTED: 0

PDT-11 EXERCISER
CVKDAA.M11

MACY11 27(654) 20-SEP-78 10:40 PAGE 136

G 11

SEQ 0136

*Z:CVKDAA/I,Z:CVKDAA.SEQ/CRF/SOL=CVKDAA.M11
RUN-TIME: 24 15 3 SECONDS
CORE USED: 25K