

VS60

INSTRUCTION TEST PART 1
MD-11-DZVSA-A

EP-DZVSA-A-DL-A
COPYRIGHT © 1976
FICHE 1 OF 1

NOV 1976
digital
MADE IN USA

This section of the document contains a grid of 150 small, illegible data tables or charts, arranged in 10 columns and 15 rows. Each cell in the grid appears to contain a small table with multiple columns and rows of text, likely representing test results or data points. The text is too small to be read, but the layout is consistent across the entire grid.

A small, illegible table or chart located in the bottom right corner of the page. It consists of a few columns and rows of text, but the content is not legible.

801

.REM :

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZVSA-A-D
PRODUCT NAME: VS-60 INSTRUCTION TEST PART 1
DATE: FEBRUARY 1976
MAINTAINER: DIAGNOSTIC ENGINEERING
AUTHORS: R. SHOOP & R. MOORE

COPYRIGHT (C) 1976
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

THIS SOFTWARE IS FURNISHED UNDER A LICENSE FOR USE ONLY ON A SINGLE COMPUTER SYSTEM AND MAY BE COPIED ONLY WITH THE INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE, OR ANY OTHER COPIES THEREOF, MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE TERMS. TITLE TO AND OWNERSHIP OF THE SOFTWARE SHALL AT ALL TIMES REMAIN IN DEC.

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

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

0-4001 1001 1002 1003 1004 1005 1006 1007 1008 1009 1010 1011 1012 1013 1014 1015 1016 1017 1018 1019 1020 1021 1022 1023 1024 1025 1026 1027 1028 1029 1030 1031 1032 1033 1034 1035 1036 1037 1038 1039 1040 1041 1042 1043 1044 1045 1046 1047 1048 1049 1050 1051 1052 1053 1054 1055 1056 1057 1058 1059 1060 1061 1062 1063 1064 1065 1066 1067 1068 1069 1070 1071 1072 1073 1074 1075 1076 1077 1078 1079 1080 1081 1082 1083 1084 1085 1086 1087 1088 1089 1090 1091 1092 1093 1094 1095 1096 1097 1098 1099 1100 1101 1102 1103 1104 1105 1106 1107 1108 1109 1110 1111 1112 1113 1114 1115 1116 1117 1118 1119 1120 1121 1122 1123 1124 1125 1126 1127 1128 1129 1130 1131 1132 1133 1134 1135 1136 1137 1138 1139 1140 1141 1142 1143 1144 1145 1146 1147 1148 1149 1150 1151 1152 1153 1154 1155 1156 1157 1158 1159 1160 1161 1162 1163 1164 1165 1166 1167 1168 1169 1170 1171 1172 1173 1174 1175 1176 1177 1178 1179 1180 1181 1182 1183 1184 1185 1186 1187 1188 1189 1190 1191 1192 1193 1194 1195 1196 1197 1198 1199 1200 1201 1202 1203 1204 1205 1206 1207 1208 1209 1210 1211 1212 1213 1214 1215 1216 1217 1218 1219 1220 1221 1222 1223 1224 1225 1226 1227 1228 1229 1230 1231 1232 1233 1234 1235 1236 1237 1238 1239 1240 1241 1242 1243 1244 1245 1246 1247 1248 1249 1250 1251 1252 1253 1254 1255 1256 1257 1258 1259 1260 1261 1262 1263 1264 1265 1266 1267 1268 1269 1270 1271 1272 1273 1274 1275 1276 1277 1278 1279 1280 1281 1282 1283 1284 1285 1286 1287 1288 1289 1290 1291 1292 1293 1294 1295 1296 1297 1298 1299 1300 1301 1302 1303 1304 1305 1306 1307 1308 1309 1310 1311 1312 1313 1314 1315 1316 1317 1318 1319 1320 1321 1322 1323 1324 1325 1326 1327 1328 1329 1330 1331 1332 1333 1334 1335 1336 1337 1338 1339 1340 1341 1342 1343 1344 1345 1346 1347 1348 1349 1350 1351 1352 1353 1354 1355 1356 1357 1358 1359 1360 1361 1362 1363 1364 1365 1366 1367 1368 1369 1370 1371 1372 1373 1374 1375 1376 1377 1378 1379 1380 1381 1382 1383 1384 1385 1386 1387 1388 1389 1390 1391 1392 1393 1394 1395 1396 1397 1398 1399 1400 1401 1402 1403 1404 1405 1406 1407 1408 1409 1410 1411 1412 1413 1414 1415 1416 1417 1418 1419 1420 1421 1422 1423 1424 1425 1426 1427 1428 1429 1430 1431 1432 1433 1434 1435 1436 1437 1438 1439 1440 1441 1442 1443 1444 1445 1446 1447 1448 1449 1450 1451 1452 1453 1454 1455 1456 1457 1458 1459 1460 1461 1462 1463 1464 1465 1466 1467 1468 1469 1470 1471 1472 1473 1474 1475 1476 1477 1478 1479 1480 1481 1482 1483 1484 1485 1486 1487 1488 1489 1490 1491 1492 1493 1494 1495 1496 1497 1498 1499 1500 1501 1502 1503 1504 1505 1506 1507 1508 1509 1510 1511 1512 1513 1514 1515 1516 1517 1518 1519 1520 1521 1522 1523 1524 1525 1526 1527 1528 1529 1530 1531 1532 1533 1534 1535 1536 1537 1538 1539 1540 1541 1542 1543 1544 1545 1546 1547 1548 1549 1550 1551 1552 1553 1554 1555 1556 1557 1558 1559 1560 1561 1562 1563 1564 1565 1566 1567 1568 1569 1570 1571 1572 1573 1574 1575 1576 1577 1578 1579 1580 1581 1582 1583 1584 1585 1586 1587 1588 1589 1590 1591 1592 1593 1594 1595 1596 1597 1598 1599 1600 1601 1602 1603 1604 1605 1606 1607 1608 1609 1610 1611 1612 1613 1614 1615 1616 1617 1618 1619 1620 1621 1622 1623 1624 1625 1626 1627 1628 1629 1630 1631 1632 1633 1634 1635 1636 1637 1638 1639 1640 1641 1642 1643 1644 1645 1646 1647 1648 1649 1650 1651 1652 1653 1654 1655 1656 1657 1658 1659 1660 1661 1662 1663 1664 1665 1666 1667 1668 1669 1670 1671 1672 1673 1674 1675 1676 1677 1678 1679 1680 1681 1682 1683 1684 1685 1686 1687 1688 1689 1690 1691 1692 1693 1694 1695 1696 1697 1698 1699 1700 1701 1702 1703 1704 1705 1706 1707 1708 1709 1710 1711 1712 1713 1714 1715 1716 1717 1718 1719 1720 1721 1722 1723 1724 1725 1726 1727 1728 1729 1730 1731 1732 1733 1734 1735 1736 1737 1738 1739 1740 1741 1742 1743 1744 1745 1746 1747 1748 1749 1750 1751 1752 1753 1754 1755 1756 1757 1758 1759 1760 1761 1762 1763 1764 1765 1766 1767 1768 1769 1770 1771 1772 1773 1774 1775 1776 1777 1778 1779 1780 1781 1782 1783 1784 1785 1786 1787 1788 1789 1790 1791 1792 1793 1794 1795 1796 1797 1798 1799 1800 1801 1802 1803 1804 1805 1806 1807 1808 1809 1810 1811 1812 1813 1814 1815 1816 1817 1818 1819 1820 1821 1822 1823 1824 1825 1826 1827 1828 1829 1830 1831 1832 1833 1834 1835 1836 1837 1838 1839 1840 1841 1842 1843 1844 1845 1846 1847 1848 1849 1850 1851 1852 1853 1854 1855 1856 1857 1858 1859 1860 1861 1862 1863 1864 1865 1866 1867 1868 1869 1870 1871 1872 1873 1874 1875 1876 1877 1878 1879 1880 1881 1882 1883 1884 1885 1886 1887 1888 1889 1890 1891 1892 1893 1894 1895 1896 1897 1898 1899 1900 1901 1902 1903 1904 1905 1906 1907 1908 1909 1910 1911 1912 1913 1914 1915 1916 1917 1918 1919 1920 1921 1922 1923 1924 1925 1926 1927 1928 1929 1930 1931 1932 1933 1934 1935 1936 1937 1938 1939 1940 1941 1942 1943 1944 1945 1946 1947 1948 1949 1950 1951 1952 1953 1954 1955 1956 1957 1958 1959 1960 1961 1962 1963 1964 1965 1966 1967 1968 1969 1970 1971 1972 1973 1974 1975 1976 1977 1978 1979 1980 1981 1982 1983 1984 1985 1986 1987 1988 1989 1990 1991 1992 1993 1994 1995 1996 1997 1998 1999 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016 2017 2018 2019 2020 2021 2022 2023 2024 2025 2026 2027 2028 2029 2030 2031 2032 2033 2034 2035 2036 2037 2038 2039 2040 2041 2042 2043 2044 2045 2046 2047 2048 2049 2050 2051 2052 2053 2054 2055 2056 2057 2058 2059 2060 2061 2062 2063 2064 2065 2066 2067 2068 2069 2070 2071 2072 2073 2074 2075 2076 2077 2078 2079 2080 2081 2082 2083 2084 2085 2086 2087 2088 2089 2090 2091 2092 2093 2094 2095 2096 2097 2098 2099 2100 2101 2102 2103 2104 2105 2106 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 2152 2153 2154 2155 2156 2157 2158 2159 2160 2161 2162 2163 2164 2165 2166 2167 2168 2169 2170 2171 2172 2173 2174 2175 2176 2177 2178 2179 2180 2181 2182 2183 2184 2185 2186 2187 2188 2189 2190 2191 2192 2193 2194 2195 2196 2197 2198 2199 2200 2201 2202 2203 2204 2205 2206 2207 2208 2209 2210 2211 2212 2213 2214 2215 2216 2217 2218 2219 2220 2221 2222 2223 2224 2225 2226 2227 2228 2229 2230 2231 2232 2233 2234 2235 2236 2237 2238 2239 2240 2241 2242 2243 2244 2245 2246 2247 2248 2249 2250 2251 2252 2253 2254 2255 2256 2257 2258 2259 2260 2261 2262 2263 2264 2265 2266 2267 2268 2269 2270 2271 2272 2273 2274 2275 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 2324 2325 2326 2327 2328 2329 2330 2331 2332 2333 2334 2335 2336 2337 2338 2339 2340 2341 2342 2343 2344 2345 2346 2347 2348 2349 2350 2351 2352 2353 2354 2355 2356 2357 2358 2359 2360 2361 2362 2363 2364 2365 2366 2367 2368 2369 2370 2371 2372 2373 2374 2375 2376 2377 2378 2379 2380 2381 2382 2383 2384 2385 2386 2387 2388 2389 2390 2391 2392 2393 2394 2395 2396 2397 2398 2399 2400 2401 2402 2403 2404 2405 2406 2407 2408 2409 2410 2411 2412 2413 2414 2415 2416 2417 2418 2419 2420 2421 2422 2423 2424 2425 2426 2427 2428 2429 2430 2431 2432 2433 2434 2435 2436 2437 2438 2439 2440 2441 2442 2443 2444 2445 2446 2447 2448 2449 2450 2451 2452 2453 2454 2455 2456 2457 2458 2459 2460 2461 2462 2463 2464 2465 2466 2467 2468 2469 2470 2471 2472 2473 2474 2475 2476 2477 2478 2479 2480 2481 2482 2483 2484 2485 2486 2487 2488 2489 2490 2491 2492 2493 2494 2495 2496 2497 2498 2499 2500 2501 2502 2503 2504 2505 2506 2507 2508 2509 2510 2511 2512 2513 2514 2515 2516 2517 2518 2519 2520 2521 2522 2523 2524 2525 2526 2527 2528 2529 2530 2531 2532 2533 2534 2535 2536 2537 2538 2539 2540 2541 2542 2543 2544 2545 2546 2547 2548 2549 2550 2551 2552 2553 2554 2555 2556 2557 2558 2559 2560 2561 2562 2563 2564 2565 2566 2567 2568 2569 2570 2571 2572 2573 2574 2575 2576 2577 2578 2579 2580 2581 2582 2583 2584 2585 2586 2587 2588 2589 2590 2591 2592 2593 2594 2595 2596 2597 2598 2599 2600 2601 2602 2603 2604 2605 2606 2607 2608 2609 2610 2611 2612 2613 2614 2615 2616 2617 2618 2619 2620 2621 2622 2623 2624 2625 2626 2627 2628 2629 2630 2631 2632 2633 2634 2635 2636 2637 2638 2639 2640 2641 2642 2643 2644 2645 2646 2647 2648 2649 2650 2651 2652 2653 2654 2655 2656 2657 2658 2659 2660 2661 2662 2663 2664 2665 2666 2667 2668 2669 2670 2671 2672 2673 2674 2675 2676 2677 2678 2679 2680 2681 2682 2683 2684 2685 2686 2687 2688 2689 2690 2691 2692 2693 2694 2695 2696 2697 2698 2699 2700 2701 2702 2703 2704 2705 2706 2707 2708 2709 2710 2711 2712 2713 2714 2715 2716 2717 2718 2719 2720 2721 2722 2723 2724 2725 2726 2727 2728 2729 2730 2731 2732 2733 2734 2735 2736 2737 2738 2739 2740 2741 2742 2743 2744 2745 2746 2747 2748 2749 2750 2751 2752 2753 2754 2755 2756 2757 2758 2759 2760 2761 2762 2763 2764 2765 2766 2767 2768 2769 2770 2771 2772 2773 2774 2775 2776 2777 2778 2779 2780 2781 2782 2783 2784 2785 2786 2787 2788 2789 2790 2791 2792 2793 2794 2795 2796 2797 2798 2799 2800 2801 2802 2803 2804 2805 2806 2807 2808 2809 2810 2811 2812 2813 2814 2815 2816 2817 2818 2819 2820 2821 2822 2823 2824 2825 2826 2827 2828 2829 2830 2831 2832 2833 2834 2835 2836 2837 2838 2839 2840 2841 2842 2843 2844 2845 2846 2847 2848 2849 2850 2851 2852 2853 2854 2855 2856 2857 2858 2859 2860 2861 2862 2863 2864 2865 2866 2867 2868 2869 2870 2871 2872 2873 2874 2875 2876 2877 2878 2879 2880 2881 2882 2883 2884 2885 2886 2887 2888 2889 2890 2891 2892 2893 2894 2895 2896 2897 2898 2899 2900 2901 2902 2903 2904 2905 2906 2907 2908 2909 2910 2911 2912 2913 2914 2915 2916 2917 2918 2919 2920 2921 2922 2923 2924 2925 2926 2927 2928 2929 2930 2931 2932 2933 2934 2935 2936 2937 2938 2939 2940 2941 2942 2943 2944 2945 2946 2947 2948 2949 2950 2951 2952 2953 2954 2955 2956 2957 2958 2959 2960 2961 2962 2963 2964 2965 2966 2967 2968 2969 2970 2971 2972 2973 2974 2975 2976 2977 2978 2979 2980 2981 2982 2983 2984 2985 2986 2987 2988 2989 2990 2991 2992 2993 2994 2995 2996 2997 2998 2999 3000 3001 3002 3003 3004 3005 3006 3007 3008 3009 3010 3011 3012 3013 3014 3015 3016 3017 3018 3019 3020 3021 3022 3023 3024 3025 3026 3027 3028 3029 3030 3031 3032 3033 3034 3035 3036 3037 3038 3039 3040 3041 3042 3043 3044 3045 3046 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 3100 3101 3102 3103 3104 3105 3106 3107 3108 3109 3110 3111 3112 3113 3114 3115 3116 3117 3118 3119 3120 3121 3122 3123 3124 3125 3126 3127 3128 3129 3130 3131 3132 3133 3134 3135 3136 3137 3138 3139 3140 3141 3142 3143 3144 3145 3146 3147 3148 3149 3150 3151 3152 3153 3154 3155 3156 3157 3158 3159 3160 3161 3162 3163 3164 3165 3166 3167 3168 3169 3170 3171 3172 3173 3174 3175 3176 3177 3178 3179 3180 3181 3182 3183 3184 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 3223 3224 3225 3226 3227 3228 3229 3230 3231 3232 3233 3234 3235 3236 3237 3238 3239 3240 3241 3242 3243 3244 3245 3246 3247 3248 3249 3250 3251 3252 3253 3254 3255 3256 3257 3258 3259 3260 3261 3262 3263 3264 3265 3266 3267 3268 3269 3270 3271 3272 3273 3274 3275 3276 3277 3278 3279 3280 3281 3282 3283 3284 3285 3286 3287 3288 3289 3290 3291 3292 3293 3294 3295 3296 3297 3298 3299 3300 3301 3302 3303 3304 3305 3306 3307 3308 3309 3310 3311 3312 3313 3314 3315 3316 3317 3318 3319 3320 3321 3322 3323 3324 3325 3326 3327 3328 3329 3330 3331 3332 3333 3334 3335 3336 3337 3338 3339 3340 3341 3342 3343 3344 3345 3346 3347 3348 3349 3350 3351 3352 3353 3354 3355 3356 3357 3358 3359 3360 3361 3362 3363 3364 3365 3366 3367 3368 3369 3370 3371 3372 3373 3374 3375 3376 3377 3378 3379 3380 3381 3382 3383 3384 3385 3386 3387 3388 3389 3390 3391 3392 3393 3394 3395 3396 3397 3398 3399 3400 3401 3402 3403 3404 3405 3406 3407 3408 3409 3410 3411 3412 3413 3414 3415 3416 3417 3418 3419 3420 3421 3422 3423 3424 3425 3426 3427 3428 3429 3430 3431 3432 3433 3434 3435 3436 3437 3438 3439 344

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

6.0 ERROR REPORTING

6.1 ERROR COMMENT

ALL ERRORS ARE ACCOMPANIED WITH AN ENGLISH LANGUAGE DESCRIPTIVE COMMENT AS TO THE TYPE OF FAILURE. FURTHER QUALIFICATION OF THE ERROR CAN BE OBTAINED IF NEEDED FROM THE COMMENT AT THE ERROR PC OF THE LISTING.

6.2 ERROR DATA

PC LISTING ADDRESS WHERE THE ERROR WAS DETECTED.
TSTNUM TEST NUMBER WHERE THE ERROR OCCURRED.
BUSADRS DISPLAY BUS ADDRESS WHERE THE RESULTANT DATA WAS EXPECTED
EXPCY DATA THAT WAS EXPECTED.
RCVD DATA THAT WAS RECEIVED.

7.0 MISCELLANEOUS

7.1 VS60 BUS/VECTOR ADDRESS MODIFICATION

MODIFY LOCATION 'SVECT!' IF BASE VECTOR ADDRESS IS NOT 320.
MODIFY LOCATION 'SBSAE' IF BASE BUS ADDRESS IS NOT 172000.

NOTE: A RESTART IS REQUIRED AFTER THE ABOVE ADDRESS MODIFICATION.
THE ABOVE LOCATIONS ARE LOCATED IN THE 'APT MAILBOX-ETABLE'

7.2 XXDP/APT NOTES

THIS DIAGNOSTIC IS CHAINABLE UNDER XXDP.
THIS DIAGNOSTIC INCLUDES THE 'APT' SOFTWARE HOOKS HOWEVER, THEY HAVE NOT BEEN TESTED.

7.3 POWER FAIL

A POWER FAILURE WILL CAUSE A RESTART MESSAGE ON POWER UP AT WHICH TIME THE PROGRAM IS RESTARTED AT THE BEGINNING.

7.4 SINGLE VS60 TESTING

THIS DIAGNOSTIC DOES NOT TEST MULTIPLE VS60'S.

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

8.0 EXECUTION TIME

EXECUTION TIME RANGES FROM ABOUT 3 SECONDS WITH NO ITERATIONS TO ABOUT 2 MINUTES WITH ITERATIONS ENABLED. AN 'END PASS' MESSAGE IS TYPED EACH PASS THRU THE DIAGNOSTIC.

9.0 PROGRAM TEST DESCRIPTIONS

9.1 GENERAL

THE TABLE OF CONTENTS AT THE BEGINNING OF THE LISTING CONTAINS A BRIEF DESCRIPTION OF EACH TEST. IT WILL ALSO IDENTIFY WHAT MAINTENANCE MODE A PARTICULAR TEST USES. THE COMMENT FIELD IN EACH TEST ALSO CAN BE HELPFUL IN UNDERSTANDING A TEST.

9.2 TYPICAL TEST SEQUENCE

MOST TESTS ARE ORGANIZED AROUND THE FOLLOWING FORMAT: A MAINTENANCE MODE SWITCH IS SET (SW1, 2 OR 3). THEN A VS60 REGISTER BUS REGISTER ADDRESS IS LOADED INTO LOCATION '\$BDADR' WHICH IS WHERE THE EXPECTED DATA WILL BE READ. NEXT STEP WILL BE TO LOAD THE EXPECTED DATA INTO LOCATION '\$GDDAT'. THEN ONE OR MORE INSTRUCTIONS ARE LOADED STARTING AT LOCATION 'BUFFER'. THE DISPLAY PC IS THEN LOADED WITH ADDRESS 'BUFFER' WHICH CAUSES AN NPR TO 'BUFFER'. AT THE COMPLETION OF THE 'NPR' A NUMBER OF RESUMES COULD BE ISSUED IF ADDITIONAL 'NPRS' ARE REQUIRED FOR INSTRUCTIONS OR DATA. AT THIS TIME THE CONDITION UNDER TEST SHOULD HAVE BEEN SET UP AND AVAILABLE AT THE BUS ADDRESS SET UP IN '\$BDADR'. THIS CONDITION IS USUALLY LOADED INTO LOCATION '\$BDDAT' AND COMPARED TO THE EXPECTED RESULTS PREVIOUSLY SET UP IN '\$GDDAT'. IF AN ERROR IS DETECTED THEN AN ERROR TRAP (EMT) IS PERFORMED ACCORDING TO THE ITEM NUMBER IN THE '\$ERRTB' (ERROR TABLE). THIS TABLE DIRECTS OR POINTS TO THE PERTINENT 'TYPE OUT' INFORMATION.

10.0 LISTING

```

%
.TITLE MAINDEC-11-DZVSA-A VS60 INSTRUCTION TEST PART 1
.*COPYRIGHT (C) 1976
.*DIGITAL EQUIPMENT CORP.
.*MAYNARD, MASS. 01754
.*
.*PROGRAM BY R. MOORE
.*
.*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
.*PACKAGE (MAINDEC-11-DZQAC-B1),AUG 19,1975.
.*
.SBTTL BASIC DEFINITIONS

.*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
STACK= 1100
.EQUIV EMT,ERROR ;;BASIC DEFINITION OF ERROR CALL
.EQUIV IOT,SCOPE ;;BASIC DEFINITION OF SCOPE CALL
PS= 177776 ;;PROCESSOR STATUS WORD
    
```

001100

177776

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

177774
177772
177570
177570

.EQUIV PS,PSW
STKLMT= 177774
PIRQ= 177772
DSWR= 177570
DDISP= 177570

::: STACK LIMIT REGISTER
::: PROGRAM INTERRUPT REQUEST REGISTER
::: HARDWARE SWITCH REGISTER
::: HARDWARE DISPLAY REGISTER

::: *GENERAL PURPOSE REGISTER DEFINITIONS

000000
000001
000002
000003
000004
000005
000006
000007

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

::: GENERAL REGISTER
::: GENERAL REGISTER
::: GENERAL REGISTER
::: GENERAL REGISTER
::: GENERAL REGISTER
::: GENERAL REGISTER
::: GENERAL REGISTER
::: GENERAL REGISTER

.EQUIV R6,SP
.EQUIV R7,PC

::: STACK POINTER
::: PROGRAM COUNTER

::: *PRIORITY LEVEL DEFINITIONS

000000
000040
000100
000140
000200
000240
000300
000340

PR0= 0
PR1= 40
PR2= 100
PR3= 140
PR4= 200
PR5= 240
PR6= 300
PR7= 340

::: PRIORITY LEVEL 0
::: PRIORITY LEVEL 1
::: PRIORITY LEVEL 2
::: PRIORITY LEVEL 3
::: PRIORITY LEVEL 4
::: PRIORITY LEVEL 5
::: PRIORITY LEVEL 6
::: PRIORITY LEVEL 7

::: *"SWITCH REGISTER" SWITCH DEFINITIONS

100000
040000
020000
010000
004000
002000
001000
000400
000200
000100
000040
000020
000010
000004
000002
000001

SW15= 100000
SW14= 40000
SW13= 20000
SW12= 10000
SW11= 4000
SW10= 2000
SW09= 1000
SW08= 400
SW07= 200
SW06= 100
SW05= 40
SW04= 20
SW03= 10
SW02= 4
SW01= 2
SW00= 1

.EQUIV SW09,SW9
.EQUIV SW08,SW8
.EQUIV SW07,SW7
.EQUIV SW06,SW6
.EQUIV SW05,SW5
.EQUIV SW04,SW4
.EQUIV SW03,SW3
.EQUIV SW02,SW2
.EQUIV SW01,SW1
.EQUIV SW00,SW0

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

.SBTTL OPERATIONAL SWITCH SETTINGS

```

*
* SWITCH USE
* -----
*      15 HALT ON ERROR
*      14 LOOP ON TEST
*      13 INHIBIT ERROR TYPEOUTS
*      12 HALT AT END OF DIAGNOSTIC
*      11 INHIBIT ITERATIONS
*      10 BELL ON ERROR
*      9  LOOP ON ERROR
*      8  LOOP ON TEST IN SWR<7:0>

```

.SBTTL TRAP CATCHER

```

000000
.=0
;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
.=174
DISPREG: .WORD 0      ;;SOFTWARE DISPLAY REGISTER
SWREG:   .WORD 0      ;;SOFTWARE SWITCH REGISTER

```

.SBTTL STARTING ADDRESS(ES)

```

000200 000137 002144
JMP     @#BEGIN      ;;JUMP TO STARTING ADDRESS OF PROGRAM
;*****

```

.SBTTL ACT11 HOOKS

```

;HOOKS REQUIRED BY ACT11
$SVPC=.              ;SAVE PC
.=46
$ENDAD              ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .SEOP
.=52
.WORD 0              ;;2)SET LOC.52 TO ZERO
.$SVPC              ;; RESTORE PC
.=1000

```

;*****

.SBTTL APT PARAMETER BLOCK

```

;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
;*****
.$X=.                ;;SAVE CURRENT LOCATION
.=24                 ;;SET POWER FAIL TO POINT TO START OF PROGRAM
200                  ;FOR APT START UP
.=44                 ;POINT TO APT INDIRECT ADDRESS PNTR.
$APTHDR              ;POINT TO APT HEADER BLOCK
.=.$X                ;RESET LOCATION COUNTER

```

```

;*****
;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
;INTERFACE SPEC.

```

```

001000 $APTHD:
001000 $HIBTS: .WORD 0      ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
000000

```

JO1

MAINDEC-11-DZVSA-A VS60 INSTRUCTION TEST PART 1
DZVSA.P11 APT PARAMETER BLOCK

MACY11 27(732) 20-SEP-76 09:57 PAGE 9

391 001002 001200
392 001004 C JO12
393 001006 000036
394 001010 000000
395 001012 000052
396
397
398
399
400 000204 000204
401 000137 002506

\$MBADR: .WORD \$MAIL ;; ADDRESS OF APT MAILBOX (BITS 0-15)
\$TSTM: .WORD 10. ;; RUN TIM OF LONGEST TEST
\$PASTM: .WORD 30. ;; RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
\$UNITM: .WORD 0 ;; ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
.WORD \$ETEND-\$MAIL/2 ;; LENGTH MAILBOX-ETABLE(WORDS)

;RESTART ADDRESS

.=204
JMP @#RESTR ;RESTART GOOD AFTER START 200

```

402
403
404
405
406
407
408
409      001100
410      001100 000000
411      001100 000000
412      001102 000
413      001103 000
414      001104 000000
415      001106 000000
416      001110 000000
417      001112 000000
418      001114 000
419      001115 001
420      001116 000000
421      001120 000000
422      001122 000000
423      001124 000000
424      001126 000000
425      001130 000000
426      001132 000000
427      001134 000000
428      001136 177570
429      001140 177570
430      001142 177560
431      001144 177562
432      001146 177564
433      001150 177566
434      001152 000
435      001153 002
436      001154 012
437      001155 000
438      001156 000000
439
440      001160 000000
441      001162 000000
442      001164 000000
443      001166 000000
444      001170 177607 000377
445      001174 077
446      001175 015
447      001176 000012

```

```

;*****
.SBTTL COMMON TAGS
;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
;*USED IN THE PROGRAM.

$CMTAG:      .=1100      ;;START OF COMMON TAGS

$TSTNM:      .WORD      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
$GDADR:      .WORD      0      ;;CONTAINS 'GOOD' DATA
$BDADR:      .WORD      0      ;;CONTAINS 'BAD' DATA
$RESV1:      .WORD      0      ;;RESERVED--NOT TO BE USED
$RESV2:      .WORD      0
$RESV3:      .WORD      0
$RESV4:      .WORD      0
$RESV5:      .WORD      0
$RESV6:      .WORD      0
$RESV7:      .WORD      0
$RESV8:      .WORD      0
$RESV9:      .WORD      0
$RESV10:     .WORD      0
$RESV11:     .WORD      0
$RESV12:     .WORD      0
$RESV13:     .WORD      0
$RESV14:     .WORD      0
$RESV15:     .WORD      0
$RESV16:     .WORD      0
$RESV17:     .WORD      0
$RESV18:     .WORD      0
$RESV19:     .WORD      0
$RESV20:     .WORD      0
$RESV21:     .WORD      0
$RESV22:     .WORD      0
$RESV23:     .WORD      0
$RESV24:     .WORD      0
$RESV25:     .WORD      0
$RESV26:     .WORD      0
$RESV27:     .WORD      0
$RESV28:     .WORD      0
$RESV29:     .WORD      0
$RESV30:     .WORD      0
$RESV31:     .WORD      0
$RESV32:     .WORD      0
$RESV33:     .WORD      0
$RESV34:     .WORD      0
$RESV35:     .WORD      0
$RESV36:     .WORD      0
$RESV37:     .WORD      0
$RESV38:     .WORD      0
$RESV39:     .WORD      0
$RESV40:     .WORD      0
$RESV41:     .WORD      0
$RESV42:     .WORD      0
$RESV43:     .WORD      0
$RESV44:     .WORD      0
$RESV45:     .WORD      0
$RESV46:     .WORD      0
$RESV47:     .WORD      0
$RESV48:     .WORD      0
$RESV49:     .WORD      0
$RESV50:     .WORD      0
$RESV51:     .WORD      0
$RESV52:     .WORD      0
$RESV53:     .WORD      0
$RESV54:     .WORD      0
$RESV55:     .WORD      0
$RESV56:     .WORD      0
$RESV57:     .WORD      0
$RESV58:     .WORD      0
$RESV59:     .WORD      0
$RESV60:     .WORD      0
$RESV61:     .WORD      0
$RESV62:     .WORD      0
$RESV63:     .WORD      0
$RESV64:     .WORD      0
$RESV65:     .WORD      0
$RESV66:     .WORD      0
$RESV67:     .WORD      0
$RESV68:     .WORD      0
$RESV69:     .WORD      0
$RESV70:     .WORD      0
$RESV71:     .WORD      0
$RESV72:     .WORD      0
$RESV73:     .WORD      0
$RESV74:     .WORD      0
$RESV75:     .WORD      0
$RESV76:     .WORD      0
$RESV77:     .WORD      0
$RESV78:     .WORD      0
$RESV79:     .WORD      0
$RESV80:     .WORD      0
$RESV81:     .WORD      0
$RESV82:     .WORD      0
$RESV83:     .WORD      0
$RESV84:     .WORD      0
$RESV85:     .WORD      0
$RESV86:     .WORD      0
$RESV87:     .WORD      0
$RESV88:     .WORD      0
$RESV89:     .WORD      0
$RESV90:     .WORD      0
$RESV91:     .WORD      0
$RESV92:     .WORD      0
$RESV93:     .WORD      0
$RESV94:     .WORD      0
$RESV95:     .WORD      0
$RESV96:     .WORD      0
$RESV97:     .WORD      0
$RESV98:     .WORD      0
$RESV99:     .WORD      0
$RESV100:    .WORD      0

$SWR:        .WORD      DSWR      ;;ADDRESS OF SWITCH REGISTER
$DISP:       .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"
$STPFLG:     .BYTE      0      ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
$REGAD:      .WORD      0      ;;CONTAINS THE ADDRESS FROM WHICH ($REGO) WAS OBTAINED
$REGO:       .WORD      0      ;;CONTAINS (($REGAD)+0)
$REG1:       .WORD      0      ;;CONTAINS (($REGAD)+2)
$TIMES:      0          ;;MAX. NUMBER OF ITERATIONS
$ESCAPE:     0          ;;ESCAPE ON ERROR ADDRESS
$BELL:       .ASCIZ    <207><377><377> ;;CODE FOR BELL
$QUES:       .ASCII    /?/      ;;QUESTION MARK
$CRLF:       .ASCII    <15>     ;;CARRIAGE RETURN
$LF:         .ASCIZ    <12>     ;;LINE FEED

```

24

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

.SBTTL APT MAILBOX-ETABLE

.EVEN

001200
001200 000000
001202 000000
001204 000000
001206 000000
001210 000000
001212 000000
001214 000000
001216 000000
001220
001220 000
001221 000
001222 000000
001224 000000
001226 000000

001230 000
001231 000

001232 000000

001234 000
001235 000
001236 000000
001240 000
001241 000
001242 000000
001244 000
001245 000
001246 000000
001250 320
001251 000
001252 004
001253 000

001254 172000
001256 000000
001260 000000
001262 000000
001264 000000

\$MAIL: .WORD
\$MSGTY: .WORD
\$FATAL: .WORD
\$TESTN: .WORD
\$PASS: .WORD
\$DEVCT: .WORD
\$UNIT: .WORD
\$MSGAD: .WORD
\$MSGLG: .WORD
\$ETABLE:
\$ENV: .BYTE
\$ENVM: .BYTE
\$SWREG: .WORD
\$USWR: .WORD
\$CPUOP: .WORD

\$MAMS1: .BYTE
\$MTYP1: .BYTE

\$MADR1: .WORD

\$MAMS2: .BYTE
\$MTYP2: .BYTE
\$MADR2: .WORD
\$MAMS3: .BYTE
\$MTYP3: .BYTE
\$MADR3: .WORD
\$MAMS4: .BYTE
\$MTYP4: .BYTE
\$MADR4: .WORD
\$VECT1: .BYTE
\$VECT2: .BYTE
\$PRIOR: .BYTE

\$BASE: .WORD
\$DEVN: .WORD
\$CDW1: .WORD
\$CDW2: .WORD
\$DDWO: .WORD

APT MAILBOX
MESSAGE TYPE CODE
FATAL ERROR NUMBER
TEST NUMBER
PASS COUNT
DEVICE COUNT
I/O UNIT NUMBER
MESSAGE ADDRESS
MESSAGE LENGTH
APT ENVIRONMENT TABLE
ENVIRONMENT BYTE
ENVIRONMENT MODE BITS
APT SWITCH REGISTER
USER SWITCHES
CPU TYPE, OPTIONS
BITS 15-11=CPU TYPE
11/04=01, 11/05=02, 11/20=03, 11/40=04, 11/45=05
11/70=06, PDQ=07, Q=10
BIT 10=REAL TIME CLOCK
BIT 9=FLOATING POINT PROCESSOR
BIT 8=MEMORY MANAGEMENT
HIGH ADDRESS, M.S. BYTE
MEM. TYPE, BLK#1
MEM. TYPE BYTE -- (HIGH BYTE)
900 NSEC CORE=001
300 NSEC BIPOLAR=002
500 NSEC MOS=003
HIGH ADDRESS, BLK#1
MEM. LAST ADDR.=3 BYTES, THIS WORD AND LOW OF "TYPE" ABOVE
HIGH ADDRESS, M.S. BYTE
MEM. TYPE, BLK#2
MEM. LAST ADDRESS, BLK#2
HIGH ADDRESS, M.S. BYTE
MEM. TYPE, BLK#3
MEM. LAST ADDRESS, BLK#3
HIGH ADDRESS, M.S. BYTE
MEM. TYPE, BLK#4
MEM. LAST ADDRESS, BLK#4
INTERRUPT VECTOR#1
INTERRUPT VECTOR#2
BUS PRIORITY #1, #2
SPARE, NOT USED

BASE ADDRESS OF EQUIPMENT UNDER TEST
DEVICE MAP
CONTROLLER DESCRIPTION WORD#1
CONTROLLER DESCRIPTION WORD#2
DEVICE DESCRIPTOR WORD#0

504 001266 000000
 505 001270 000000
 506 001272 000000
 507 001274 000000
 508 001276 000000
 509 001300 000000
 510 001302 000000
 511 001304 000000
 512 001306 000000
 513 001310 000000
 514 001312 000000
 515 001314 000000
 516 001316 000000
 517 001320 000000
 518 001322 000000
 519
 520
 521 001324
 522
 523
 524
 525
 526
 527
 528
 529
 530
 531
 532
 533
 534
 535
 536
 537
 538 001324
 539
 540
 541
 542 001324 025646
 543 001326 027444
 544 001330 027512
 545 001332 000000
 546
 547
 548
 549 001334 025670
 550 001336 027444
 551 001340 027512
 552 001342 000000
 553
 554
 555
 556 001344 025724
 557 001346 027444
 558 001350 027512
 559 001352 000000

\$DDW1: .WORD ADDW1 ;; DEVICE DESCRIPTOR WORD#1
 \$DDW2: .WORD ADDW2 ;; DEVICE DESCRIPTOR WORD#2
 \$DDW3: .WORD ADDW3 ;; DEVICE DESCRIPTOR WORD#3
 \$DDW4: .WORD ADDW4 ;; DEVICE DESCRIPTOR WORD#4
 \$DDW5: .WORD ADDW5 ;; DEVICE DESCRIPTOR WORD#5
 \$DDW6: .WORD ADDW6 ;; DEVICE DESCRIPTOR WORD#6
 \$DDW7: .WORD ADDW7 ;; DEVICE DESCRIPTOR WORD#7
 \$DDW8: .WORD ADDW8 ;; DEVICE DESCRIPTOR WORD#8
 \$DDW9: .WORD ADDW9 ;; DEVICE DESCRIPTOR WORD#9
 \$DDW10: .WORD ADDW10 ;; DEVICE DESCRIPTOR WORD#10
 \$DDW11: .WORD ADDW11 ;; DEVICE DESCRIPTOR WORD#11
 \$DDW12: .WORD ADDW12 ;; DEVICE DESCRIPTOR WORD#12
 \$DDW13: .WORD ADDW13 ;; DEVICE DESCRIPTOR WORD#13
 \$DDW14: .WORD ADDW14 ;; DEVICE DESCRIPTOR WORD#14
 \$DDW15: .WORD ADDW15 ;; DEVICE DESCRIPTOR WORD#15

\$SETEND:

.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.
 ;; *NOTE: 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:

; ITEM 1

EM1 ;; MAINT. SWITCH REGISTER
 DH1 ;; ERRPC TSTNUM BUSADR EXPCT RCVD
 DT1 ;; \$ERRPC TSTNUM \$BDADR \$GDDAT \$BDDAT
 0

; ITEM 2

EM2 ;; LOADING D.P.C. REGISTER <STATIC>
 DH1 ;; ERRPC TSTNUM BUSADR EXPCT RCVD
 DT1 ;; \$ERRPC TSTNUM \$BDADR \$GDDAT \$BDDAT
 0

; ITEM 3

EM3 ;; LOADING MODE REGISTER
 DH1 ;; ERRPC TSTNUM BUSADR EXPCT RCVD
 DT1 ;; \$ERRPC TSTNUM \$BDADR \$GDDAT \$BDDAT
 0

560
561
562
563
564
565
566
567
568
569
570
571
572
573

001354 025745
001356 027444
001360 027512
001362 000000

001364 025770
001366 027444
001370 027512
001372 000000

;ITEM 4

EM4
DH1
DT1
0

; SEQUENCING DPC REGISTER
;ERRPC TSTNUM BUSADR EXPCT RCVD
;SERRPC TSTNUM \$BODR \$GDDAT \$BDDAT

;ITEM 5

EM5
DH1
DT1
0

; LOADING LINE TYPE REGISTER
;ERRPC TSTNUM BUSADR EXPCT RCVD
;SERRPC TSTNUM \$BODR \$GDDAT \$BDDAT

001404	026016	EM6	:LOADING INTENSITY LEVEL REGISTER
001406	027444	DH1	:ERRPC TSTNUM BUSADR EXPCT RCVD
001408	027512	DT1	:SERRPC TSTNUM SBDADR SGDDAT SBDDAT
00140A	000000	0	
:ITEM 6			
00140C	026052	EM7	:LOADING GRAPHLOT INCREMENT REGISTER
00140E	027444	DH1	:ERRPC TSTNUM BUSADR EXPCT RCVD
001410	027512	DT1	:SERRPC TSTNUM SBDADR SGDDAT SBDDAT
001412	000000	0	
:ITEM 7			
001414	026104	EM10	:DISPLAY JUMP ABSOLUTE TO AN ADDRESS
001416	027444	DH1	:ERRPC TSTNUM BUSADR EXPCT RCVD
001418	027512	DT1	:SERRPC TSTNUM SBDADR SGDDAT SBDDAT
00141A	000000	0	
:ITEM 10			
00141C	026140	EM11	:DISPLAY JUMP RELATIVE TO AN ADDRESS
00141E	027444	DH1	:ERRPC TSTNUM BUSADR EXPCT RCVD
001420	027512	DT1	:SERRPC TSTNUM SBDADR SGDDAT SBDDAT
001422	000000	0	
:ITEM 11			
001424	026174	EM12	:DISPLAY JSR ABSOLUTE TO AN ADDRESS
001426	027444	DH1	:ERRPC TSTNUM BUSADR EXPCT RCVD
001428	027512	DT1	:SERRPC TSTNUM SBDADR SGDDAT SBDDAT
00142A	000000	0	
:ITEM 12			
00142C	026227	EM13	:DISPLAY JSR RELATIVE TO AN ADDRESS
00142E	027444	DH1	:ERRPC TSTNUM BUSADR EXPCT RCVD
001430	027512	DT1	:SERRPC TSTNUM SBDADR SGDDAT SBDDAT
001432	000000	0	
:ITEM 13			
001434	026262	EM14	:LOADING X POSITION REGISTER
001436	027444	DH1	:ERRPC TSTNUM BUSADR EXPCT RCVD
001438	027512	DT1	:SERRPC TSTNUM SBDADR SGDDAT SBDDAT
00143A	000000	0	
:ITEM 14			

769			:ITEM	42				
770								
771	001734	027161		EM42		:L.P. INTR ENABLE - CONSOLE O/I ERROR		
772	001736	027444		DH1		:ERRPC	TSTNUM	BUSADR EXPCT RCVD
773	001740	027512		DT1		:SERRPC	TSTNUM	\$BDADR \$GDDAT \$BDDAT
774	001742	000000		0				
775								
776			:ITEM	43				
777								
778	001744	027222		EM43		:L.P. SWITCH INTR ENABLE - CONSOLE O/I ERROR		
779	001746	027444		DH1		:ERRPC	TSTNUM	BUSADR EXPCT RCVD
780	001750	027512		DT1		:SERRPC	TSTNUM	\$BDADR \$GDDAT \$BDDAT
781	001752	000000		0				
782								
783			:ITEM	44				
784								
785	001754	027272		EM44		:DISPLAY BUSY ERROR		
786	001756	027444		DH1		:ERRPC	TSTNUM	BUSADR EXPCT RCVD
787	001760	027512		DT1		:SERRPC	TSTNUM	\$BDADR \$GDDAT \$BDDAT
788	001762	000000		0				
789								
790			:ITEM	45				
791	001764	027312		EM45		:EXTERNAL STOP LOGIC ERROR		
792	001766	027444		DH1		:ERRPC	TSTNUM	BUSADR EXPCT RCVD
793	001770	027512		DT1		:SERRPC	TSTNUM	\$BDADR \$GDDAT \$BDDAT
794	001772	000000		0				
795								
796			:ITEM	46				
797								
798	001774	027341		EM46		:TIME-OUT INTERRUPT ERROR		
799	001776	027444		DH1		:ERRPC	TSTNUM	BUSADR EXPCT RCVD
800	002000	027512		DT1		:SERRPC	TSTNUM	\$BDADR \$GDDAT \$BDDAT
801	002002	000000		0				
802								
803			:ITEM	47				
804								
805	002004	027362		EM47		:NAME MATCH INTERRUPT ERROR		
806	002006	027444		DH1		:ERRPC	TSTNUM	BUSADR EXPCT RCVD
807	002010	027512		DT1		:SERRPC	TSTNUM	\$BDADR \$GDDAT \$BDDAT
808	002012	000000		0				
809								
810			:ITEM	50				
811								
812	002014	027405		EM50		:L.P. FLAG INTERRUPT ERROR		
813	002016	027444		DH1		:ERRPC	TSTNUM	BUSADR EXPCT RCVD
814	002020	027512		DT1		:SERRPC	TSTNUM	\$BDADR \$GDDAT \$BDDAT
815	002022	000000		0				
816								
817			:ITEM	51				
818								
819	002024	027427		EM51		:STACK PTR ER		
820	002026	027444		DH1		:ERRPC	TSTNUM	BUSADR EXPCT RCVD
821	002030	027512		DT1		:SERRPC	TSTNUM	\$BDADR \$GDDAT \$BDDAT
822	002032	000000		0				

```

823
824 ;COMMON PROGRAM TAGS AND STORAGE LOCATIONS
825
826 002034 027526 DBUF: BUFFER ;FIRST WORD IN THE DISPLAY BUFFER
827 002036 027530 DBUF1: BUFFER+2 ;SECOND WORD
828 002040 027532 DBUF2: BUFFER+4 ;THIRD WORD
829 002042 027534 DBUF3: BUFFER+6 ;FOURTH WORD
830 002044 027536 DBUF4: BUFFER+10 ;FIFTH WORD
831 002046 027540 DBUF5: BUFFER+12 ;SIXTH WORD
832 002050 000000 DSAVE: 0 ;TEMP REG.
833 002052 000000 DSAVE1: 0
834 002054 000000 SIZE: 0 ;BUFFER SIZE
835 002056 000000 CNTR: 0
836 002060 000000 TSTNUM: 0 ;CONTAINS TEST NO. ON ER
837 002062 000020 DELAY: 20 ;COUNT USED FOR GENERAL PURPOSE DELAYS
838
839 ;VS60 ADDRESSES AND VECTORS POINTERS
840
841 002064 172000 DPC: 172000 ;DISPLAY PC REGISTER
842 002066 172002 SREG0: 172002 ;DISPLAY STATUS REGISTER
843 002070 172004 XPOS: 172004 ;X AXIS REGISTER AND GRAPHLOT REG <READ ONLY>
844 002072 172006 YPOS: 172006 ;Y AXIS REGISTER AND CHARACTER REG <READ ONLY>
845 002074 172010 RLO: 172010 ;RELOCATION REGISTER
846 002076 172012 SREG1: 172012 ;MISC STATUS REGISTER #1
847 002100 172014 XDOFF: 172014 ;X DYNAMIC OFFSET REGISTER
848 002102 172016 YDOFF: 172016 ;Y DYNAMIC OFFSET REGISTER
849 002104 172020 ANAME: 172020 ;ASSOCIATIVE NAME REGISTER
850 002106 172022 CONS: 172022 ;CONSOLE STATUS REGISTER
851 002110 172024 DNAME: 172024 ;DISPLAY NAME REGISTER
852 002112 172026 STKVAL: 172026 ;VALUE OF CURRENT STACK WORD
853 002114 172030 TERMCH: 172030 ;TERMINATE CHARACTER REGISTER
854 002116 172032 STKPT: 172032 ;STACK POINTER REGISTER
855 002120 172034 ZPOS: 172034 ;Z POSITION REGISTER <READ ONLY>
856 002122 172036 ZDOFF: 172036 ;Z DYNAMIC OFFSET REGISTER
857
858 002124 000320 DDONE: 320 ;DISPLAY STOP <DONE> VECTOR
859 002126 000322 DDCNE1: 322
860
861 002130 000324 LPVCT: 324 ;DISPLAY LIGHT PEN VECTOR
862 002132 000326 LPVCT1: 326
863
864 002134 000330 TIMEVT: 330 ;DISPLAY TIME-OUT <NXM.> ERROR VECTOR
865 002136 000332 TMEVT1: 332 ;OR "SHIFT-OUT" VECTOR
866
867 002140 000334 NAMEVT: 334 ;NAME MATCH VECTOR
868 002142 000336 NAMEV1: 336
    
```

```

859                                     ;SOFTWARE INITIALIZATION ROUTINE
870
871 002144                                     BEGIN:
872 002144 012706 001100      MOV      #SCMTAG,R6          ;;FIRST LOCATION TO BE CLEARED
873 002150 005026          CLR      (R6)+          ;;CLEAR MEMORY LOCATION
874 002152 022706 001126      CMP      #SBDDAT,R6      ;;DONE?
875 002156 001374          BNE     -6              ;;LOOP BACK IF NO
876 002160 012706 001100      MOV      #STACK,SP      ;;SETUP THE STACK POINTER
877 002164 012737 023202 000020      MOV      #SCOPE, @IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
878 002172 012737 000340 000022      MOV      #340, @IOTVEC+2 ;;LEVEL 7
879 002200 012737 023726 000030      MOV      #ERROR, @EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
880 002206 012737 000340 000032      MOV      #340, @EMTVEC+2 ;;LEVEL 7
881 002214 012737 025520 000034      MOV      #TRAP, @TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
882 002222 012737 000340 000036      MOV      #340, @TRAPVEC+2 ;;LEVEL 7
883 002230 012737 024736 000024      MOV      #SPWRDN, @PWRVEC ;;POWER FAILURE VECTOR
884 002236 012737 000340 000026      MOV      #340, @PWRVEC+2 ;;LEVEL 7
885 002244 013737 023046 023040      MOV      $ENDCT, $EOPCT ;;SETUP END-OF-PROGRAM COUNTER
886 002252 005037 001164          CLR      $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
887 002256 005037 001166          CLR      $ESCAPE        ;;CLEAR THE ESCAPE ON ERROR ADDRESS
888 002262 112737 000001 001115      MOV      #1, $ERMAX      ;;ALLOW ONE ERROR PER TEST
889 002270 012737 002270 001106      MOV      #, $LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
890 002276 012737 002276 001110      MOV      #, $LPERR       ;;SETUP THE ERROR LOOP ADDRESS
891 002304 013746 000004          MOV      @#4, -(SP)      ;;SAVE ERROR VECTOR
892 002310 013746 000006          MOV      @#6, -(SP)
893 002314 012737 002330 000004      MOV      #15, 4          ;;SET UP TIMEOUT VECTOR
894 002322 005777 176610          TST     $SWR             ;;TRY TO REFERENCE HARDWARE SWR
895 002326 000407          BR     2$              ;;BRANCH IF NO TIMEOUT TRAP OCCURS
896 002330 012737 000176 001136 1$: MOV      #SWREG, SWR      ;;POINT TO SOFTWARE SWITCH REG.
897 002336 012737 000174 001140      MOV      #DISPREG, DISPLAY ;;POINT TO SOFTWARE DISPLAY REG.
898 002344 022626          CMP      (SP)+, (SP)+   ;;RESTORE STACK
899 002346 012637 000006 2$: MOV      (SP)+, @#6      ;;RESTORE ERROR VECTOR
900 002352 012637 000004          MOV      (SP)+, @#4
901 002356 005037 001206          CLR      $PASS          ;;CLEAR PASS COUNT
902 002362 132737 000200 001221      BITB   #APTSIZE, $ENVM   ;;TEST USER SIZE UNDER APT
903 002370 001403          BEQ     3$              ;;YES, USE NON-APT SWITCH
904 002372 012737 001222 001136      MOV      #SSWREG, SWR   ;;NO, USE APT SWITCH REGISTER
905 002400
906 002400 012700 002064      SETUP1: MOV      #DPC, R0   ;;SET UP VS60 REG ADRS POINTERS
907 002404 013701 001254      MOV      $BASE, R1      ;;GET VS60 BASE ADRS FROM APT MAILBOX-ETABLE
908 002410 010120      SETUP2: MOV      R1, (0)+
909 002412 062701 000002      ADD     #2, R1
910 002416 022700 002124      CMP     #DDONE, R0      ;;TEST FOR LAST ADDRESS
911 002422 001372          BNE     SETUP2
912 002424 012700 002124      MOV     #DDONE, R0      ;;SET UP VS60 VECTOR ADRS POINTERS
913 002430 013701 001250      MOV     $VECT1, R1      ;;GET VS60 BASE VECTOR ADRS FROM APT MAILBOX-ETABLE
914 002434 010120      SETUP3: MOV      R1, (0)+
915 002436 062701 000002      ADD     #2, R1
916 002442 022700 002144      CMP     #BEGIN, R0      ;;TEST FOR LAST VECTOR
917 002446 001372          BNE     SETUP3
918 002450 004737 023134      JSR     PC, RSTVEC      ;;GO SET UP VS60 VECTOR ADR'S WITH HL'S
919 002454 004737 025140      JSR     PC, $SIZE       ;;DETERMINE LAST MEM LOCATION
920 002460 013737 025232 002054      MOV     $LSTAD, SIZE    ;;LOAD LAST MEM ADDRESS
921 002466 162737 006000 002054      SUB     #6000, $SIZE     ;;SAVE LOADERS AND XXDP
922 002474 005737 000042          TST     @#42            ;;TEST IF RUNNING CHAIN MODE UNDER XXDP
923 002500 001002          BNE     RESTRT         ;;BR IF SO
924 002502 104400          TYPE

```



```

930 ;*****
931 ;*TEST 1 SEE IF ALL VS60 REGS ARE THERE
932 ;*****
933 002526 000004          †ST1: SCOPE
934 002530 012737 002564 001110      MOV #15,$LPERR ;SET UP SCOPE LOOP ADRS
935 002536 012737 002604 000004      MOV #25,2#ERRVEC ;SET UP TIME OUT TRAP
936 002544 012737 000340 000006      MOV #340,2#ERRVEC+2 ;SET UP TIME OUT PRIORITY
937 002552 013737 001254 001122      MOV $BASE,$BDADR ;PUT 1ST VS60 BUS ADRS IN LOC $BDADR
938 002560 012700 000020             MOV #20,R0 ;SET UP REG COUNT
939 002564 005777 176332             1$: TST 2$BDADR ;LOOK FOR VS60 REG
940 002570 062737 000002 001122      ADD #2,$BDADR ;OK-ADVANCE TO NEXT
941 002576 005300             DEC R0 ;HAVE WE TRIED ALL REGS?
942 002600 001371             BNE 1$ ;BR IF NOT
943 002602 000402             BR 3$ ;GO RESTORE LOCS 4 & 6
944 002604 022626             2$: CMP (R6)+,(R6)+ ;FIX STK SINCE NO RTI
945 002606 104016             ERROR 16 ;REG ADRS TIME OUT ER
946 002610 012737 000006 000004      3$: MOV #ERRVEC+2,2#ERRVEC ;POINT TO HALT IN ADRS 6
947 002616 005037 000006             CLR 2#ERRVEC+2 ;SET UP HALT
948
949 ;*****
950 ;*TEST 2 FLOAT A ONE THRU TERMINATE CHARACTER REG
951 ;*****
952 002622 000004          †ST2: SCOPE
953 002624 012737 002646 001110      MOV #15,$LPERR ;SET UP SCOPE LOOP ADRS
954 002632 013737 002114 001122      MOV TERMCH,$BDADR ;SET UP TERMCH ADRS
955 002640 012737 000100 001124      MOV #100,$GDDAT ;EXPECT 100 INITIALLY
956 002646 013700 001124             1$: MOV $GDDAT,R0 ;PUT PATTERN IN R0
957 002652 052700 000200             BIS #200,R0 ;SET THE ENABLE BIT
958 002656 010077 177232             MOV R0,2TERMCH ;LOAD UP TERMCH REG
959 002662 017737 177226 001126      MOV 2TERMCH,$BDDAT ;READ IT BACK
960 002670 023737 001124 001126      CMP $GDDAT,$BDDAT ;IS IT CORRECT?
961 002676 001401             BEQ 2$ ;BR IF OK
962 002700 104017             ERROR 17 ;TERMINATE CHARACTER READ/WRITE ER
963 002702 006237 001124             2$: ASR $GDDAT ;SHIFT ONE BIT RIGHT
964 002706 001357             BNE 1$ ;BR IF NOT DONE
965
966 ;*****
967 ;*TEST 3 FLOAT A ZERO THRU TERMINATE CHARACTER REG
968 ;*****
969 002710 000004          †ST3: SCOPE
970 002712 012737 002740 001110      MOV #15,$LPERR ;SET UP SCOPE LOOP ADRS
971 002720 013737 002114 001122      MOV TERMCH,$BDADR ;SET UP TERMCH ADRS
972 002726 012737 000077 001124      MOV #77,$GDDAT ;EXPECT 77 INITIALLY
973 002734 012700 177677             MOV #177677,R0 ;PUT PATTERN IN R0
974 002740 010077 177150             1$: MOV R0,2TERMCH ;LOAD UP TERMCH REG - SEND HIGH BITS FOR NOISE
975 002744 017737 177144 001126      MOV 2TERMCH,$BDDAT ;READ IT BACK
976 002752 023737 001124 001126      CMP $GDDAT,$BDDAT ;IS IT CORRECT?
977 002760 001401             BEQ 2$ ;BR IF OK
978 002762 104017             ERROR 17 ;TERMINATE CHARACTER READ/WRITE ER
979 002764 006200             2$: ASR R0 ;SHIFT ZERO BIT RIGHT
980 002766 006237 001124             ASR $GDDAT ;SHIFT PATTERN IN COMPARE LOC
981 002772 052737 000100 001124      BIS #100,$GDDAT ;SET MSB
982 003000 103757             BCS 1$ ;BR IF NOT DONE
983
984 ;*****
985 ;*TEST 4 TEST TERMINATE CHARACTER REG IS NOT WRITEABLE WHEN CHANGE ENA=0

```

K02

MAINDEC-11-DZVSA-A
DZVSA.P11

VS60 INSTRUCTION TEST PART 1
T4 TEST TERMINATE CHARACTER REG IS NOT WRITEABLE WHEN CHANGE ENA=0

MACY11 27(732) 20-SEP-76 09:57 PAGE 23

```

996
997 003002 000004
998 003004 013737 002114 001122
999 003012 012737 000177 001124
990 003020 012777 177777 177066
991 003026 005077 177062
992 003032 017737 177056 001126
993 003040 023737 001124 001126
994 003046 001401
995 003050 104017
996
997
998
999
1000 003052 000004
1001 003054 012737 003076 001110
1002 003062 013737 002102 001122
1003 003070 012737 004000 001124
1004 003076 013777 001124 176776
1005 003104 017737 176772 001126
1006 003112 042737 170000 001126
1007 003120 023737 001124 001126
1008 003126 001401
1009 003130 104020
1010 003132 006237 001124
1011 003136 001357
1012
1013
1014
1015
1016 003140 000004
1017 003142 012737 003162 001110
1018 003150 013737 002102 001122
1019 003156 012700 173777
1020 003162 010037 001124
1021 003166 042737 170000 001124
1022 003174 010077 176702
1023 003200 017737 176676 001126
1024 003206 042737 170000 001126
1025 003214 023737 001124 001126
1026 003222 084101
1027 003224 104020
1028 003226 006200
1029 003230 103754
1030 003232 005077 176644
1031
1032
1033
1034
1035 003236 000004
1036 003240 012737 003262 001110
1037 003246 013737 002100 001122
1038 003254 012737 004000 001124
1039 003262 013777 001124 176610
1040 003270 017737 176604 001126
1041 003276 042737 170000 001126

```

```

*****
TST4: SCOPE
MOV TERMCH,$BDADR ;SET UP TERMCH ADRS
MOV #177,$GDDAT ;SET UP DATA PATTERN
MOV #177777,2TERMCH ;LOAD UP ALL WRITEABLE BITS
CLR 2TERMCH ;TRY TO CLEAR ALL BITS
MOV 2TERMCH,$BDDAT ;READ IT BACK
CMP $GDDAT,$BDDAT ;SEE IF ANY BITS WERE CLEARED
BEQ TST5 ;ADVANCE TO NEXT TEST IF OK
ERROR 17 ;TERMINATE CHARACTER ENABLE CHANGE ER
*****

*****
;*TEST 5 FLOAT A ONE THRU Y DYNAMIC OFFSET REG
*****
TST5: SCOPE
MOV #15,$LPERR ;SET UP SCOPE LOOP ADRS
MOV YDOFF,$BDADR ;SET UP YDOFF ADRS
MOV #4000,$GDDAT ;SET UP MSB TO A ONE
15: MOV $GDDAT,2YDOFF ;LOAD UP YDOFF REG
MOV 2YDOFF,$BDDAT ;READ IT BACK
BIC #170000,$BDDAT ;SAVE ONLY OFFSET VALUE
CMP $GDDAT,$BDDAT ;IS IT CORRECT?
BEQ 25 ;BR IF OK
ERROR 20 ;Y OFFSET READ/WRITE ER
25: ASR $GDDAT ;SHIFT ONE BIT RIGHT
BNE 15 ;BR IF NOT DONE
*****

*****
;*TEST 6 FLOAT A ZERO THRU Y DYNAMIC OFFSET REG
*****
TST6: SCOPE
MOV #15,$LPERR ;SET UP SCOPE LOOP ADRS
MOV YDOFF,$BDADR ;SET UP YDOFF ADRS
MOV #173777,R0 ;SET UP MSB TO A ZERO
15: MOV R0,$GDDAT ;SAVE PATTERN FOR COMPARE
BIC #170000,$GDDAT ;GET RID OF HIGH BITS THAT ARE NOT WRITEABLE
MOV R0,2YDOFF ;LOAD UP YDOFF REG - SEND HIGH BITS FOR NOISE
MOV 2YDOFF,$BDDAT ;READ IT BACK
BIC #170000,$BDDAT ;SAVE ONLY OFFSET VALUE
CMP $GDDAT,$BDDAT ;IS IT CORRECT?
BEQ 25 ;BR IF OK
ERROR 20 ;Y OFFSET READ/WRITE ER
25: ASR R0 ;SHIFT ZERO BIT RIGHT
BCS 15 ;BR IF NOT DONE
CLR 2YDOFF ;INSURE 0-Y OFFSET BEFORE ADVANCING
*****

*****
;*TEST 7 FLOAT A ONE THRU X DYNAMIC OFFSET REG
*****
TST7: SCOPE
MOV #15,$LPERR ;SET UP SCOPE LOOP ADRS
MOV XDOFF,$BDADR ;SET UP XDOFF ADRS
MOV #4000,$GDDAT ;SET UP MSB TO A ONE
15: MOV $GDDAT,2XDOFF ;LOAD UP XDOFF REG
MOV 2XDOFF,$BDDAT ;READ IT BACK
BIC #170000,$BDDAT ;SAVE ONLY OFFSET VALUE

```



```

1042 003304 023737 001124 001126      CMP      $GDDAT,$BDDAT      ;IS IT CORRECT?
1043 003312 001401                      BEQ      2$                  ;BR IF OK
1044 003314 104021                      ERROR   21                  ;X OFFSET READ/WRITE ER
1045 003316 006237 <001124      2$:    ASR      $GDDAT        ;SHIFT ONE BIT RIGHT
1046 003222 001357                      BNE     1$                  ;BR IF NOT DONE
1047
1048
1049
1050
1051 003324 000004                      ;*****
1052 003326 012737 003346 001110      ;*TEST 10  FLOAT A ZERO THRU X DYNAMIC OFFSET REG
1053 003334 013737 002100 001122      ;*****
1054 003342 012700 173777                      ;*TEST 10:  SCOPE
1055 003346 010037 001124                      MOV     #1$,$LPERR          ;SET UP SCOPE LOOP ADRS
1056 003352 042737 170000 001124      1$:    MOV     XDOFF,$BDADR     ;SET UP XDOFF ADRS
1057 003360 010077 176514                      MOV     #173777,R0         ;SET MBB TO A ZERO
1058 003364 017737 176510 001126      MOV     R0,$GDDAT          ;SAVE PATTERN FOR COMPARE
1059 003372 042737 170000 001126      BIC     #170000,$GDDAT     ;GET RID IF HIGH BITS THAT ARE NOT WRITEABLE
1060 003400 023737 001124 001126      MOV     R0,XDOFF           ;LOAD UP XDOFF REG - SEND HIGH BITS FOR NOISE
1061 003406 001401                      MOV     #170000,$BDDAT    ;READ IT BACK
1062 003410 104021                      BIC     #170000,$BDDAT    ;SAVE ONLY OFFSET VALUE
1063 003412 006200      2$:    CMP     $GDDAT,$BDDAT    ;IS IT CORRECT?
1064 003414 103754                      BEQ     2$                  ;BR IF OK
1065 003416 010377 176456      ERROR   21                  ;X OFFSET READ/WRITE ER
1066
1067
1068
1069
1070 003422 000004                      ;*****
1071 003424 012737 003452 001110      ;*TEST 11  TEST THAT THE Y OFFSET POLARITY BIT CAN SET & RESET
1072 003432 013737 002122 001122      ;*****
1073 003440 012700 020000                      ;*TEST 11:  SCOPE
1074 003444 012737 040000 001124      MOV     #1$,$LPERR          ;SET UP SCOPE LOOP ADRS
1075 003452 010077 176424      1$:    MOV     ZDOFF,$BDADR     ;SET UP REG 36 ADRS
1076 003456 017737 176440 001126      MOV     #20000,R0          ;R0 HAS BIT
1077 003464 042737 037777 001126      MOV     #40000,$GDDAT     ;LD EXPCTED
1078 003472 023737 001124 001126      MOV     R0,XDOFF          ;SEND IT OUT
1079 003500 001401                      MOV     #20000,$BDDAT     ;READ AT REG 36
1080 003502 104020                      BIC     #37777,$BDDAT     ;SAVE X & Y POLARITY BITS
1081 003504 005037 001124      2$:    CMP     $GDDAT,$BDDAT    ;CORRECT?
1082 003510 162700 020000                      BEQ     2$                  ;BR IF SO
1083 003514 100356                      ERROR   20                  ;Y OFFSET POLARITY READ/WRITE ER
1084
1085
1086
1087
1088 003516 000004                      ;*****
1089 003520 012737 003546 001110      ;*TEST 12  TEST THAT THE X OFFSET POLARITY BIT CAN SET & RESET
1090 003526 013737 002122 001122      ;*****
1091 003534 012700 020000                      ;*TEST 12:  SCOPE
1092 003540 012737 100000 001124      MOV     #1$,$LPERR          ;SET UP SCOPE LOOP ADRS
1093 003546 010077 176326      1$:    MOV     ZDOFF,$BDADR     ;SET UP REG 34 ADRS
1094 003552 017737 176344 001126      MOV     #20000,R0          ;R0 HAS BIT
1095 003560 042737 037777 001126      MOV     #100000,$GDDAT    ;LD EXPECTED
1096 003566 023737 001124 001126      MOV     R0,XDOFF          ;SEND IT OUT
1097 003574 001401                      MOV     #20000,$BDDAT     ;READ IT BACK
                      BIC     #37777,$BDDAT    ;SAVE X & Y POLARITY BITS
                      CMP     $GDDAT,$BDDAT    ;IS IT CORRECT?
                      BEQ     2$                  ;BR IF SO

```

M02

```

1098 003576 104021          ERROR 21          ;X OFFSET POLARITY READ/WRITE ER
1099 003600 005037 001124 2$: CLR $GDDAT ;EXPECT ZERO
1100 003604 162737 020000 001124 SUB #20000,$GDDAT ;ADVANCE TO RESET STATE
1101 003612 100355          BPL 1$          ;BR IF NOT YET TESTED
1102
1103 ;*****
1104 ;*TEST 13  FLOAT A ONE THRU RELOCATE REG
1105 ;*****
1106 003614 000004          †ST13: SCOPE
1107 003616 012737 003640 001110 MOV #1$,$LPERR ;SET UP SCOPE LOOP ADRS
1108 003624 013737 002074 001122 MOV RLO,$BDADR ;SET UP RLO ADRS
1109 003632 012737 004000 001124 MOV #4000,$GDDAT ;SET UP MSB TO A ONE
1110 003640 013777 001124 176226 1$: MOV $GDDAT,RLO ;LOAD UP RLO REG
1111 003646 017737 176222 001126 MOV RLO,$BDDAT ;READ IT BACK
1112 003654 023737 001124 001126 CMP $GDDAT,$BDDAT ;IS IT CORRECT?
1113 003662 001401          BEQ 2$          ;BR IF OK
1114 003664 104022          ERROR 22          ;RELOCATE REG READ/WRITE ER
1115 003666 006237 001124 2$: ASR $GDDAT ;SHIFT ONE BIT RIGHT
1116 003672 001362          BNE 1$          ;BR IF NOT DONE
1117
1118 ;*****
1119 ;*TEST 14  FLOAT A ZERO THRU RELOCATE REG
1120 ;*****
1121 003674 000004          †ST14: SCOPE
1122 003676 012737 003716 001110 MOV #1$,$LPERR ;SET UP SCOPE LOOP ADRS
1123 003704 013737 002074 001122 MOV RLO,$BDADR ;SET UP RLO ADRS
1124 003712 012700 173777          MOV #173777,R0 ;SET MSB TO A ZERO
1125 003716 010037 001124 1$: MOV R0,$GDDAT ;SAVE PATTERN FOR COMPARE
1126 003722 042737 170000 001124 BIC #170000,$GDDAT ;GET RID OF HIGH BITS THAT DO NOT READ/WRITE
1127 003730 010077 176140          MOV R0,RLO ;LOAD UP RLO REG - SEND HIGH BITS FOR NOISE
1128 003734 017737 176134 001126 MOV RLO,$BDDAT ;READ IT BACK
1129 003742 023737 001124 001126 CMP $GDDAT,$BDDAT ;IS IT CORRECT?
1130 003750 001401          BEQ 2$          ;BR IF OK
1131 003752 104022          ERROR 22          ;RELOCATE REG READ/WRITE ER
1132 003754 006200          2$: ASR R0 ;SHIFT ZERO BIT RIGHT
1133 003756 103757          BCS 1$          ;BR IF NOT DONE
1134 003760 005077 176110          CLR RLO ;CLR RELOCATE REG BEFORE ADVANCING
1135
1136 ;*****
1137 ;*TEST 15  TEST THAT ALL MAINT TYPE BITS ARE READ/WRITEABLE
1138 ;*****
1139 003764 000004          †ST15: SCOPE
1140 003766 013737 002116 001122 MOV STKPT,$BDADR ;SET UP REG 32 ADRS
1141 003774 012737 004010 001110 MOV #1$,$LPERR ;SET UP SCOPE LOOP ADRS
1142 004002 012737 170000 001124 MOV #170000,$GDDAT ;EXPECT ALL BITS INITIALLY
1143 004010 013777 001124 176100 1$: MOV $GDDAT,STKPT ;LOAD UP MAINT BITS
1144 004016 017737 176074 001126 MOV STKPT,$BDDAT ;READ THEM BACK
1145 004024 042737 007777 001126 BIC #7777,$BDDAT ;SAVE ONLY MAINT SWITCHES
1146 004032 023737 001124 001126 CMP $GDDAT,$BDDAT ;ARE THEY CORRECT?
1147 004040 001401          BEQ 2$          ;BR IF SO
1148 004042 104001          ERROR 1 ;MAINT SWITCH READ/WRITE ERROR
1149 004044 162737 010000 001124 2$: SUB #10000,$GDDAT ;GO TO NEXT PATTERN
1150 004052 100356          BPL 1$          ;BR IF ALL STATES NOT TESTED
1151
1152 ;*****
1153 ;*TEST 16  TEST THAT RESET WILL CLEAR THE RELOCATE REG
    
```

```

1154
1155 004054 000004
1156 004056 012737 000010 001164
1157 004064 013737 002074 001122
1158 004072 005037 001124
1159 004076 012777 007777 175770
1160 004104 000005
1161 004106 017737 175762 001126
1162 004114 001401
1163 004116 104022
1164
1165
1166
1167
1168 004120 000004
1169 004122 012737 000010 001164
1170 004130 013737 002100 001122
1171 004136 005037 001124
1172 004142 012777 007777 175730
1173 004150 000005
1174 004152 017737 175722 001126
1175 004160 001401
1176 004162 104021
1177
1178
1179
1180
1181 004164 000004
1182 004166 012737 000010 001164
1183 004174 013737 002102 001122
1184 004202 005037 001124
1185 004206 012777 007777 175666
1186 004214 000005
1187 004216 017737 175660 001126
1188 004224 001401
1189 004226 104020
1190
1191
1192
1193
1194 004230 000004
1195 004232 012737 000040 001164
1196 004240 013737 002122 001122
1197 004246 005037 001124
1198 004252 012777 020000 175620
1199 004260 012777 020000 175614
1200 004266 000005
1201 004270 017737 175626 001126
1202 004276 042737 037777 001126
1203 004304 023737 001124 001126
1204 004312 001401
1205 004314 104032
1206
1207
1208
1209

```

```

*****
TST16: SCOPE
MOV #10,$TIMES ;:DO 10 ITERATIONS
MOV RLO,$BDADR ;:SET UP REG 10 ADRS
CLR $GDDAT ;:EXPECT ALL ZEROS
MOV #7777,@RLO ;:LOAD IT UP
RESET ;:CLR IT OUT
MOV @RLO,$BDDAT ;:READ IT
BEQ TST17 ;:GO TO NEXT TEST IF CLEARED
ERROR 22 ;:RESET FAILED TO CLR RELOCATE REG

*****
*TEST 17 TEST THAT RESET WILL CLEAR THE X DYNAMIC OFFSET REG
*****
TST17: SCOPE
MOV #10,$TIMES ;:DO 10 ITERATIONS
MOV XDOFF,$BDADR ;:SET UP REG 14 ADRS
CLR $GDDAT ;:EXPECT ALL ZEROS
MOV #7777,@XDOFF ;:LOAD IT UP
RESET ;:CLR IT OUT
MOV @XDOFF,$BDDAT ;:READ IT
BEQ TST20 ;:GO TO NEXT TEST IF CLEARED
ERROR 21 ;:RESET FAILED TO CLR X DYNAMIC OFFSET REG

*****
*TEST 20 TEST THAT RESET WILL CLEAR THE Y DYNAMIC OFFSET REG
*****
TST20: SCOPE
MOV #10,$TIMES ;:DO 10 ITERATIONS
MOV YDOFF,$BDADR ;:SET UP REG 16 ADRS
CLR $GDDAT ;:EXPECT ALL ZEROS
MOV #7777,@YDOFF ;:LOAD IT UP
RESET ;:CLR IT OUT
MOV @YDOFF,$BDDAT ;:READ IT
BEQ TST21 ;:GO TO NEXT TEST IF CLEARED
ERROR 20 ;:RESET FAILED TO CLR Y DYNAMIC OFF SET REG

*****
*TEST 21 TEST THAT RESET WILL CLEAR X & Y OFFSET POLARITY BITS
*****
TST21: SCOPE
MOV #40,$TIMES ;:DO 40 ITERATIONS
MOV ZDOFF,$BDADR ;:SET UP REG 36 ADRS
CLR $GDDAT ;:EXPECT ZERO
MOV #20000,@XDOFF ;:SET X POLARIYT BIT
MOV #20000,@YDOFF ;:SET Y POLARITY BIT
RESET ;:CLR THEM
MOV @ZDOFF,$BDDAT ;:READ REG 36
BIC #37777,$BDDAT ;:SAVE ONLY POLARITY BITS
CMP $GDDAT,$BDDAT ;:CORRECT?
BEQ TST22 ;:NEXT TEST IF SO
ERROR 32 ;:RESET FAILED TO CLR OFFSET POLARIY BITS

*****
*TEST 22 TEST THAT RESET WILL CLEAR TERMINATE CHARACTER REG
*****

```

004316 000004
004320 012737 000010 001164
004326 013737 002114 001122
004334 005037 001124
004340 012777 000377 175546
004346 000005
004350 017737 175540 001126
004356 001401
004360 104017

TST22: SCOPE
MOV #10, \$TIMES ;: DO 10 ITERATIONS
MOV TERMCH, \$BDADR ;: SET UP REG 30 ADPS
CLR \$GDDAT ;: EXPECT ALL ZEROS
MOV #377, \$TERMCH ;: LOAD IT UP
RESET ;: CLR IT OUT
MOV \$TERMCH, \$BDAT ;: READ IT
BEQ TST23 ;: GO TO NEXT TEST IF CLEARED
ERROR 17 ;: RESET FAILED TO CLR TERMINATE CHAR REG

*TEST 23 TEST THAT RESET CLEARS ALL MAINT BITS

004362 000004
004364 012737 000100 001164
004372 013737 002116 001122
004400 005037 001124
004404 012777 170000 175504
004412 000005
004414 017737 175476 001126
004422 042737 003777 001126
004430 001401
004432 104032

TST23: SCOPE
MOV #100, \$TIMES ;: DO 100 ITERATIONS
MOV \$TKPT, \$BDADR ;: SET UP REG ADPS 32
CLR \$GDDAT ;: LOAD EXPECTED
MOV #170000, \$TKPT ;: LOAD REGISTER
RESET ;: INITILIZE
MOV \$TKPT, \$BDAT ;: READ REG.
BIC #377, \$BDAT ;: MASK OUT OTHER BITS
BEQ TST24 ;: BR IF ALL CLEARED
ERROR 32 ;: RESET FAILED TO CLEAR MAINT. BITS

*TEST 24 TEST THAT THE STACK SELECTION BITS ARE READ/WRITEABLE

004434 000004
004436 012737 004460 001110
004444 012737 002116 001122
004452 012737 000037 001124
004460 013777 001124 175430
004466 017737 175424 001126
004474 042737 177740 001126
004502 023737 001124 001126
004510 001401
004512 104051
004514 005337 001124
004520 100357

TST24: SCOPE
MOV #15, \$LPERR ;: SET UP SCOPE LOOP ADPS
MOV \$TKPT, \$BDADR ;: SET UP REG ADPS 32
MOV #37, \$GDDAT ;: EXPECT ALL BITS INITIALLY
15: MOV \$GDDAT, \$TKPT ;: SEND PATTERN OUT
MOV \$TKPT, \$BDAT ;: READ IT BACK
BIC #177740, \$BDAT ;: SAVE ONLY STACK SELECTION BITS
CMP \$GDDAT, \$BDAT ;: CORRECT?
BEQ 25 ;: BR IF SO
ERROR 51 ;: STACK POINTER READ/WRITE ER
25: DEC \$GDDAT ;: ADVANCE PATTERN
BPL 15 ;: BR IF ALL PATTERNS NOT TESTED

*TEST 25 TEST THAT 'TOP OF STACK' AND RESET CLEARS STACK SELECTION BITS

004522 000004
004524 012737 000040 001164
004532 013737 002116 001122
004540 012737 000041 001124
004546 012777 000037 175342
004554 012777 000077 175334
004562 017737 175330 001126
004570 042737 177700 001126
004576 023737 001124 001126
004604 001401
004606 104051
004610 012737 000040 001124
004616 012777 000037 175272

TST25: SCOPE
MOV #40, \$TIMES ;: DO 40 ITERATIONS
MOV \$TKPT, \$BDADR ;: SET UP REG ADPS 32
MOV #41, \$GDDAT ;: LOAD EXPECTED
MOV #37, \$TKPT ;: SET ALL BITS
MOV #77, \$TKPT ;: CLR ALL BITS
MOV \$TKPT, \$BDAT ;: READ STACK SELECTION BITS
BIC #177700, \$BDAT ;: SAVE SEL BITS ONLY
CMP \$GDDAT, \$BDAT ;: CORRECT?
BEQ 15 ;: BR IF SO
ERROR 51 ;: 'TOP OF STACK' FAILED TO CLR BITS
15: MOV #40, \$GDDAT ;: LD EXPECTED
MOV #37, \$TKPT ;: SET ALL BITS

```

1286 004624 000005 RESET ;RESET CLRS THEM & SET 'TOP OF STACK'
1287 004626 017737 175264 001126 MOV 2STKPT,$BDCAT ;READ THEM
1288 004634 023737 001124 001126 CMP $GDDAT,$BDCAT ;CORRECT?
1289 004642 001401 BEQ TST26 ;NEXT TEST IF SO
1290 004644 104051 ERROR 51 ;RESET FAILED TO SET UP 'TOP OF STACK'

```

```

.SBTTL
.SBTTL THESE TESTS USE MAINT SWITCH #1
.SBTTL

```

```

*****
;*TEST 26 FLOAT A 1 ACROSS THE D.P.C.
*****

```

```

1291 004646 000004 TST26: SCOPE
1292 004650 012737 000040 001164 MOV #40,$TIMES ;DO 40 ITERATIONS
1293 004656 013737 002064 001122 MOV DPC,$BDCAT ;SET UP REG ADRS 00
1294 004664 012737 000002 001124 MOV #BIT1,$GDDAT ;LOAD EXPECTED
1295 004672 012777 010000 175216 MOV #BIT12,2STKPT ;SET MAINT SWITCH #1
1296 004700 013777 001124 175156 15: MOV $GDDAT,DPC ;LOAD D.P.C.
1297 004706 017737 175152 001126 MOV DPC,$BDCAT ;READ D.P.C. INTO $BDCAT
1298 004714 023737 001124 001126 CMP $GDDAT,$BDCAT ;COMPARE VALUE EXPECTED TO READ
1299 004722 001402 BEQ 25 ;BR IF SAME
1300 004724 104002 ERROR 2 ;FAILED TO LOAD DPC WITH A FLOATING 1
1301 004726 000403 BR TST27 ;
1302 004730 006337 001124 25: ASL $GDDAT ;CHANGE DATA EXPECTED
1303 004734 103361 BCC 15 ;BR IF NOT COMPLETED

```

```

*****
;*TEST 27 TEST THAT THE RELOCATE REG WILL CORRECTLY SET UP THE DPC
*****

```

```

1304 004736 000004 TST27: SCOPE
1305 004740 012737 000040 001164 MOV #40,$TIMES ;DO 40 ITERATIONS
1306 004746 012737 005006 001110 MOV #15,$LPERR ;SET UP SCOPE LOOP ADRS
1307 004754 012777 010000 175134 MOV #BIT12,2STKPT ;SET MAINT SWITCH #1
1308 004762 013737 002064 001122 MOV DPC,$BDCAT ;SET UP REG 00 ADRS
1309 004770 005077 175070 CLR DPC ;CLEAR DPC
1310 004774 012737 000100 001124 MOV #100,$GDDAT ;EXPECT 100 INITIALLY
1311 005002 012700 000001 MOV #1,R0 ;START WITH 1
1312 005006 010077 175062 15: MOV R0,DPC ;LOAD RELOCATE REG
1313 005012 017737 175046 001126 MOV DPC,$BDCAT ;READ RELOCATE REG - SHOULD BE TIMES 64
1314 005020 023737 001124 001126 CMP $GDDAT,$BDCAT ;IS IT CORRECT?
1315 005026 001401 BEQ 25 ;BR IF SO
1316 005030 104022 ERROR 22 ;RELOCATE FAILED TO SET UP DPC CORRECTLY
1317 005032 062737 000100 001124 25: ADD #100,$GDDAT ;SET UP NEXT EXPECTED VALUE
1318 005040 001402 BEQ TST30 ;GO TO NEXT TEST IF ALL VALUES TESTED
1319 005042 005200 INC R0 ;SET UP NEXT PATTERN
1320 005044 000760 BR 15 ;GO TRY IT

```

```

*****
;*TEST 30 TEST THAT RESET CLEARS THE D.P.C.
*****

```

```

1321 005046 000004 TST30: SCOPE
1322 005050 012737 000100 001164 MOV #100,$TIMES ;DO 100 ITERATIONS
1323 005056 013737 002064 001122 MOV DPC,$BDCAT ;SET UP REG ADRS 00
1324 005064 012777 010000 175024 MOV #BIT12,2STKPT ;SET MAINT SWITCH #1
1325 005072 005037 001124 CLR $GDDAT ;LOAD EXPECTED

```

```

1322 005076 012777 177776 174760      MOV      #2,DPCC      ;LOAD REGISTER
1323 005104 000005                RESET              ;INITILIZE
1324 005106 017737 174752 001126      MOV      DPCC,$BDDAT ;READ D.P.C. REG.
1325 005114 001401                BEQ      TST31      ;;BR IF ALL BITS CLEARED
1326 005116 104032                ERROR     32        ;RESET FAILED TO CLEAR THE D.P.C.
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377

```

```

.SBTTL
.SBTTL THESE TESTS USE MAINT. SWITCH 2
.SBTTL

;*****
;*TEST 31 LOAD #17 INTO THE "MODE" REGISTERS
;*****
TST31: SCOPE
MOV      SREG0,$B0ADR ;SET UP REG ADRS 02
MOV      #BIT13,$STKPT ;SET MAINT SWITCH #2
MOV      #174000,BUFFER ;LOAD MODE REGISTER=17
MOV      DBUF,DPCC ;LOAD DISPLAY PC.
MOV      #74000,$GDDAT ;LOAD EXPECTED
MOV      $SREG0,$BDDAT ;READ DISPLAY STATUS REGISTER
BIC      #103777,$BDDAT ;MASK TO BITS 14-11
CMP      $GDDAT,$BDDAT ;COMPARE EXPCT TO RCVD
BEQ      TST32      ;;BR IF EQUAL
ERROR    3          ;MODE BITS (14-11) FAILED TO SET

```

```

;*****
;*TEST 32 LOAD #10 INTO THE "MODE" REGISTERS
;*****
TST32: SCOPE
MOV      SREG0,$B0ADR ;SET UP REG ADRS 02
MOV      #BIT13,$STKPT ;SET MAINT SWITCH #2
MOV      #140000,BUFFER ;LOAD MODE REGISTER=10
MOV      DBUF,DPCC ;LOAD DISPLAY P.C.
MOV      #40000,$GDDAT ;LOAD EXPECTED
MOV      $SREG0,$BDDAT ;READ DISPLAY STATUS REGISTER
BIC      #103777,$BDDAT ;MASK TO BITS 14-11
CMP      $GDDAT,$BDDAT ;COMPARE EXPCT TO RCVD
BEQ      TST33      ;;BR IF EQUAL
ERROR    3          ;MODE BIT 14 FAILED TO SET

```

```

;*****
;*TEST 33 LOAD #14 INTO THE "MODE" REGISTERS
;*****
TST33: SCOPE
MOV      SREG0,$B0ADR ;SET UP REG ADRS 02
MOV      #BIT13,$STKPT ;SET MAINT SWITCH #2
MOV      #160000,BUFFER ;LOAD MODE REGISTER=14
MOV      DBUF,DPCC ;LOAD DISPLAY P.C.
MOV      #60000,$GDDAT ;LOAD EXPECTED
MOV      $SREG0,$BDDAT ;READ DISPLAY STATUS REGISTER
BIC      #103777,$BDDAT ;MASK TO BITS 14-11
CMP      $GDDAT,$BDDAT ;COMPARE EXPCT TO RCVD
BEQ      TST34      ;;BR IF EQUAL
ERROR    3          ;MODE BIT 13 FAILED TO SET

```

E03

1378
1379
1380
1381 005362 000004
1382 005364 013737 002066 001122
1383 005372 012777 020000 174516
1384 005400 012737 170000 027526
1385 005406 013777 002034 174450
1386 005414 012737 070000 001124
1387 005422 017737 174440 001126
1388 005430 042737 103777 001126
1389 005436 023737 001124 001126
1390 005444 001401
1391 005446 104003
1392
1393
1394
1395 005450 000004
1396 005452 013737 002064 001122
1397 005460 012777 020000 174430
1398 005466 012700 027526
1399 005472 012720 164000
1400 005476 012720 164000
1401 005502 012720 164000
1402 005506 012720 164000
1403 005512 013777 002034 174344
1404 005520 012737 027530 001124
1405 005526 017737 174332 001126
1406 005534 023737 001124 001126
1407 005542 001402
1408 005544 104004
1409 005546 000451
1410 005550 005277 174310 15:
1411 005554 062737 000002 001124
1412 005562 017737 174276 001126
1413 005570 023737 001124 001126
1414 005576 001402
1415 005500 104004
1416 005602 000433
1417 005604
1418 005604 005277 174254 25:
1419 005610 062737 000002 001124
1420 005616 017737 174242 001126
1421 005624 023737 001124 001126
1422 005632 001402
1423 005634 104004
1424 005636 000415
1425 005640
1426 005640 005277 174220 35:
1427 005644 062737 000002 001124
1428 005652 017737 174206 001126
1429 005660 023737 001124 001126
1430 005666 001401
1431 005670 104004
1432
1433

```
*****  
: *TEST 34 LOAD #16 INTO THE "MODE" REGISTERS  
*****  
TST34: SCOPE  
MOV SREGO,$B0ADR ;SET UP REG ADRS 02  
MOV #BIT13,$STKPT ;SET MAINT SWITCH #2  
MOV #170000,BUFFER ;LOAD MODE REGISTER=16  
MOV DBUF,$DPC ;LOAD DISPLAY P.C.  
MOV #70000,$GDDAT ;LOAD EXPECTED  
MOV $SREGO,$BDDAT ;READ DISPLAY STATUS REGISTER  
BIC #103777,$BDDAT ;MASK TO BITS 14-11  
CMP $GDDAT,$BDDAT ;COMPARE EXPCT TO RCVD  
SEQ TST35 ;:BR IF EQUAL  
ERROR 3 ;MODE BIT 12 FAILED TO SET  
*****  
: *TEST 35 SIMPLE INCREMENT D.P.C. TEST  
*****  
TST35: SCOPE  
MOV DPC,$B0ADR ;SET UP REG ADRS 00  
MOV #BIT13,$STKPT ;SET MAINT SWITCH #2  
MOV #BUFFER,R0 ;LOAD BUFFER POINTER  
MOV #164000,(R0)+ ;LOAD DISPLAY NOP  
MOV #164000,(R0)+  
MOV #164000,(R0)+  
MOV #164000,(R0)+  
MOV DBUF,$DPC ;START THE DISPLAY  
MOV #BUFFER+2,$GDDAT ;LOAD EXPECTED  
MOV $DPC,$BDDAT ;READ THE DISPLAY P.C.  
CMP $GDDAT,$BDDAT ;TEST FOR INCREMENT  
BEQ 15 ;DISPLAY P.C. FAILED TO INCREMENT  
ERROR 4 ;  
BR TST36 ;  
15: INC $DPC ;STEP THE DISPLAY P.C.  
ADD #2,$GDDAT ;UPDATE EXPECTED  
MOV $DPC,$BDDAT ;READ THE DISPLAY P.C.  
CMP $GDDAT,$BDDAT ;TEST FOR INCREMENT  
BEQ 25 ;  
ERROR 4 ;DISPLAY P.C. FAILED TO INCREMENT  
BR TST36 ;  
25: INC $DPC ;SINGLE STEP THE DISPLAY  
ADD #2,$GDDAT ;UPDATE EXPECTED  
MOV $DPC,$BDDAT ;READ THE DISPLAY P.C.  
CMP $GDDAT,$BDDAT ;TEST FOR INCREMENT  
BEQ 35 ;  
ERROR 4 ;DISPLAY P.C. FAILED TO INCREMENT  
BR TST36 ;  
35: INC $DPC ;SINGLE STEP THE DISPLAY  
ADD #2,$GDDAT ;UPDATE EXPECTED  
MOV $DPC,$BDDAT ;READ THE DISPLAY P.C.  
CMP $GDDAT,$BDDAT ;TEST FOR INCREMENT  
BEQ TST36 ;:BR IF EQUAL  
ERROR 4 ;DISPLAY P.C. FAILED TO INCREMENT  
*****
```

F03

```

1434      ;*TEST 36      EXPANDED D.P.C. INCREMENT TEST
1435      ;*****
1436      †TST36:  SCOPE
1437      MOV      #40,$TIMES      ;;DO 40 ITERATIONS
1438      MOV      DPC,$BADDR      ;;SET UP REG ADRS 00
1439      MOV      #B1+?,$STKPT      ;;SET MAINT SWITCH #2
1440      MOV      DBUF,R2      ;;SET UP POINTER
1441      MOV      #164000,(R2)+      ;;MOVE DNOP INTO THE BUFFER
1442      CMP      SIZE,R2      ;;FINISHED FILLING THE BUFFER?
1443      BNE      1$      ;;NO
1444      MOV      DBUF,$DPC      ;;YES, START THE DISPLAY
1445      MOV      #BUFFER,$GDDAT      ;;LOAD EXPECTED
1446      MOV      SIZE,R2      ;;SETUP A COUNT
1447      CMP      -(R2),-(R2)      ;;DEC BY 2
1448      ADD      #2,$GDDAT      ;;UPDATE EXPECTED
1449      MOV      $DPC,$BDDAT      ;;READ DISPLAY P.C.
1450      CMP      $GDDAT,$BDDAT      ;;DID IT INCREMENT BY 2?
1451      BEQ      3$      ;;YES
1452      ERROR    4      ;;DISPLAY PC FAILED TO INCREMENT
1453      BR       TST37      ;;
1454
1455      3$:      CMP      R2,$GDDAT      ;;FINISHED THE BUFFER
1456      BEQ      TST37      ;;BR TO NEXT TEST
1457      INC      $DPC      ;;SINGLE STEP THE DISPLAY
1458      BR       2$      ;;TRY AGAIN
1459
1460      ;*****
1461      ;*TEST 37      TEST THAT THE NAME INSTR CAN FLOAT A ONE THRU DPU NAME REG
1462      ;*****
1463      †TST37:  SCOPE
1464      MOV      #BIT13,$STKPT      ;;SET MAINT SWITCH #2
1465      MOV      #15,$LPERR      ;;SET UP SCOPE LOOP ADRS
1466      MOV      #2000,$GDDAT      ;;SET UP MSB
1467      MOV      #15000,BUFFER      ;;SET UP INSTR LOC
1468      BIS      $GDDAT,BUFFER      ;;SET NAME BIT AT INSTR LOC
1469      MOV      #BUFFER,$DPC      ;;START
1470      MOV      DNAME,$BADDR      ;;SET UP REG ADRS 24
1471      MOV      $DNAME,$BDDAT      ;;READ NAME REG
1472      BIC      #174000,$BDDAT      ;;ONLY INTERESTED IN THE NAME BITS
1473      CMP      $GDDAT,$BDDAT      ;;DID IT GET LOADED PROPERLY?
1474      BEQ      2$      ;;BR IF OK
1475      ERROR    23      ;;DPU NAME INSTR ER
1476      ASR      $GDDAT      ;;SHIFT 'ONE' BIT RIGHT
1477      BCC      1$      ;;BR IF MORE BIT POSITIONS
1478
1479      ;*****
1480      ;*TEST 40      TEST THAT THE NAME INSTR CAN FLOAT A ZERO THRU DPU NAME REG
1481      ;*****
1482      †TST40:  SCOPE
1483      MOV      #BIT13,$STKPT      ;;SET MAINT SWITCH #2
1484      MOV      #15,$LPERR      ;;SET UP SCOPE LOOP ADRS
1485      MOV      #175777,R0      ;;SET MSB TO A ZERO
1486      MOV      R0,$GDDAT      ;;PUT PATTERN IN COMPARE LOC
1487      BIC      #174000,$GDDAT      ;;GET RID OF EXTRA BITS
1488      MOV      #15000,BUFFER      ;;SET UP INSTR
1489      BIS      $GDDAT,BUFFER      ;;SET UP PATTERN AT INSTR LOC

```

1490	006202	012777	027526	173654	MOV	#BUFFER, DPC	; START
1491	006210	013737	002110	001122	MOV	DNAME, \$BDADR	; SET UP REG ADRS 24
1492	006216	017737	173666	001126	MOV	DNAME, \$BDDAT	; READ NAME REG
1493	006224	042737	170000	001126	BIC	#170000, \$BDDAT	; ONLY INTERESTED IN THE NAME BITS
1494	006232	023737	001124	001126	CMP	\$GDDAT, \$BDDAT	; DID IT GET LOADED PROPERLY?
1495	006240	001401			BEQ	25	; BR IF OK
1496	006242	104023			ERROR	23	; DPU NAME INSTR ER
1497	006244	006200			ASR	RO	; SHIFT ZERO BIT RIGHT
1498	006246	103742			BCS	15	; BR IF MORE BIT POSITIONS

 ;*TEST 41 TEST THAT RESET WILL CLEAR DPU NAME

1503	006250	000004			TST41: SCOPE		
1504	006252	012737	000040	001164	MOV	#40, \$TIMES	; DO 40 ITERATIONS
1505	006260	012777	020000	173630	MOV	#BIT13, \$STKPT	; SET MAINT SWITCH 2
1506	006266	005037	001124		CLR	\$GDDAT	; EXPECT ALL ZEROS
1507	006272	012737	153777	027526	MOV	#153777, BUFFER	; SET UP NAME INSTR
1508	006300	012777	027526	173556	MOV	#BUFFER, DPC	; START
1509	006306	013737	002110	001122	MOV	DNAME, \$BDADR	; SET UP REG ADRS 24
1510	006314	000005			RESET		; CLR DPU NAME REG
1511	006316	017737	173566	001126	MOV	DNAME, \$BDDAT	; READ IT
1512	006324	001401			BEQ	TST42	; NEXT TEST IF ALL CLEARED
1513	006326	104023			ERROR	23	; RESET FAILED TO CLEAR DPU NAME

 ;*TEST 42 TEST THAT THE SEARCH CODE BITS CAN BE READ AT REG 24

1518	006330	000004			TST42: SCOPE		
1519	006332	012737	006360	001110	MOV	#15, \$LPERR	; SET UP SCOPE LOOP ADRS
1520	006340	013737	002110	001122	MOV	DNAME, \$BDADR	; SET UP REG 24 ADRS
1521	006346	012737	030000	001124	MOV	#30000, \$GDDAT	; EXPECT BOTH BITS INITIALLY
1522	006354	012700	070000		MOV	#70000, RO	; SET UP RO WITH ENABLE
1523	006360	010077	173520		MOV	RO, DNAME	; SEND SEARCH CODE
1524	006364	017737	173520	001126	MOV	DNAME, \$BDDAT	; READ IT BACK
1525	006372	042737	147777	001126	BIC	#147777, \$BDDAT	; SAVE ONLY SEARCH CODE
1526	006400	023737	001124	001126	CMP	\$GDDAT, \$BDDAT	; IS IT CORRECT?
1527	006406	001401			BEQ	25	; BR IF OK
1528	006410	104024			ERROR	24	; SEARCH CODE READ/WRITE ER
1529	006412	162700	010000		SUB	#10000, RO	; SET UP NEXT SEARCH CODE
1530	006416	162737	010000	001124	SUB	#10000, \$GDDAT	; SET UP NEXT SEARCH CODE
1531	006424	100355			BPL	15	; BR IF MORE CODES TO TEST

 ;*TEST 43 TEST THAT THE SEARCH CODE DOES NOT CHANGE WHEN ENA=0

1536	006426	000004			TST43: SCOPE		
1537	006430	013737	002110	001122	MOV	DNAME, \$BDADR	; SET UP REG 24 ADRS
1538	006436	012737	030000	001124	MOV	#30000, \$GDDAT	; EXPECT BOTH BITS
1539	006444	012777	070000	173432	MOV	#70000, DNAME	; LOAD UP SEARCH CODE
1540	006452	005077	173426		CLR	DNAME	; TRY TO CLR SEARCH CODE WITH ENA=0
1541	006456	017737	173426	001126	MOV	DNAME, \$BDDAT	; READ IT BACK
1542	006464	042737	147777	001126	BIC	#147777, \$BDDAT	; SAVE ONLY SEARCH CODE
1543	006472	023737	001124	001126	CMP	\$GDDAT, \$BDDAT	; IS IT CORRECT?
1544	006500	001401			BEQ	TST44	; ADVANCE TO NEXT TEST IF OK
1545	006502	104024			ERROR	24	; SEARCH CODE CHANGE ENABLE ER

H03

```
1546
1547
1548
1549
1550 006504 000004
1551 006506 012737 000040 001164
1552 006514 013737 002110 001122
1553 006522 005037 001124
1554 006526 012777 070000 173350
1555 006534 000005
1556 006536 017737 173346 001126
1557 006544 001401
1558 006546 104024
1559
1560
1561
1562
1563 006550 000004
1564 006552 012737 000010 001164
1565 006560 012777 020000 173330
1566 006566 012737 000340 177776
1567 006574 012737 006630 001110
1568 006602 012737 033600 001124
1569 006610 052737 044000 001124
1570 006616 013777 001124 173260
1571 006624 012700 003600
1572 006630 042737 144000 001124
1573 006636 012737 150000 027526
1574 006644 050037 027526
1575 006650 012777 027526 173206
1576 006656 013737 002110 001122
1577 006664 017737 173220 001126
1578 006672 100410
1579 006674 023737 001124 001126
1580 006702 001017
1581 006704 052737 100000 001124
1582 006712 104024
1583 006714 042737 100000 001126
1584 006722 023737 001124 001126
1585 006730 001404
1586 006732 052737 100000 001126
1587 006740 104024
1588 006742 162700 000200
1589 006746 100330
1590 006750 162737 000200 001124
1591 006756 022737 027600 001124
1592 006764 001311
1593
1594
1595
1596
1597 006766 000004
1598 006770 012737 000004 001164
1599 006776 012777 020000 173112
1600 007004 012737 000340 177776
1601 007012 012737 007046 001110

:*****
:*TEST 44 TEST THAT RESET WILL CLEAR SEARCH CODE
:*****
†ST44: SCOPE
MOV #40,$TIMES ;;DO 40 ITERATIONS
MOV DNAME,$BDADR ;;SET UP REG 24 ADRS
CLR $GDDAT ;;EXPECT RESET STATE
MOV #70000,$ANAME ;;SET BOTH SEARCH CODE BITS
RESET ;;CLR THEM
MOV $DNAME,$BDDAT ;;READ REG
BEQ TST45 ;;NEXT TEST IF ALL CLEARED
ERROR 24 ;;RESET FAILED TO CLR SEARCH CODE

:*****
:*TEST 45 TEST THAT THE NAME MATCH FLAG WILL SET ON 4 BIT COMPARES
:*****
†ST45: SCOPE
MOV #10,$TIMES ;;DO 10 ITERATIONS
MOV #BIT13,$STKPT ;;SET MAINT SWITCH #2
MOV #340,$PSW ;;SET UP PRIORITY TO HIGHEST LEVEL
MOV #25,$LPERR ;;SET UP SCOPE LOOP ADRS
MOV #33600,$GDDAT ;;START WITH MAX VALUE
1$: BIS #44000,$GDDAT ;;SET THE SEARCH CODE ENA BIT
MOV $GDDAT,$ANAME ;;SET UP ASSOCIATIVE NAME REG
MOV #3600,$R0 ;;SET UP R0 WITH MAX VALUE
2$: BIC #144000,$GDDAT ;;CLR MATCH FLAG & SEARCH ENA BIT
MOV #150000,$BUFFER ;;SET UP NAME INSTR
BIS $R0,$BUFFER ;;SET UP NAME VALUE
MOV #BUFFER,$DPC ;;START
MOV DNAME,$BDADR ;;SET UP REG ADRS 24
MOV $DNAME,$BDDAT ;;READ NAME REG
BMI 3$ ;;BR IF A MATCH OCCURRED
CMP $GDDAT,$BDDAT ;;SHOULD A MATCH HAVE OCCURRED?
BNE 4$ ;;BR IF NOT
BIS #100000,$GDDAT ;;INDICATE IT SHOULD BE SET
ERROR 24 ;;NAME MATCH FLAG FAILED TO SET
3$: BIC #100000,$BDDAT ;;CLR OUT MATCH FLAG BIT
CMP $GDDAT,$BDDAT ;;MAKE SURE THERE IS A MATCH
BEQ 4$ ;;BR IF OK
BIS #100000,$BDDAT ;;RESTORE THE MATCH FLAG BIT
ERROR 24 ;;NAME MATCH FLAG SET IN ER
4$: SUB #200,$R0 ;;ADVANCE NAME VALUE
BPL 2$ ;;BR IF MORE VALUES TO TEST
SUB #200,$GDDAT ;;ADVANCE VALUE FOR ASSO. NAME REG
CMP #27600,$GDDAT ;;HAVE ALL VALUES BEEN TESTED?
BNE 1$ ;;BR IF NOT

:*****
:*TEST 46 TEST THAT THE NAME MATCH FLAG WILL SET ON 8 BIT COMPARES
:*****
†ST46: SCOPE
MOV #4,$TIMES ;;DO 4 ITERATIONS
MOV #BIT13,$STKPT ;;SET MAINT SWITCH #2
MOV #340,$PSW ;;SET PRIORITY TO HIGEST LEVEL
MOV #25,$LPERR ;;SET UP SCOPE LOOP ADRS
```

TEST THAT THE NAME MATCH FLAG WILL SET ON 8 BIT COMPARES

1602	007020	012737	023770	001124		MOV	#23770,\$GDDAT	:START WITH MAX VALUE
1603	007026	052737	044000	001124	1\$:	BIS	#44000,\$GDDAT	:SET SEARCH ENA BIT
1604	007034	013777	001124	173042		MOV	\$GDDAT,\$ANAME	:SET UP ASSOCIATIVE NAME REG
1605	007042	012700	003770			MOV	#3770,R0	:SET UP R0 WITH MAX VALUE
1606	007046	042737	144000	001124	2\$:	BIC	#144000,\$GDDAT	:CLR MATCH FLAG & SEARCH ENA
1607	007054	012737	150000	027526		MOV	#150000,BUFFER	:SET UP NAME INSTR
1608	007062	050037	027526			BIS	R0,BUFFER	:SET UP NAME VALUE
1609	007066	012777	027526	172770		MOV	#BUFFER,\$DPC	:START
1610	007074	013737	002110	001122		MOV	DNAME,\$BADDR	:SET UP REG ADRS 24
1611	007102	017737	173002	001126		MOV	\$DNAME,\$BDDAT	:READ NAME REG
1612	007110	100410				BMI	3\$:BR IF A MATCH OCCURED
1613	007112	023737	001124	001126		CMP	\$GDDAT,\$BDDAT	:SHOULD A MATCH HAVE OCCURED?
1614	007120	001017				BNE	4\$:BR IF NOT
1615	007122	052737	100000	001124		BIS	#100000,\$GDDAT	:INDICATE IT SHOULD BE SET
1616	007130	104024				ERROR	24	:NAME MATCH FLAG FAILED TO SET
1617	007132	042737	100000	001126	3\$:	BIC	#100000,\$BDDAT	:CLR OUT MATCH FLAG BIT
1618	007140	023737	001124	001126		CMP	\$GDDAT,\$BDDAT	:MAKE SURE THERE IS A MATCH
1619	007146	001404				BEQ	4\$:BR IF OK
1620	007150	052737	100000	001126		BIS	#100000,\$BDDAT	:RESTORE THE MATCH FLAG BIT
1621	007156	104024				ERROR	24	:NAME MATCH FLAG SET IN ER
1622	007160	162700	000010		4\$:	SUB	#10,R0	:ADVANCE NAME VALUE
1623	007164	100330				BPL	2\$:BR IF MORE VALUES TO TEST
1624	007166	162737	000010	001124		SUB	#10,\$GDDAT	:ADVANCE VALUE FOR ASSO. NAME REG
1625	007174	022737	017770	001124		CMP	#17770,\$GDDAT	:HAVE ALL VALUES BEEN TESTED?
1626	007202	001311				BNE	1\$:BR IF NOT

 ;*TEST 47 TEST THAT THE NAME MATCH FLAG WILL SET ON 11 BIT COMPARES

1630						↑ST47: SCOPE		
1631	007204	000004				MOV	#2,\$TIMES	::DO 2 ITERATIONS
1632	007206	012737	000002	001164		MOV	#BIT13,\$STKPT	:SET MAINT SWITCH #2
1633	007214	012777	020000	172674		MOV	#2\$,\$LPERR	:SET UP SCOPE LOOP ADRS
1634	007222	012737	007256	001110		MOV	#10077,\$GDDAT	:START WITH 77 - WORK ONLY LOWER ORDER BITS
1635	007230	012737	010077	001124		MOV	#44000,\$GDDAT	:SET SEARCH ENA BIT
1636	007236	052737	044000	001124	1\$:	BIS	#44000,\$GDDAT	:SET UP ASSOCIATIVE NAME REG
1637	007244	013777	001124	172632		MOV	\$GDDAT,\$ANAME	:SET UP R0 WITH 77
1638	007252	012700	000077			MOV	#77,R0	:CLR MATCH FLAG & SEARCH ENA
1639	007256	042737	144000	001124	2\$:	BIC	#144000,\$GDDAT	:SET UP NAME INSTR
1640	007264	012737	150000	027526		MOV	#150000,BUFFER	:SET UP NAME VALUE
1641	007272	050037	027526			BIS	R0,BUFFER	:START
1642	007276	012777	027526	172560		MOV	#BUFFER,\$DPC	:SET UP REG ADRS 24
1643	007304	013737	002110	001122		MOV	DNAME,\$BADDR	:ALLOW TIME FOR NPR
1644	007312	000240				NOP		:READ NAME REG
1645	007314	017737	172570	001126		MOV	\$DNAME,\$BDDAT	:BR IF A MATCH OCCURED
1646	007322	100410				BMI	3\$:SHOULD A MATCH HAVE OCCURED?
1647	007324	023737	001124	001126		CMP	\$GDDAT,\$BDDAT	:BR IF NOT
1648	007332	001017				BNE	4\$:INDICATE IT SHOULD BE SET
1649	007334	052737	100000	001124		BIS	#100000,\$GDDAT	:NAME MATCH FLAG FAILED TO SET
1650	007342	104024				ERROR	24	:CLR OUT MATCH FLAG BIT
1651	007344	042737	100000	001126	3\$:	BIC	#100000,\$BDDAT	:MAKE SURE THERE IS A MATCH
1652	007352	023737	001124	001126		CMP	\$GDDAT,\$BDDAT	:BR IF OK
1653	007360	001404				BEQ	4\$:RESTORE THE MATCH FLAG BIT
1654	007362	052737	100000	001126		BIS	#100000,\$BDDAT	:NAME MATCH FLAG SET IN ER
1655	007370	104024				ERROR	24	:ADVANCE NAME VALUE
1656	007372	005300			4\$:	DEC	R0	:BR IF MORE VALUES TO TEST
1657	007374	100330				BPL	2\$	

J03

MAINDEC-11-DZVSA-A VS60 INSTRUCTION TEST PART 1
DZVSA.P11 T47

MACY11 27(732) 20-SEP-76 09:57 PAGE 35

TEST THAT THE NAME MATCH FLAG WILL SET ON 11 BIT COMPARES

```
1658 007376 005337 001124          DEC      $GDDAT          ;ADVANCE VALUE FOR ASSO. NAME REG
1659 007402 022737 007777 001124    CMP      #7777,$GDDAT   ;HAVE ALL VALUES BEEN TESTED?
1660 007410 001312          BNE      15             ;BR IF NOT
1661
1662 ;*****
1663 ;*TEST 50      TEST THAT RESET WILL CLEAR THE NAME MATCH FLAG
1664 ;*****
1665 †TST50:  SCOPE
1666 007412 000004          MOV      #40,$TIMES     ;;DO 40 ITERATIONS
1667 007414 012737 000040 001164    MOV      #BIT13,$STKPT  ;SET MAINT SWITCH 2
1668 007422 012777 020000 172466    MOV      #5777,$ANAME   ;SET ANAME FOR 11 BIT COMPARE ON 3777
1669 007430 012777 057777 172446    MOV      #15377,$BUFFER ;SET UP NAME INSTR
1670 007436 012737 153777 027526    CLR      $GDDAT        ;EXPECT ALL ZEROS
1671 007444 005037 001124          MOV      #BUFFER,$DPC   ;START
1672 007450 012777 027526 172406    MOV      DNAME,$BDADR   ;SET UP REG ADRS 24
1673 007456 013737 002110 001122    RESET    ;CLR DPU NAME REG (MATCH)
1674 007464 000005          MOV      $DNAME,$BDDAT ;READ IT
1675 007466 017737 172416 001126    BEQ      TST51         ;NEXT TEST IF ALL CLEARED
1676 007474 001401          BEQ      TST51         ;RESET FAILED TO CLEAR NAME MATCH
1677 007476 104023          ERROR    23
1678
1679 ;*****
1680 ;*TEST 51      TEST THAT THE ASSOCIATIVE NAME REG DOES NOT CHANGE WITH ENA=0
1681 ;*****
1682 †TST51:  SCOPE
1683 007500 000004          MOV      #BIT13,$STKPT  ;SET MAINT SWITCH #2
1684 007502 012777 020000 172406    MOV      #340,$PSW      ;DON'T ALLOW INTERRUPT
1685 007510 012737 000340 177776    MOV      DNAME,$BDADR   ;SET UP REG 24 ADRS
1686 007516 013737 002110 001122    MOV      #5777,$ANAME   ;SET UP FOR 11 BIT CMP ON VALUE #3777
1687 007524 012777 057777 172352    MOV      #50000,$ANAME  ;NOW TRY WRITING 0 WITH ENA DISABLED
1688 007532 012777 050000 172344    MOV      #15377,$BUFFER ;SET UP NAME INSTR WITH VALUE #3777
1689 007540 012737 153777 027526    MOV      #BUFFER,$DPC   ;START
1690 007546 012777 027526 172310    MOV      #113777,$GDDAT ;EXPECT NAME MATCH ON 11 BIT CMP VALUE #3777
1691 007554 012737 113777 001124    MOV      $DNAME,$BDDAT ;GET DPU NAME REG CONTENTS
1692 007562 017737 172322 001126    CMP      $GDDAT,$BDDAT ;IS THERE A MATCH?
1693 007570 023737 001124 001126    BEQ      TST52         ;ADVANCE TO NEXT TEST IF SO
1694 007576 001401          BEQ      TST52         ;ASSOCIATIVE NAME CHANGE ENABLE ER
1695 007600 104024          ERROR    24
1696
1697 ;*****
1698 ;*TEST 52      TEST THAT AN INTERRUPT WILL OCCUR ON 11 BIT NAME MATCH COMPARES
1699 ;*****
1700 †TST52:  SCOPE
1701 007602 000004          MOV      #2,$TIMES     ;;DO 2 ITERATIONS
1702 007604 012737 000002 001164    MOV      #BIT13,$STKPT  ;SET MAINT SWITCH #2
1703 007612 012777 020000 172276    MOV      #340,$PSW      ;SET PSW TO 7
1704 007620 012737 000340 177776    MOV      #25,$LPERR     ;SET UP SCOPE LOOP ADRS
1705 007626 012737 007702 001110    MOV      $VECT1,R1      ;GET VECTOR ADRS
1706 007634 013701 001250          MOV      #35,14(R1)     ;SET UP NAME INT SER ADRS
1707 007640 012761 007756 000014    MOV      #340,16(R1)    ;SET UP PSW TO 7 ON INT
1708 007646 012761 000340 000016    MOV      #10077,$GDDAT  ;START WITH MAX VALUE
1709 007654 012737 010077 001124    MOV      #44000,$GDDAT  ;SET SEACH CHANGE & NAME CHANGE ENABLES
1710 007662 052737 044000 001124    BIS     #44000,$GDDAT   ;SET UP ASSOCIATIVE NAME REG
1711 007670 013777 001124 172206    MOV      $GDDAT,$ANAME  ;SET UP RO WITH MAX VALUE
1712 007676 012700 000077          MOV      #77,RO        ;CLR ENA BITS
1713 007702 042737 044000 001124    BIC     #44000,$GDDAT   ;SET UP NAME INSTR
1714 007710 012737 150000 027526    MOV      #150000,BUFFER ;SET UP NAME VALUE
1715 007716 050037 027526          BIS     RO,BUFFER      ;START
1716 007722 012777 027526 172134    MOV      #BUFFER,$DPC
```

K03

```

1714 007730 012737 002110 001122      MOV      DNAME,$BDDADR      ;SET UP REG ADRS 24
1715 007736 005037 177776          CLR      PSW                ;ALLOW INT TO OCCUR
1716 007742 017737 172142 001126      MOV      @DNAME,$BDDAT     ;READ THE NAME REG
1717 007750 100010          BPL      4$                ;BR IF NO MATCH
1718 007752 104047          ERROR   47                ;NAME MATCH FAILED TO INTERRUPT
1719 007754 000416          BR       5$                ;GO LOOK FOR LOOP ON TEST SWITCH
1720 007756 022626          3$:    CMP      (R6)+,(R6)+   ;FIX STK SINCE NO RTI
1721 007760 017737 172124 001126      MOV      @DNAME,$BDDAT     ;READ NAME REG
1722 007765 100401          BMI     4$                ;BR IF MATCH FLAG SET
1723 007770 104047          ERROR   47                ;NAME MATCH INTERRUPTED BUT NO FLAG
1724 007772 005300          4$:    DEC      R0           ;ADVANCE NAME VALUE IN R0
1725 007774 100342          BPL     2$                ;BR IF MORE VALUES TO TEST
1726 007776 005337 001124          DEC     $GDDAT            ;ADVANCE VALUE FOR ASSO. NAME REG
1727 010002 022737 007777 001124      CMP      #7777,$GDDAT      ;SEE IF ALL VALUES HAVE BEEN TESTED
1728 010010 001324          BNE     1$                ;BR IF NOT
1729 010012 004737 023134          5$:    JSR      PC,RSTVEC    ;RESTORE NAME MATCH VECTOR WITH HALT
1730
1731
1732
1733

```

```

;*****
;*TEST 53      LOAD #1 INTO LINE TYPE REGISTERS
;*****
†T53:

```

```

1734 010016 000004          SCOPE
1735 010020 012777 020000 172070      MOV      #BIT13,@STKPT     ;SET MAINT SWITCH #2
1736 010026 012777 100005 172000      MOV      #100005,@DBUF    ;LINE TYPE ENABLE =1 LINE TYPE =1
1737 010034 012737 000001 001124      MOV      #1,$GDDAT        ;LOAD EXPECTED
1738 010042 013777 002034 172014      MOV      DBUF,@DPC        ;LOAD DISPLAY P.C.
1739 010050 013737 002066 001122      MOV      SREG0,$BDDADR    ;SET UP REG ADRS 02
1740 010056 017737 172004 001126      MOV      @SREG0,$BDDAT    ;READ DISPLAY STATUS REGISTER
1741 010064 042737 177774 001126      BIC     #177774,$BDDAT    ;MASK TO BITS 1-0
1742 010072 023737 001124 001126      CMP     $GDDAT,$BDDAT     ;COMPARE EXPCT TO RCVD
1743 010100 001401          BEQ     T54               ;BR IF EQUAL
1744 010102 104005          ERROR   5                 ;LINE BIT 0 FAILED TO SET
1745
1746
1747

```

```

;*****
;*TEST 54      LOAD #2 INTO LINE TYPE REGISTERS
;*****
†T54:

```

```

1749 010104 000004          SCOPE
1750 010106 012777 020000 172002      MOV      #BIT13,@STKPT     ;SET MAINT SWITCH #2
1751 010114 012777 100006 171712      MOV      #100006,@DBUF    ;LINE TYPE ENABLE =1 LINE TYPE =2
1752 010122 012737 000002 001124      MOV      #2,$GDDAT        ;LOAD EXPECTED
1753 010130 013777 002034 171726      MOV      DBUF,@DPC        ;LOAD DISPLAY P.C.
1754 010136 013737 002066 001122      MOV      SREG0,$BDDADR    ;SET UP REG ADRS 02
1755 010144 017737 171716 001126      MOV      @SREG0,$BDDAT    ;READ DISPLAY STATUS REGISTER
1756 010152 042737 177774 001126      BIC     #177774,$BDDAT    ;MASK TO BITS 1-0
1757 010160 023737 001124 001126      CMP     $GDDAT,$BDDAT     ;COMPARE EXPCT TO RCVD
1758 010166 001401          BEQ     T55               ;BR IF EQUAL
1759 010170 104005          ERROR   5                 ;LINE BIT 1 FAILED TO SET
1760
1761
1762

```

```

;*****
;*TEST 55      CHECK LINE TYPE ENABLE GATE
;*****
†T55:

```

```

1765 010172 000004          SCOPE
1766 010174 012777 020000 171714      MOV      #BIT13,@STKPT     ;SET MAINT SWITCH #2
1767 010202 012777 100007 171624      MOV      #100007,@DBUF    ;LINE TYPE ENABLE =1 LINE TYPE =3
1768 010210 013777 002034 171646      MOV      DBUF,@DPC        ;LOAD DISPLAY P.C.
1769 010216 012777 100000 171610      MOV      #100000,@DBUF

```

```

1770 010224 012737 000003 001124      MOV      #3,$GDDAT          ;LOAD EXPECTED
1771 010232 013777 002034 171624      MOV      DBUF,ADPC
1772 010240 013737 002066 001122      MOV      SREG0,$BDADR     ;SET UP REG ADRS 02
1773 010246 017737 171614 001126      MOV      ASREG0,$BDDAT    ;READ DISPLAY STATUS REGISTER
1774 010254 042737 177774 001126      BIC      #177774,$BDDAT   ;MASK TO BITS 1-0
1775 010262 023737 001124 001126      CMP      $GDDAT,$BDDAT    ;COMPARE EXPCT TO RCVD
1776 010270 001401          BEQ      TST56             ;;BR IF EQUAL
1777 010272 104005          ERROR    5                ;LINE TYPE ENABLE FAILED TO INHIBIT
1778                                     ;CHANGING OF LINETYPE VALUE
1779

```

```

;*****
;*TEST 56      LOAD #0 INTO BLINK REGISTERS
;*****
TST56: SCOPE

```

```

1783 010274 000004          MOV      #BIT13,ASTKPT     ;SET MAINT SWITCH #2
1784 010276 012777 020000 171612      MOV      #100020,ADBUF    ;BLINK ENABLE =1  BLINK =0
1785 010304 012777 100020 171522      CLR      $GDDAT          ;LOAD EXPECTED
1786 010312 005037 001124          MOV      DBUF,ADPC       ;LOAD DISPLAY P.C.
1787 010316 013777 002034 171540      MOV      SREG0,$BDADR     ;SET UP REG ADRS 02
1788 010324 013737 002066 001122      MOV      ASREG0,$BDDAT    ;READ DISPLAY STATUS REGISTER
1789 010332 017737 171530 001126      BIC      #177767,$BDDAT   ;MASK TO BIT 3
1790 010340 042737 177767 001126      CMP      $GDDAT,$BDDAT    ;COMPARE EXPCT TO RCVD
1791 010346 023737 001124 001126      BEQ      TST57             ;;BR IF EQUAL
1792 010354 001401          ERROR    27              ;BLINK BIT FAILED TO RESET
1793 010356 104027
1794

```

```

;*****
;*TEST 57      LOAD #1 INTO BLINK REGISTERS
;*****
TST57: SCOPE

```

```

1797 010360 000004          MOV      #BIT13,ASTKPT     ;SET MAINT SWITCH #2
1798 010362 012777 020000 171526      MOV      #100030,BUFFER   ;BLINK ENABLE =1  BLINK =1
1799 010370 012737 100030 027526      MOV      #10,$GDDAT       ;LOAD EXPECTED
1800 010376 012737 000010 001124      MOV      DBUF,ADPC       ;LOAD DISPLAY P.C.
1801 010404 013777 002034 171452      MOV      SREG0,$BDADR     ;SET UP REG ADRS 02
1802 010412 013737 002066 001122      MOV      ASREG0,$BDDAT    ;READ DISPLAY STATUS REGISTER
1803 010420 017737 171442 001126      BIC      #177767,$BDDAT   ;MASK TO BIT 3
1804 010426 042737 177767 001126      CMP      $GDDAT,$BDDAT    ;COMPARE EXPCT TO RCVD
1805 010434 023737 001124 001126      BEQ      TST60             ;;BR IF EQUAL
1806 010442 001401          ERROR    27              ;BLINK BIT FAILED TO SET
1807 010444 104027
1808

```

```

;*****
;*TEST 60      CHECK BLINK ENABLE GATE
;*****
TST60: SCOPE

```

```

1813 010446 000004          MOV      #BIT13,ASTKPT     ;SET MAINT SWITCH #2
1814 010450 012777 020000 171440      MOV      #100030,ADBUF    ;LOAD BLINK ON
1815 010456 012777 100030 171350      MOV      DBUF,ADPC
1816 010464 013777 002034 171372      MOV      #BIT3,$GDDAT     ;LOAD EXPECTED
1817 010472 012737 000010 001124      MOV      #100000,ADBUF    ;BLINK ENABLE =0  BLINK =0
1818 010500 012777 100000 171326      MOV      DBUF,ADPC       ;LOAD DISPLAY P.C.
1819 010506 013777 002034 171350      MOV      SREG0,$BDADR     ;SET UP REG ADRS 02
1820 010514 013737 002066 001122      MOV      ASREG0,$BDDAT    ;READ DISPLAY STATUS REGISTER
1821 010522 017737 171340 001126      BIC      #177767,$BDDAT   ;MASK TO BIT 3
1822 010530 042737 177767 001126      CMP      $GDDAT,$BDDAT    ;COMPARE EXPCT TO RCVD
1823 010536 023737 001124 001126      BEQ      TST61             ;;BR IF EQUAL
1824 010544 001401          ERROR    27              ;BLINK ENABLE FAILED TO INHIBIT
1825 010546 104027

```

M03

;CHANGING OF THE BLINK BIT

1826
1827
1828
1829
1830
1831 010550 000004
1832 010552 012777 020000 171336
1833 010560 012737 100100 001124
1834 010566 013777 001124 171240
1835 010574 013777 002034 171262
1836 010602 013737 002066 001122
1837 010610 017737 171252 001126
1838 010616 032737 000200 001126
1839 010624 001401
1840 010626 104034
1841
1842
1843
1844
1845 010630 000004
1846 010632 012777 020000 171256
1847 010640 012737 100140 001124
1848 010646 013777 001124 171160
1849 010654 013777 002034 171202
1850 010662 013737 002066 001122
1851 010670 017737 171172 001126
1852 010676 032737 000200 001126
1853 010704 001401
1854 010706 104034
1855
1856
1857
1858
1859 010710 000004
1860 010712 012777 020000 171176
1861 010720 012777 103000 171106
1862 010726 012737 002000 001124
1863 010734 013777 002034 171122
1864 010742 013737 002066 001122
1865 010750 017737 171112 001126
1866 010756 042737 174377 001126
1867 010764 023737 001124 001126
1868 010772 001401
1869 010774 104006
1870
1871
1872
1873
1874 010776 000004
1875 011000 012777 020000 171110
1876 011006 012777 102400 171020
1877 011014 012737 001000 001124
1878 011022 013777 002034 171034
1879 011030 013737 002066 001122
1880 011036 017737 171024 001126
1881 011044 042737 174377 001126

```
*****  
; *TEST 61 TEST FOR FALSE L.P. STATUS FLAG (L.P. - NOT ENABLED)  
*****  
TST61: SCOPE  
MOV #BIT13, @STKPT ;SET MAINT SWITCH #2  
MOV #100100, $GDDAT ;LP ENABLE =1 LP=0  
MOV $GDDAT, @DBUF ;LOAD BUFFER  
MOV DBUF, @DPC ;LOAD DISPLAY P.C.  
MOV SREG0, $BDADR ;SET UP REG ADRS 02  
MOV @SREG0, $BDDAT ;READ STATUS  
BIT #200, $BDDAT  
BEQ TST62 ;;BR IF EQUAL  
ERROR 34 ;LIGHT PEN FLAG SET IN ERROR  
  
*****  
; *TEST 62 TEST FOR FALSE L.P. STATUS FLAG (L.P. - ENABLED)  
*****  
TST62: SCOPE  
MOV #BIT13, @STKPT ;SET MAINT SWITCH #2  
MOV #100140, $GDDAT ;LP ENABLE =1 LP=1  
MOV $GDDAT, @DBUF ;LOAD BUFFER  
MOV DBUF, @DPC ;LOAD DISPLAY P.C.  
MOV SREG0, $BDADR ;SET UP REG ADRS 02  
MOV @SREG0, $BDDAT ;READ STATUS  
BIT #200, $BDDAT  
BEQ TST63 ;;BR IF EQUAL  
ERROR 34 ;LIGHT PEN FLAG SET IN ERROR  
  
*****  
; *TEST 63 LOAD #4 INTO INT LEVEL REGISTER  
*****  
TST63: SCOPE  
MOV #BIT13, @STKPT ;SET MAINT SWITCH #2  
MOV #103000, @DBUF ;INTENSITY LEVEL ENABLE =1 LEVEL =4  
MOV #2000, $GDDAT ;LOAD EXPECTED  
MOV DBUF, @DPC ;LOAD DISPLAY P.C.  
MOV SREG0, $BDADR ;SET UP REG ADRS 02  
MOV @SREG0, $BDDAT ;READ DISPLAY STATUS REGISTER  
BIC #174377, $BDDAT ;MASK TO BITS 8-10  
CMP $GDDAT, $BDDAT ;COMPARE EXPCT TO RCVD  
BEQ TST64 ;;BR IF EQUAL  
ERROR 6 ;INTENSITY LEVEL BIT 10 FAILED  
  
*****  
; *TEST 64 LOAD #2 INTO INT LEVEL REGISTER  
*****  
TST64: SCOPE  
MOV #BIT13, @STKPT ;SET MAINT SWITCH #2  
MOV #102400, @DBUF ;INTENSITY LEVEL ENABLE =1 LEVEL =2  
MOV #1000, $GDDAT ;LOAD EXPECTED  
MOV DBUF, @DPC ;LOAD DISPLAY P.C.  
MOV SREG0, $BDADR ;SET UP REG ADRS 02  
MOV @SREG0, $BDDAT ;READ DISPLAY STATUS REGISTER  
BIC #174377, $BDDAT ;MASK TO BITS 8-10
```

```

1882 011052 023737 001124 001126
1883 011060 001401
1884 011062 104006
1885
1886
1887
1888
1889 011064 000004
1890 011066 012777 020000 171022
1891 011074 012777 102200 170732
1892 011102 012737 000400 001124
1893 011110 013777 002034 170746
1894 011116 013737 002066 001122
1895 011124 017737 170736 001126
1896 011132 042737 174377 001126
1897 011140 023737 001124 001126
1898 011146 001401
1899 011150 104006
1900
1901
1902
1903
1904
1905 011152 000004
1906 011154 012777 020000 170734
1907 011162 012777 103200 170644
1908 011170 012777 100000 170640
1909 011176 013777 002034 170660
1910 011204 005277 170654
1911 011210 012737 002400 001124
1912 011216 013737 002066 001122
1913 011224 017737 170636 001126
1914 011232 042737 174377 001126
1915 011240 023737 001124 001126
1916 011246 001401
1917 011250 104006
1918
1919
1920
1921
1922
1923 011252 000004
1924 011254 012777 020000 170634
1925 011262 012777 170040 170544
1926 011270 005037 001124
1927 011274 013777 002034 170562
1928 011302 013737 002066 001122
1929 011310 017737 170552 001126
1930 011316 042737 177757 001126
1931 011324 023737 001124 001126
1932 011332 001401
1933 011334 104030
1934
1935
1936
1937

```

```

CMP $GDDAT,$BDDAT ;COMPARE EXPCT TO RCVD
BEQ TST65 ;:BR IF EQUAL
ERROR 6 ;INTENSITY LEVEL BIT 9 FAILED

;*****
;*TEST 65 LOAD #1 INTO INT LEVEL REGISTER
;*****
TST65: SCOPE
MOV #BIT13,2STKPT ;SET MAINT SWITCH #2
MOV #102200,2DBUF ;INTENSITY LEVEL ENABLE =1 LEVEL =1
MOV #400,$GDDAT ;LOAD EXPECTED
MOV DBUF,2DPC ;LOAD DISPLAY P.C.
MOV SREG0,$BDADR ;SET UP REG ADRS 02
MOV 2SREG0,$BDDAT ;READ DISPLAY STATUS REGISTER
BIC #174377,$BDDAT ;MASK TO BITS 8-10
CMP $GDDAT,$BDDAT ;COMPARE EXPCT TO RCVD
BEQ TST66 ;:BR IF EQUAL
ERROR 6 ;INTENSITY LEVEL BIT 8 FAILED

;*****
;*TEST 66 CHECK INTENSITY LEVEL ENABLE GATE
;*****
TST66: SCOPE
MOV #BIT13,2STKPT ;SET MAINT SWITCH #2
MOV #103200,2DBUF ;LOAD LEVEL
MOV #100000,2DBUF1 ;INTENSITY LEVEL ENABLE =0 LEVEL =0
MOV DBUF,2DPC ;LOAD DISPLAY P.C.
INC 2DPC ;SINGLE STEP THE DISPLAY
MOV #2400,$GDDAT ;LOAD EXPECTED
MOV SREG0,$BDADR ;SET UP REG ADRS 02
MOV 2SREG0,$BDDAT ;READ DISPLAY STATUS REGISTER
BIC #174377,$BDDAT ;MASK TO BITS 8-10
CMP $GDDAT,$BDDAT ;COMPARE EXPCT TO RCVD
BEQ TST67 ;:BR IF EQUAL
ERROR 6 ;INTENSITY LEVEL ENABLE FAILED TO INHIBIT
;INTENSITY LEVEL CHANGE

;*****
;*TEST 67 LOAD #0 INTO ITALIC REGISTER
;*****
TST67: SCOPE
MOV #BIT13,2STKPT ;SET MAINT SWITCH #2
MOV #170040,2DBUF ;ITALICS ENABLE=1 ITALICS=0
CLR $GDDAT ;LOAD EXPECTED
MOV DBUF,2DPC ;LOAD DISPLAY P.C.
MOV SREG0,$BDADR ;SET UP REG ADRS 02
MOV 2SREG0,$BDDAT ;READ DISPLAY STATUS REGISTER
BIC #177757,$BDDAT ;MASK TO BIT 4
CMP $GDDAT,$BDDAT ;COMPARE EXPCT TO RCVD
BEQ TST70 ;:BR IF EQUAL
ERROR 30 ;ITALICS BIT FAILED TO RESET

;*****
;*TEST 70 LOAD #1 INTO ITALIC REGISTER

```

```

1938
1939 011336 000004
1940 011340 012777 020000 170550
1941 011346 012777 170060 170460
1942 011354 012737 000020 001124
1943 011362 013777 002034 170474
1944 011370 013737 002066 001122
1945 011376 017737 170464 001126
1946 011404 042737 177757 001126
1947 011412 023737 001124 001126
1948 011420 001401
1949 011422 104030

```

```

*****
†TST70: SCOPE
MOV #BIT13,JUSTKPT ;SET MAINT SWITCH #2
MOV #170060,JUSTBUF ;ITALICS ENABLE=1 ITALICS=1
MOV #BIT4,$GDDAT ;LOAD EXPECTED
MOV DBUF,JUSTPC ;LOAD DISPLAY P.C.
MOV SREG0,$BDADR ;SET UP REG ADRS 02
MOV JSREG0,$BDAT ;READ DISPLAY STATUS REGISTER
BIC #177757,$BDAT ;MASK TO BIT 4
CMP $GDDAT,$BDAT ;COMPARE EXPCT TO RCVD
BEQ TST71 ;;BR IF EQUAL
ERROR 30 ;ITALICS BIT FAILED TO SET

```

```

1955 011424 000004
1956 011426 012777 020000 170462
1957 011434 012777 170060 170372
1958 011442 012777 170000 170366
1959 011450 013777 002034 170406
1960 011456 012737 000020 001124
1961 011464 005277 170374
1962 011470 013737 002066 001122
1963 011476 017737 170364 001126
1964 011504 042737 177757 001126
1965 011512 023737 001124 001126
1966 011520 001401
1967 011522 104030

```

```

*****
; *TEST 71 CHECK ITALIC ENABLE GATE
*****
†TST71: SCOPE
MOV #BIT13,JUSTKPT ;SET MAINT SWITCH #2
MOV #170060,JUSTBUF ;LOAD ENABLE
MOV #170000,JUSTBUF1 ;ITALICS ENABLE=0 ITALICS=0
MOV DBUF,JUSTPC
MOV #BIT4,$GDDAT ;LOAD EXPECTED
INC JUSTPC ;SINGLE STEP THE DISPLAY
MOV SREG0,$BDADR ;SET UP REG ADRS 02
MOV JSREG0,$BDAT ;READ DISPLAY STATUS REGISTER
BIC #177757,$BDAT ;MASK TO BITS 4
CMP $GDDAT,$BDAT ;COMPARE EXPCT TO RCVD
BEQ TST72 ;;BR IF EQUAL
ERROR 30 ;ITALICS ENABLE FAILED TO INHIBIT
;CLEARING OF ITALICS BIT

```

```

1974 011524 000004
1975 011526 013737 002066 001122
1976 011534 012777 020000 170354
1977 011542 012737 172000 027526
1978 011550 013777 002034 170306
1979 011556 012737 100000 001124
1980 011572 100401
1981 011574 104037
1982 011576 012737 160000 027526
1983 011604 012777 027526 170252
1984 011612 005037 001124
1985 011616 017737 170244 001126
1986 011624 100001
1987 011626 104040

```

```

*****
; *TEST 72 TEST INTERNAL STOP FLAG CAN SET AND RESET
*****
†TST72: SCOPE
MOV SREG0,$BDADR ;SET JP REG ADRS 02
MOV #BIT13,JUSTKPT ;SET MAINT SWITCH #2
MOV #172000,BUFFER ;"STOP" BIT =1
MOV DBUF,JUSTPC ;LOAD DISPLAY P.C.
MOV #BIT15,$GDDAT ;LOAD EXPECTED
MOV JSREG0,$BDAT ;READ DISPLAY STATUS REGISTER
BMI IS ;;BR IF SET
ERROR 37 ;"INTERNAL-STOP" BIT FAILED TO SET
IS: MOV #160000,BUFFER ;LOAD "NOP"
MOV #BUFFER,JUSTPC ;START DISPLAY
CLR $GDDAT ;CLEAR EXPECTED
MOV JSREG0,$BDAT ;READ DISPLAY STATUS REGISTER
BPL TST73 ;;BR IF CLEARED
ERROR 40 ;"INTERNAL-STOP" BIT FAILED TO RESET

```

```

1992 011630 000004
1993 011632 012737 000010 001164

```

```

*****
; *TEST 73 TEST THAT RESET WILL CLEAR STOP, MODE, INTENSITY, ITALICS, BLINK & LINE
*****
†TST73: SCOPE
MOV #10,$TIMES ;;DO 10 ITERATIONS

```

C04

```

1994 011640 012777 020000 170250      MOV      #BIT13,STKPT      ;SET UP MAINT SW 2
1995 011646 012737 102637 027526      MOV      #102637,BUFFER  ;SET UP CHAR
1996 011654 012737 170060 027530      MOV      #170060,BUFFER+2 ;SET UP ITALICS INSTR
1997 011662 012777 027526 170174      MOV      #BUFFER,JDPC    ;START
1998 011670 012737 002000 001124      MOV      #2000,$GDDAT    ;EXPECT INTENSITY ONLY
1999 011676 005277 170162          INC      JDPC            ;PICK UP ITALICS INSTR
2000 011702 012737 172000 027532      MOV      #172000,BUFFER+4 ;SET UP STOP INSTR
2001 011710 005277 170150          INC      JDPC            ;PICK UP STOP INSTR
2002 011714 013737 002066 001122      MOV      SREG0,$BADDR    ;SET UP REG ADRS 02
2003 011722 000005          RESET          ;CLR REG 02
2004 011724 017737 170136 001126      MOV      %SREG0,$BDDAT   ;READ REG
2005 011732 023737 001124 001126      CMP      $GDDAT,$BDDAT   ;ALL BUT THE MSB OF INTENSITY SHOULD HAVE BEEN CLEARED
2006 011740 001401          BEQ      TST74          ;NEXT TEST IF OK
2007 011742 104032          ERROR      32          ;RESET FAILED TO SET UP REG 02

```

```

*****
;*TEST 74      TEST THAT INTERNAL STOP WILL CAUSE AN INTERRUPT
*****

```

```

2010
2011
2012 011744 000004          TST74: SCOPE
2013 011746 012737 000100 001164      MOV      #100,$TIMES    ;DO 100 ITERATIONS
2014 011754 012737 000074 001204      MOV      #STN-1,STESTN  ;SET TEST NUMBER IN MAIL BOX
2015 011762 012737 000340 177776      MOV      #340,PSW       ;HIGHEST PRIORITY
2016 011770 013737 002066 001122      MOV      SREG0,$BADDR   ;SET UP REG ADRS 02
2017 011776 012737 172000 001124      MOV      #172000,$GDDAT ;EXPECT STOP MODE AND INTENSITY LEVEL ONLY
2018 012004 013700 001250          MOV      $VECT1,RO      ;GET INTR ADRS
2019 012010 012710 012070          MOV      #25,(RO)       ;SET UP RETURN ADRS
2020 012014 012760 000340 000002      MOV      #340,2(RO)    ;HIGHEST PRIORITY ON INTR
2021 012022 012777 020040 170066      MOV      #20040,STKPT   ;SET MAINT SW 2
2022 012030 012737 173400 027526      MOV      #173400,BUFFER ;ENABLE STOP INTR + STJP
2023 012036 012777 027526 170020 1$: MOV      #BUFFER,JDPC   ;START
2024 012044 005037 177776          CLR      PSW           ;ALLOW INTR
2025 012050 021616          CMP      (SP),.SF)     ;FUMBLE
2026 012052 005700          TST      RO           ;EXPECT INTR?
2027 012054 001430          BEQ      $$           ;BR IF NOT
2028 012056 017737 170004 001126      MOV      %SREG0,$BDDAT ;READ STATUS
2029 012064 104000          ERROR          ;INTERNAL STOP FAILED TO INTR
2030 012066 000423          BR      $$           ;LOOK FOR LOOP ON TEST SW
2031 012070 022626          CMP      (SP)+,(SP)+  ;FIX STACK SINCE NO RTI
2032 012072 017737 167770 001126 2$: MOV      %SREG0,$BDDAT ;READ REG 02
2033 012100 005700          TST      RO           ;EXPECT INTR?
2034 012102 001002          BNE     3$           ;BR IF 50
2035 012104 104033          ERROR          ;INTERNAL STOP INTR'ED WHEN NOT ENABLED
2036 012106 000413          BR      5$           ;GO LOOK FOR LOOP ON TEST SW
2037 012110 023737 001124 001126 3$: CMP      $GDDAT,$BDDAT ;STATUS CORRECT?
2038 012116 001402          BEQ     4$           ;BR IF 50
2039 012120 104033          ERROR          ;STOP FLAG NOT SET ON INTR
2040 012122 000405          BR      5$           ;GO LOOK FOR LOOP ON TEST SW
2041 012124 012737 173000 027526 4$: MOV      #173000,BUFFER ;SET UP NOW WITH INTR NOT ENABLED
2042 012132 005000          CLR      RO           ;INDICATE NO INTR
2043 012134 000740          BR      1$           ;REPEAT TEST
2044 012136 004737 023134 5$: JSR      PC,RSTVEC    ;GO RESTORE VECTORS

```

```

*****
;*TEST 75      LOAD #0 INTO MENU REGISTER
*****
TST75: SCOPE

```

```

2049 012142 000004

```

```

2050 012144 012777 020000 167744 MOV #BIT13,2STKPT ;SET MAINT SWITCH #2
2051 012152 012777 170002 167654 MOV #170002,2DBUF ;MENU ENABLE=1 MENU=0
2052 012160 005037 001124 CLR $GDDAT ;LOAD EXPECTED
2053 012164 013777 002034 167672 MOV DBUF,2DPC ;LOAD DISPLAY P.C.
2054 012172 013737 002076 001122 MOV SREG1,$B0ADR ;SET UP REG ADRS12
2055 012200 017737 167672 001126 MOV 2SREG1,$BDDAT ;READ DISP STATUS REGISTER
2056 012206 042737 177677 001126 BIC #177677,$BDDAT ;MASK TO BIT 2
2057 012214 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPCT TO RCVD
2058 012222 001401 BEQ TST76 ;;BR IF EQUAL
2059 012224 104031 ERROR 31 ;MENU BIT FAILED TO RESET

```

```

*****
;*TEST 76 LOAD #1 INTO MENU REGISTER
*****

```

```

2065 012226 000004 TST76: SCOPE
2066 012230 012777 020000 167660 MOV #BIT13,2STKPT ;SET MAINT SWITCH #2
2067 012236 012777 170003 167570 MOV #170003,2DBUF ;MENU ENABLE=1 MENU=1
2068 012244 012737 000100 001124 MOV #BIT6,$GDDAT ;LOAD EXPECTED
2069 012252 013777 002034 167604 MOV DBUF,2DPC ;LOAD DISPLAY P.C.
2070 012260 013737 002076 001122 MOV SREG1,$B0ADR ;SET UP REG ADRS 12
2071 012266 017737 167504 001126 MOV 2SREG1,$BDDAT ;READ DISPLAY STATUS REGISTER
2072 012274 042737 177677 001126 BIC #177677,$BDDAT ;MASK TO BIT 2
2073 012302 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPCT TO RCVD
2074 012310 001401 BEQ TST77 ;;BR IF EQUAL
2075 012312 104031 ERROR 31 ;MENU BIT FAILED TO SET

```

```

*****
;*TEST 77 CHECK MENU ENABLE GATE
*****

```

```

2081 012314 000004 TST77: SCOPE
2082 012316 013737 002076 001122 MOV SREG1,$B0ADR ;SET UP REG ADRS 12
2083 012324 012777 020000 167564 MOV #BIT13,2STKPT ;SET MAINT SWITCH #2
2084 012332 012737 170002 027526 MOV #170002,BUFFER ;MENU ENABLE = 1 MENU = 0
2085 012340 012777 027526 167516 MOV #BUFFER,2DPC ;START DISPLAY
2086 012346 012737 170001 027530 MOV #170001,BUFFER+2 ;MENU ENABLE=0 MENU=1
2087 012354 005277 167504 INC 2DPC ;RESUME
2088 012360 005037 001124 CLR $GDDAT ;LOAD EXPECTED
2089 012364 017737 167506 001126 MOV 2SREG1,$BDDAT ;READ DISPLAY STATUS REGISTER
2090 012372 042737 177677 001126 BIC #177677,$BDDAT ;MASK TO BIT 2
2091 012400 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPCT TO RCVD
2092 012406 001401 BEQ TST100 ;;BR IF EQUAL
2093 012410 104031 ERROR 31 ;MENU ENABLE FAILED TO INHIBIT
;RESETTING OF MENU BIT

```

```

*****
;*TEST 100 TEST THAT LOAD STATUS C INSTR CAN SET THE VECTOR SCALE
*****

```

```

2099 012412 000004 TST100: SCOPE
2100 012414 012777 020000 167474 MOV #BIT13,2STKPT ;SET MAINT SWITCH #2
2101 012422 012737 012436 001110 MOV #15,$LPERR ;SET UP SCOPE LOOP ADRS
2102 012430 012737 000017 001124 MOV #17,$GDDAT ;START WITH ALL ONES
2103 012436 012737 154020 027526 1$: MOV #154020,BUFFER ;SET UP INSTR AT BUFFER
2104 012444 053737 001124 027526 BIS $GDDAT,BUFFER ;SET UP VECTOR SCALE AT INSTR LOC
2105 012452 012777 027526 167404 MOV #BUFFER,2DPC ;START

```

E04

```

2106 012460 013737 002076 001122      MOV      SREG1,$B0ADR      ;SET UP REG ADRS 12
2107 012466 017737 167404 001126      MOV      JSREG1,$BDDAT    ;READ VECTOR SCALE
2108 012474 042737 177760 001126      BIC      #177760,$BDDAT   ;ONLY INTERESTED IN VECTOR SCALE
2109 012502 023737 001124 001126      CMP      $GDDAT,$BDDAT   ;ARE THEY CORRECT?
2110 012510 001401                BEQ      25               ;BR IF OK
2111 012512 104025                ERROR    25               ;VECTOR SCALE INSTR ER
2112 012514 005337 001124 25:      DEC      $GDDAT          ;ADVANCE PATTERN
2113 012520 100346                BPL      15               ;BR IF NOT DONE

```

```

;*****
;*TEST 101      TEST THAT THE VECTOR SCALE DOES NOT CHANGE WHEN CHANGE ENA=C
;*****

```

```

†TST101: SCOPE
2119 012522 000004                MOV      #BIT13,$STKPT    ;SET MAINT SWITCH #2
2120 012524 012777 020000 167364      MOV      #154037,BUFFER  ;SET UP INSTR AT BUFFER-VECTOR SCALE ENA SET
2121 012532 012737 154037 027526      MOV      #17,$GDDAT      ;SET UP VECTOR SCALE EXPECTED
2122 012540 012737 000017 001124      MOV      #BUFFER,$DPC    ;START
2123 012546 012777 027526 167310      MOV      #154000,BUFFER+2 ;SET UP INSTR AT BUFFER-NO VECTOR SCALE ENA
2124 012554 012737 154000 027530      INC      $DPC            ;RESUME
2125 012562 005277 167276                MOV      SREG1,$B0ADR    ;SET UP REG ADRS 12
2126 012566 013737 002076 001122      MOV      JSREG1,$BDDAT   ;READ VECTOR SCALE
2127 012574 017737 167276 001126      BIC      #177760,$BDDAT  ;SAVE VECTOR SCALE ONLY
2128 012602 042737 177760 001126      CMP      $GDDAT,$BDDAT  ;DOES IT STILL EQ 17
2129 012610 023737 001124 001126      BEQ      TST102          ;ADVANCE TO NEXT TEST IF OK
2130 012616 001401                ERROR    25               ;VECTOR SCALE CHANGE ENABLE INSTR ER
2131 012620 104025

```

```

;*****
;*TEST 102      TEST THAT LOAD STATUS C INSTR CAN SET THE CHARACTER SCALE
;*****

```

```

†TST102: SCOPE
2135 012622 000004                MOV      #BIT13,$STKPT    ;SET MAINT SWITCH #2
2136 012624 012777 020000 167264      MOV      #15,$LPERR      ;SET UP SCOPE LOOP ADRS
2137 012632 012737 012654 001110      MOV      #154340,BUFFER  ;SET UP INSTR AT BUFFER
2138 012640 012737 154340 027526      MOV      #1400,$GDDAT    ;START WITH ALL ONES
2139 012646 012737 001400 001124      MOV      #BUFFER,$DPC    ;START
2140 012654 012777 027526 167202 15:      MOV      SREG1,$B0ADR    ;SET UP REG ADRS 12
2141 012662 013737 002076 001122      MOV      JSREG1,$BDDAT   ;READ CHAR SCALE
2142 012670 017737 167202 001126      BIC      #176377,$BDDAT  ;SAVE CHAR SCALE ONLY
2143 012676 042737 176377 001126      CMP      $GDDAT,$BDDAT  ;IS IT CORRECT?
2144 012704 023737 001124 001126      BEQ      25               ;BR IF OK
2145 012712 001402                BEQ      TST103          ;ADVANCE TO NEXT TEST IF OK
2146 012714 001411                ERROR    26               ;CHARACTER SCALE INSTR ER
2147 012716 104026                SUB      #400,$GDDAT     ;ADVANCE PATTERN
2148 012720 162737 000400 001124 25:      BMI      TST103          ;ADVANCE TO NEXT TEST IF DONE
2149 012726 100404                SUB      #40,BUFFER      ;ADVANCE PATTERN AT INSTR LOC
2150 012730 162737 000040 027526      BR       15               ;DO NEXT CHAR SCALE
2151 012736 000746

```

```

;*****
;*TEST 103      TEST THAT THE CHAR SCALE DOES NOT CHANGE WHEN CHANGE ENA=C
;*****

```

```

†TST103: SCOPE
2156 012740 000004                MOV      #BIT13,$STKPT    ;SET MAINT SWITCH #2
2157 012742 012777 020000 167146      MOV      #154340,BUFFER  ;SET UP INSTR AT BUFFER-CHAR SCALE ENA SET
2158 012750 012737 154340 027526      MOV      #1400,$GDDAT    ;SET UP CHAR SCALE EXPECTED
2159 012756 012737 001400 001124      MOV      #BUFFER,$DPC    ;START
2160 012764 012777 027526 167072      MOV      #154000,BUFFER+2 ;SET UP INSTR AT BUFFER-NO CHAR SCALE ENA
2161 012772 012737 154000 027530

```

F04

```

2162 013000 005277 167060          INC      20PC          ;RESUME
2163 013004 013737 002076 001122    MOV      SREG1,$BDADR ;SET UP REG ADRS 12
2164 013012 017737 167060 001126    MOV      2SREG1,$BDDAT ;READ CHAR SCALE
2165 013020 042737 176377 001126    BIC      #176377,$BDDAT ;SAVE CHAR SCALE ONLY
2166 013026 023737 001124 001126    CMP      $GDDAT,$BDDAT ;IS IT CORRECT?
2167 013034 001401          BEQ      TST104        ;GO TO NEXT TEST IF NO CHANGE
2168 013036 104026          ERROR    26           ;CHARACTER SCALE CHANGE ENABLE INSTR ER
  
```

```

;*****
;*TEST 104 TEST THAT LOAD STATUS C INSTR CAN SET THE CHARACTER ROTATE
;*****
TST104: SCOPE
  
```

```

2173 013040 000004          MOV      #BIT13,2STKPT ;SET MAINT SWITCH #2
2174 013042 012777 020000 167046    MOV      #1$, $LPERR   ;SET UP SCOPE LOOP ADRS
2175 013050 012737 013072 001110    MOV      #155400,BUFFER ;SET UP INSTR AT BUFFER
2176 013056 012737 155400 027526    MOV      #2000,$GDDAT  ;EXPECT CHAR ROTATE BIT
2177 013064 012737 002000 001124    MOV      #BUFFER,20PC  ;START
2178 013072 012777 027526 166764 1$: MOV      SREG1,$BDADR ;SET UP REG ADRS 12
2179 013100 013737 002076 001122    MOV      2SREG1,$BDDAT ;READ CHAR ROTATE BIT
2180 013106 017737 166764 001126    BIC      #175777,$BDDAT ;SAVE CHAR ROTATE BIT ONLY
2181 013114 042737 175777 001126    BIC      #175777,$BDDAT ;SAVE CHAR ROTATE BIT ONLY
2182 013122 023737 001124 001126    CMP      $GDDAT,$BDDAT ;IS IT CORRECT?
2183 013130 001401          BEQ      2$           ;BR IF OK
2184 013132 104026          ERROR    26           ;CHARACTER ROTATE INSTR ER
2185 013134 162737 002000 001124 2$: SUB      #2000,$GDDAT ;ADVANCE PATTERN
2186 013142 100404          BMI     TST105        ;ADVANCE TO NEXT TEST IF OK
2187 013144 162737 000400 027526    SUB      #400,BUFFER  ;ADVANCE PATTERN AT INSTR LOC
2188 013152 000747          BR      1$           ;NOW WRITE A ZERO INTO CHAR ROTATE
  
```

```

;*****
;*TEST 105 TEST THAT THE CHARACTER ROTATE DOES NOT CHANGE WHEN CHANGE ENA=0
;*****
TST105: SCOPE
  
```

```

2193 013154 000004          MOV      #BIT13,2STKPT ;SET MAINT SWITCH #2
2194 013156 012777 020000 166732    MOV      #155400,BUFFER ;SET UP INSTR AT BUFFER CHAR ROTATE ENA SET
2195 013164 012737 155400 027526    MOV      #2000,$GDDAT  ;EXPECT CHAR ROTATE BIT
2196 013172 012737 002000 001124    MOV      #BUFFER,20PC  ;START
2197 013200 012777 027526 166656    MOV      #154000,BUFFER+2 ;SET UP INSTR AT BUFFER-WITHOUT CHAR ROTATE ENA
2198 013206 012737 154000 027530    INC      20PC          ;RESUME
2199 013214 005277 166644          MOV      SREG1,$BDADR ;SET UP REG ADRS 12
2200 013220 013737 002076 001122    MOV      2SREG1,$BDDAT ;READ CHAR ROTATE BIT
2201 013226 017737 166644 001126    BIC      #175777,$BDDAT ;SAVE CHAR ROTATE BIT
2202 013234 042737 175777 001126    BIC      #175777,$BDDAT ;SAVE CHAR ROTATE BIT
2203 013242 023737 001124 001126    CMP      $GDDAT,$BDDAT ;IS IT SET?
2204 013250 001401          BEQ      TST106        ;ADVANCE TO NEXT IF OK
2205 013252 104026          ERROR    26           ;CHARACTER ROTATE CHANGE ENABLE INSTR ER
  
```

```

;*****
;*TEST 106 TEST THAT RESET WILL CLEAR ROTATE, CHAR SCALE, MENU & VECTOR SCALE
;*****
TST106: SCOPE
  
```

```

2211 013254 000004          MOV      #40,$TIMES   ;DO 40 ITERATIONS
2212 013256 012737 000040 001164    MOV      #BIT13,2STKPT ;SET MAINT SW 2
2213 013264 012777 020000 166624    MOV      #155733,BUFFER ;SET UP 'LOAD STATUS C' INSTR
2214 013272 012737 155733 027526    MOV      #170003,BUFFER+2 ;SET UP MENU INSTR
2215 013300 012737 170003 027530    MOV      #BUFFER,20PC  ;START
2216 013306 012777 027526 166550    MOV      #404,$GDDAT  ;EXPECT MSB OF VECTOR SCALE ONLY
2217 013314 012737 000404 001124
  
```

G04

2218	013322	005277	166536		INC	QDPC	: PICK UP MENU INSTR
2219	013326	013737	002076	001122	MOV	SREG1, \$BDADR	: SET UP REG ADRS 12
2220	013334	000005			RESET		: CLR REG 12
2221	013336	017737	166534	001126	MOV	QDPC, \$BDADR	: READ STATUS REG 1
2222	013344	023737	001124	001126	CMP	\$GDDAT, \$BDADR	: IS IT CORRECT?
2223	013352	001401			BEQ	TST107	: NEXT TEST IF OK
2224	013354	104032			ERROR	32	: RESET FAILED TO SET UP REG 12

; GRAPH PLOT INCREMENT REGISTER TESTS

 ; *TEST 107 LOAD 0-77 INTO GRAPH PLOT INCREMENT REGISTER

2231	013356	000004			TST107: SCOPE		
2232	013360	012737	013406	001110	MOV	#15, \$LPERR	: SET UP SCOPE LOOP ADRS
2233	013366	012777	020000	166522	MOV	#BIT13, \$STKPT	: SET MAINT SW 2
2234	013374	012737	174100	027526	MOV	#174100, \$BUFFER	: LOAD GRAPH PLOT INSTR
2235	013402	005037	001124		CLR	\$GDDAT	: LD EXPECTED
2236	013406	012777	027526	166450	1S: MOV	\$BUFFER, QDPC	: START
2237	013414	013737	002070	001122	MOV	XPOS, \$BDADR	: SET UP REG ADRS 04
2238	013422	017737	166442	001126	MOV	QXPOS, \$BDADR	: READ GRAPH PLOT
2239	013430	042737	001777	001126	BIC	#1777, \$BDADR	: SAVE ONLY GRAPH PLOT
2240	013436	023737	001124	001126	CMP	\$GDDAT, \$BDADR	: CORRECT?
2241	013444	001401			BEQ	25	: BR IF 50
2242	013446	104007			ERROR	7	: GRAPH PLOT FAILED TO LOAD CORRECTLY
2243	013450	005237	027526		2S: INC	\$BUFFER	: ADVANCE GRAPH PLOT AT INSTR
2244	013454	062737	002000	001124	ADD	#2000, \$GDDAT	: ADVANCE EXPECTED
2245	013462	103351			BCC	15	: BR IF NOT COMBS TESTED

 ; *TEST 110 CHECK THE GRAPH PLOT INCREMENT ENABLE REGISTER

2250	013464	000004			TST110: SCOPE		
2251	013466	013737	002070	001122	MOV	XPOS, \$BDADR	: SET UP REG ADRS 04
2252	013474	012777	020000	166414	MOV	#BIT13, \$STKPT	: SET MAINT SWITCH #2
2253	013502	012777	174100	166324	MOV	#174100, \$DBUF	: LOAD GRAPH PLOT COUNTER WITH 0
2254	013510	013777	002034	166346	MOV	\$DBUF, QDPC	: START DISPLAY
2255	013516	012737	174077	027530	MOV	#174077, \$BUFFER+2	: LOAD GRAPH PLOT NO ENABLE
2256	013524	005277	166334		INC	QDPC	: RESUME
2257	013530	005037	001124		CLR	\$GDDAT	: LOAD EXPECTED
2258	013534	017737	166330	001126	MOV	QXPOS, \$BDADR	: READ INCREMENT REGISTER
2259	013542	042737	001777	001126	BIC	#1777, \$BDADR	: MASK TO BITS 15-10
2260	013550	023737	001124	001126	CMP	\$GDDAT, \$BDADR	: COMPARE EXPCT TO RCVD
2261	013556	001401			BEQ	TST111	: BR IF EQUAL
2262	013560	104007			ERROR	7	: GRAPH PLOT REGISTER CHANGED WITHOUT THE ENABLE BEING SET

 ; *TEST 111 TEST THAT RESET WILL CLEAR GRAPH PLOT INCREMENT

2268	013562	000004			TST111: SCOPE		
2269	013564	012737	000040	001164	MOV	#40, \$TIMES	: DO 40 ITERATIONS
2270	013572	013737	002070	001122	MOV	XPOS, \$BDADR	: SET UP REG ADRS 04
2271	013600	012777	020000	166310	MOV	#BIT13, \$STKPT	: SET MAINT SW 2
2272	013606	012737	174177	027526	MOV	#174177, \$BUFFER	: SET UP GRAPH PLOT INCREMENT INSTR
2273	013614	012777	027526	166242	MOV	\$BUFFER, QDPC	: START

H04

```

2274 013622 005037 001124 CLR $GDDAT ;EXPECT ZERO
2275 013626 000005 RESET ;CLR REG 04
2276 013630 017737 166234 001126 MOV 2XPOS,$BDDAT ;READ REG 04
2277 013636 001401 BEQ TST112 ;NEXT TEST IF ZERO
2278 013640 104032 ERROR 32 ;RESET FAILED TO CLR GRAPHPLOT INC
2279
2280 ;*****
2281 ;*TEST 112 TEST THAT LOAD STATUS B INSTR CAN SET THE COLOR LEVEL
2282 ;*****
2283
2284 013642 000004 TST112: SCOPE
2285 013644 012777 020000 166244 MOV #BIT13,2STKPT ;SET MAINT SWITCH #2
2286 013652 012737 013674 001110 MOV #15,$LPERR ;SET UP SCOPE LOOP ADRS
2287 013660 012737 175600 027526 MOV #175600,BUFFER ;SET UP INSTR AT BUFFER
2288 013666 012737 000014 001124 MOV #14,$GDDAT ;EXPECT ALL ONES AT START
2289 013674 012777 027526 166162 15: MOV #BUFFER,2DPC ;START
2290 013702 013737 002106 001122 MOV CONS,$B0ADR ;SET UP REG ADRS 22
2291 013710 017737 166172 001126 MOV 2CONS,$BDDAT ;READ COLOR LEVEL
2292 013716 042737 177763 001126 BIC #177763,$BDDAT ;ONLY INTERESTED IN THE COLOR LEVEL
2293 013724 023737 001124 001126 CMP $GDDAT,$BDDAT ;IS IT CORRECT?
2294 013734 104035 BEQ 25 ;BR IF OK
2295 013736 162737 000004 001124 25: SUB #4,$GDDAT ;COLOR LEVEL INSTR ER
2296 013744 100404 BMI TST113 ;ADVANCE PATTERN
2297 013746 162737 000200 027526 SUB #200,BUFFER ;ADVANCE TO NEXT TEST IF ALL PATTERNS TESTED
2298 013754 000747 BR 15 ;SET UP NEXT PATTERN AT INSTR LOC
2299 ;TRY NEXT PATTERN
2300
2301 ;*****
2302 ;*TEST 113 TEST THAT THE COLOR LEVEL DOES NOT CHANGE WHEN CHANGE ENA=0
2303 ;*****
2304
2305 013756 000004 TST113: SCOPE
2306 013760 012777 020000 166130 MOV #BIT13,2STKPT ;SET MAINT SWITCH #2
2307 013766 012737 175600 027526 MOV #175600,BUFFER ;SET UP INSTR AT BUFFER-COLOR LEVEL ENA SET
2308 013774 012737 000014 001124 MOV #14,$GDDAT ;SET UP COLOR LEVEL EXPECTED
2309 014002 012777 027526 166054 MOV #BUFFER,2DPC ;START
2310 014010 012737 174000 027530 MOV #174000,BUFFER+2 ;SET UP INSTR AT BUFFER-NO COLOR LEVEL ENA
2311 014016 005277 166042 INC 2DPC ;RESUME
2312 014022 013737 002106 001122 MOV CONS,$B0ADR ;SET UP REG ADRS 22
2313 014030 017737 166052 001126 MOV 2CONS,$BDDAT ;READ COLOR LEVEL
2314 014036 042737 177763 001126 BIC #177763,$BDDAT ;ONLY INTERESTED IN THE COLOR LEVEL
2315 014044 023737 001124 001126 CMP $GDDAT,$BDDAT ;IS IT CORRECT?
2316 014052 001401 BEQ TST114 ;ADVANCE TO NEXT TEST IF OK
2317 014054 104035 ERROR 35 ;COLOR LEVEL CHANGE ENABLE INSTR ER
2318
2319 ;*****
2320 ;*TEST 114 TEST THAT NOP INSTR CAN SET & RESET INTENSITY ENABLED-CONSOLE 0
2321 ;*****
2322
2323 014056 000004 TST114: SCOPE
2324 014060 012777 020000 166030 MOV #BIT13,2STKPT ;SET MAINT SWITCH #2
2325 014066 012737 014110 001110 MOV #15,$LPERR ;SET UP SCOPE LOOP ADRS
2326 014074 012737 100000 001124 MOV #100000,$GDDAT ;EXPECT IT TO BE SET INITIALLY
2327 014102 012737 164300 027526 MOV #164300,BUFFER ;SET UP INSTR LOC
2328 014110 012777 027526 165746 15: MOV #BUFFER,2DPC ;START
2329 014116 013737 002106 001122 MOV CONS,$B0ADR ;SET UP REG ADRS 22
2330 014124 017737 165756 001126 MOV 2CONS,$BDDAT ;READ CONSOLE 0 INT ENA REG
2331 014132 042737 077777 001126 BIC #77777,$BDDAT ;SAVE ONLY INTENSITY ENABLE BIT
2332 014140 023737 001124 001126 CMP $GDDAT,$BDDAT ;IS IT CORRECT?

```

```

2330 014146 001401          BEQ      25          ;BR IF 50
2331 014150 104036          ERROR    36          ;CONSOLE 0 INTENSITY ENABLE INSTR ER
2332 014152 162737 100000 001124 25:  SUB      #100000,$GDDAT AT ;GO TO RESET STATE
2333 014160 100404          BMI      TST115      ;GO TO NEXT TEST IF BOTH STATES TESTED
2334 014162 042737 000100 027526  BIC      #100,BUFFER  ;CLR INT ENABLED BIT AT INSTR LOC
2335 014170 000747          BR       15          ;GO TRY RESET STATE
2336
2337
2338 ;*****
2339 ;*TEST 115      TEST THAT CONSOLE 0 INTENSITY ENABLED DOES NOT RESET WHEN ENA=0
2340 ;*****
2341 †TST115: SCOPE
2342 014172 000004          MOV      #BIT13,2STKPT ;SET MAINT SWITCH #2
2343 014174 312777 020000 165714  MOV      #100000,$GDDAT ;EXPECT INTENSITY TO BE SET
2344 014202 012737 100000 001124  MOV      #164300,BUFFER ;SET UP INSTR
2345 014210 012737 164300 027526  MOV      #BUFFER,2DPC  ;START
2346 014216 012777 027526 165640  MOV      #164000,BUFFER+2 ;SET UP INSTR WITH ENA=0
2347 014224 012737 164000 027530  INC      2DPC          ;ADVANCE TO NEXT INSTR
2348 014232 005277 165626          MOV      CONS,$BDADR  ;SET UP REG ADRS 22
2349 014236 013737 002106 001122  TST     2CONS         ;IS INTENSITY STILL SET?
2350 014244 005777 165636          BMI      TST116      ;ADVANCE TO NEXT TEST IF STILL SET
2351 014250 100403          CLR     $BDDAT       ;INDICATE IT WAS CLEARED
2352 014252 005037 001126          ERROR   36          ;CONSOLE 0 INTENSITY CHANGE ENABLE INSTR ER
2353
2354 ;*****
2355 ;*TEST 116      TEST THAT NOP INSTR CAN SET & RESET INTENSITY ENABLED-CONSOLE 1
2356 ;*****
2357 †TST116: SCOPE
2358 014260 000004          MOV      #BIT13,2STKPT ;SET MAINT SWITCH #2
2359 014262 012777 020000 165626  MOV      15,$LPERR    ;SET UP SCOPE LOOP ADRS
2360 014270 013737 014312 001110  MOV      #1000,$GDDAT ;EXPECT IT TO BE SET INITIALLY
2361 014276 012737 001000 001124  MOV      #164700,BUFFER ;SET UP INSTR LOCATION
2362 014304 012737 164700 027526  15:  MOV      #BUFFER,2DPC  ;START
2363 014312 012777 027526 165544  MOV      CONS,$BDADR  ;SET UP REG ADRS 22
2364 014320 013737 002106 001122  MOV      2CONS,$BDDAT ;READ CONSOLE 1 INT ENA REG
2365 014326 017737 165554 001126  BIC     #176777,$BDDAT ;SAVE ONLY INT ENA
2366 014334 042737 176777 001126  CMP     $GDDAT,$BDDAT ;IS IT CORRECT?
2367 014342 023737 001124 001126  BEQ     25          ;BR IF OK
2368 014350 001401          ERROR   36          ;CONSOLE 1 INTENSITY ENABLE INSTR ER
2369 014352 104036          SUB     #1000,$GDDAT  ;GO TO RESET STATE
2370 014354 162737 001000 001124 25:  BMI     TST117      ;GO TO NEXT TEST IF BOTH STATES TESTED
2371 014362 100404          BIC     #100,BUFFER  ;CLR INT ENABLED BIT AT INSTR LOC
2372 014364 042737 000100 027526  BR      15          ;GO TRY RESET STATE
2373
2374 ;*****
2375 ;*TEST 117      TEST THAT CONSOLE 1 INTENSITY ENABLED DOES NOT RESET WHEN ENA=0
2376 ;*****
2377 †TST117: SCOPE
2378 014374 000004          MOV      #BIT13,2STKPT ;SET MAINT SWITCH #2
2379 014376 012777 020000 165512  MOV      #1000,$GDDAT ;EXPECT INTENSITY TO BE SET
2380 014404 012737 001000 001124  MOV      #164700,BUFFER ;SET UP INSTR
2381 014412 012737 164700 027526  MOV      #BUFFER,2DPC  ;START
2382 014420 012777 027526 165436  MOV      #164400,BUFFER+2 ;SET UP INSTR WITH ENA=0
2383 014426 012737 164400 027530  INC     2DPC          ;ADVANCE TO NEXT INSTR
2384 014434 005277 165424          MOV     CONS,$BDADR  ;SET UP REG ADRS 22
2385 014440 013737 002106 001122  BIT     #1000,2CONS   ;IS INTENSITY STILL SET?
2386 014446 032777 001000 165432  BNE    TST120      ;ADVANCE TO NEXT TEST IF STILL SET
2387 014454 001003

```

2386 014456 005037 001126 CLR \$BDDAT ;INDICATE IT WAS CLEARED
2387 014462 104036 ERROR 36 ;CONSOLE 1 INTENSITY CHANGE ENABLE INSTR ER

2399
2399
2399
2390
2391
2392 014464 000004
2393 014466 012777 020000 165422
2394 014474 012737 014516 001110
2395 014502 012737 004000 001124
2396 014510 012737 164060 027526
2397 014516 012777 027526 165340 1S:
2398 014524 013737 002106 001122
2399 014532 017737 165350 001126
2400 014540 042737 173777 001126
2401 014546 023737 001124 001126
2402 014554 001401
2403 014556 104042
2404 014560 162737 004000 001124 2S:
2405 014566 100404
2406 014570 042737 000020 027526
2407 014576 000747
2408
2409

; *TEST 120 TEST THAT NOP INSTR CAN SET & RESET L.P. INTR ENABLED-CONSOLE 0

↑ST120: SCOPE
MOV #BIT13, 2STKPT ;SET MAINT SWITCH #2
MOV #15, \$LPERR ;SET UP SCOPE LOOP ADRS
MOV #4000, \$GDDAT ;EXPECT IT TO BE SET INITIALLY
MOV #164060, BUFFER ;SET UP INSTR LOC
MOV #BUFFER, 2DPC ;START
MOV CONS, \$B0ADR ;SET UP REG ADRS 22
MOV 2CONS, \$BDDAT ;READ CONSOLE 0 LP INTR ENABLED REG
BIC #173777, \$BDDAT ;SAVE ONLY LP INTR ENABLED BIT
CMP \$GDDAT, \$BDDAT ;IS IT CORRECT?
BEQ 2S ;BR IF OK
ERROR 42 ;CONSOLE 0 L.P. INTR ENABLE INSTR ER
SUB #4000, \$GDDAT ;GO TO RESET STATE
BMI TST121 ;ADVANCE TO NEXT TEST IF BOTH STATES TESTED
BIC #20, BUFFER ;CLR THE ENABLED BIT AT INSTR LOC
BR 1S ;GO TRY RESET STATE

2410
2411
2412 014600 000004
2413 014602 012777 020000 165306
2414 014610 012737 004000 001124
2415 014616 012737 164060 027526
2416 014624 012777 027526 165232
2417 014632 012737 164000 027530
2418 014640 005277 165220
2419 014644 013737 002106 001122
2420 014652 032777 004000 165226
2421 014660 001003
2422 014662 005037 001126
2423 014666 104042
2424
2425

; *TEST 121 TEST THAT CONSOLE 0 L.P. INTR ENABLED DOES NOT RESET WHEN ENA=0

↑ST121: SCOPE
MOV #BIT13, 2STKPT ;SET MAINT SWITCH #2
MOV #4000, \$GDDAT ;EXPECT L.P. INTR ENABLED TO BE SET
MOV #164060, BUFFER ;SET UP INSTR LOC
MOV #BUFFER, 2DPC ;START
MOV #164000, BUFFER+2 ;SET UP INSTR WITH ENA=0
INC 2DPC ;ADVANCE TO NEXT INSTR
MOV CONS, \$B0ADR ;SET UP REG ADRS 22
BIT #4000, 2CONS ;IS L.P. INTR ENABLED STILL SET?
BNE TST122 ;ADVANCE TO NEXT TEST IF STILL SET
CLR \$BDDAT ;INDICATE IT WAS CLEARED
ERROR 42 ;CONSOLE 0 L.P. CHANGE INTR ENABLE INSTR ER

2426
2427
2428 014670 000004
2429 014672 012777 020000 165216
2430 014700 012737 014722 001110
2431 014706 012737 000040 001124
2432 014714 012737 164460 027526
2433 014722 012777 027526 165134 1S:
2434 014730 013737 002106 001122
2435 014736 017737 165144 001126
2436 014744 042737 177737 001126
2437 014752 023737 001124 001126
2438 014760 001401
2439 014762 104042
2440 014764 162737 000040 001124 2S:
2441 014772 100404

; *TEST 122 TEST THAT NOP INSTR CAN SET & RESET L.P. INTR ENABLED-CONSOLE 1

↑ST122: SCOPE
MOV #BIT13, 2STKPT ;SET MAINT SWITCH #2
MOV #15, \$LPERR ;SET UP SCOPE LOOP ADRS
MOV #40, \$GDDAT ;EXPECT IT TO BE SET INITIALLY
MOV #164460, BUFFER ;SET UP INSTR LOC
MOV #BUFFER, 2DPC ;START
MOV CONS, \$B0ADR ;SET UP REG ADRS 22
MOV 2CONS, \$BDDAT ;READ CONSOLE 1 LP INTR ENABLED REG
BIC #177737, \$BDDAT ;SAVE ONLY LP INTR ENABLED BIT
CMP \$GDDAT, \$BDDAT ;IS IT CORRECT?
BEQ 2S ;BR IF OK
ERROR 42 ;CONSOLE 1 L.P. INTR ENABLE INSTR ER
SUB #40, \$GDDAT ;GO TO RESET STATE
BMI TST123 ;ADVANCE TO NEXT TEST IF BOTH STATES TESTED

K04

```

2442 014774 042737 000020 027526      BIC      #20,BUFFER      ;CLR THE ENABLED BIT AT INSTR LOC
2443 015002 000747                      BR        1$            ;GO TRY RESET STATE
2444
2445 ;*****
2446 ;*TEST 123      TEST THAT CONSOLE 1 L.P. INTR ENABLED DOES NOT RESET WHEN ENA=0
2447 ;*****
2448 015004 000004      TST123: SCOPE
2449 015006 012777 020000 165102      MOV      #BIT13,2STKPT      ;SET MAINT SWITCH #2
2450 015014 012737 000040 001124      MOV      #40,$GDDAT      ;EXPECT L.P. INTR ENABLED TO BE SET
2451 015022 012737 164460 027526      MOV      #164460,BUFFER    ;SET UP INSTR LOC
2452 015030 012777 027526 165026      MOV      #BUFFER,2DPC      ;START
2453 015036 012737 164400 027530      MOV      #164400,BUFFER+2  ;SET UP INSTR WITH ENA=0
2454 015044 005277 165014      INC      2DPC              ;ADVANCE TO NEXT INSTR
2455 015050 013737 002106 001122      MOV      CONS,$BDADR      ;SET UP REG ADRS 22
2456 015055 032777 000040 165022      BIT      #40,2CONS        ;IS L.P. INTR ENABLED STILL SET?
2457 015064 001003                      BNE      TST124          ;ADVANCE TO NEXT TEST IF STILL SET
2458 015066 005037 001126      CLR      $BDDAT          ;INDICATE IT WAS CLEARED
2459 015072 104042                      ERROR    42              ;CONSOLE 1 L.P. CHANGE INTR ENABLE INSTR ER
2460
2461 ;*****
2462 ;*TEST 124      TEST THAT NOP INSTR CAN SET & RESET L.P. SWITCH INTR ENABLED-CONSOLE 0
2463 ;*****
2464 015074 000004      TST124: SCOPE
2465 015076 012777 020000 165012      MOV      #BIT13,2STKPT      ;SET MAINT SWITCH #2
2466 015104 012737 015126 001110      MOV      #1$,2LPERR        ;SET UP SCOPE LOOP ADRS
2467 015112 012737 002000 001124      MOV      #2000,$GDDAT      ;EXPECT LP SW INTR ENABLED BIT
2468 015120 012737 164014 027526      MOV      #164014,BUFFER    ;SET UP INSTR
2469 015126 012777 027526 164730      1$: MOV      #BUFFER,2DPC      ;START
2470 015134 013737 002106 001122      MOV      CONS,$BDADR      ;SET UP REG ADRS 22
2471 015142 017737 164740 001126      MOV      2CONS,$BDDAT      ;READ L.P. SWITCH INTR ENABLED REG
2472 015150 042737 175777 001126      BIC      #175777,$BDDAT    ;SAVE ONLY L.P. SWITCH INTR ENABLED BIT
2473 015156 023737 001124 001126      CMP      $GDDAT,$BDDAT    ;IS IT CORRECT?
2474 015164 001401                      BEQ      2$              ;BR IF OK
2475 015166 104043                      ERROR    43              ;CONSOLE 0 L.P. SW INTR ENABLE INSTR ER
2476 015170 162737 002000 001124      2$: SUB      #2000,$GDDAT    ;GO TO RESET STATE
2477 015176 100404                      BMI      TST125          ;ADVANCE TO NEXT TEST IF BOTH STATES TESTED
2478 015200 042737 000004 027526      BIC      #4,BUFFER        ;CLR THE ENABLED BIT AT INSTR LOC
2479 015206 000747                      BR        1$            ;GO TRY RESET STATE
2480
2481 ;*****
2482 ;*TEST 125      TEST THAT L.P. SWITCH INTR ENABLED DOES NOT RESET WHEN ENA=0
2483 ;*****
2484 015210 000004      TST125: SCOPE
2485 015212 012777 020000 164676      MOV      #BIT13,2STKPT      ;SET MAINT SWITCH #2
2486 015220 012737 002000 001124      MOV      #2000,$GDDAT      ;EXPECT L.P. SWITCH INTR ENABLED TO BE SET
2487 015226 012737 164014 027526      MOV      #164014,BUFFER    ;SET UP INSTR LOC
2488 015234 012777 027526 164622      MOV      #BUFFER,2DPC      ;START
2489 015242 012737 164000 027530      MOV      #164000,BUFFER+2  ;SET UP INSTR WITH ENA=0
2490 015250 005277 164610      INC      2DPC              ;ADVANCE TO NEXT INSTR
2491 015254 013737 002106 001122      MOV      CONS,$BDADR      ;SET UP REG ADRS 22
2492 015262 032777 002000 164616      BIT      #2000,2CONS        ;IS L.P. SWITCH INTR ENABLED STILL SET?
2493 015270 001003                      BNE      TST126          ;ADVANCE TO NEXT TEST IF STILL SET
2494 015272 005037 001126      CLR      $BDDAT          ;INDICATE IT WAS CLEARED
2495 015276 104043                      ERROR    43              ;CONSOLE 0 L.P. SW CHANGE INTR ENABLE INSTR ER
2496
2497 ;*****

```

```

2498 ;*TEST 126 TEST THAT NOP INSTR CAN SET & RESET L.P. SWITCH INTR ENABLED-CONSOLE 1
2499 ;*****
2500 †TST126: SCOPE
2501 MOV #BIT13,‡STKPT ;SET MAINT SWITCH #2
2502 MOV #15,‡LPERR ;SET UP SCOPE LOOP ADRS
2503 MOV #20,‡GDDAT ;EXPECT L.P. SWITCH INTR ENAELED BIT
2504 MOV #164414,BUFFER ;SET UP INSTR
2505 1‡: MOV #BUFFER,‡DPC ;START
2506 MOV ‡CONS,‡BDADR ;SET UP REG ADRS 22
2507 MOV ‡CONS,‡BDDAT ;READ L.P. SWITCH INTR ENABLED REG
2508 BIC #177757,‡BDDAT ;SAVE ONLY L.P. SWITCH INTR ENABLED BIT
2509 CMP ‡GDDAT,‡BDDAT ;IS IT CORRECT?
2510 BEQ 2‡ ;BR IF OK
2511 ERROR 43 ;CONSOLE 1 L.P. SW INTR ENABLE INSTR ER
2512 2‡: SUB #20,‡GDDAT ;GO TO RESET STATE
2513 BMI TST127 ;ADVANCE TO NEXT TEST IF BOTH STATES TESTED
2514 BIC #4,BUFFER ;CLR THE ENABLED BIT AT INSTR LOC
2515 BR 1‡ ;GO TRY RESET STATE
2516
2517 ;*****
2518 ;*TEST 127 TEST THAT L.P. SWITCH INTR ENABLED DOES NOT RESET WHEN ENA=0
2519 ;*****
2520 †TST127: SCOPE
2521 MOV #BIT13,‡STKPT ;SET MAINT SWITCH #2
2522 MOV #20,‡GDDAT ;EXPECT L.P. SW INTR ENABLED TO BE SET
2523 MOV #164414,BUFFER ;SET UP INSTR LOC
2524 MOV #BUFFER,‡DPC ;START
2525 MOV #164400,BUFFER+2 ;SET UP INSTR LOC WITH ENA=0
2526 INC ‡DPC ;ADVANCE TO NEXT INSTR
2527 MOV ‡CONS,‡BDADR ;SET UP REG ADRS 22
2528 BIT #20,‡CONS ;IS L.P. SW INTR ENABLED STILL SET?
2529 BNE TST130 ;ADVANCE TO NEXT TEST IF STILL SET
2530 CLR ‡BDDAT ;INDICATE IT WAS CLEARED
2531 ERROR 43 ;CONSOLE 1 L.P. SW CHANGE INTR ENABLE INSTR ER
2532
2533 ;*****
2534 ;*TEST 130 TEST THAT RESET WILL CLEAR L.P. INTR ENABLES & COLOR
2535 ;*****
2536 †TST130: SCOPE
2537 MOV #40,‡TIMES ;DO 40 ITERATIONS
2538 MOV ‡CONS,‡BDADR ;SET UP REG ADRS 22
2539 MOV #BIT13,‡STKPT ;SET MAINT SW 2
2540 MOV #164274,BUFFER ;SET UP INSTR FOR CONSOLE 0 L.P. INTR ENAB
2541 MOV #BUFFER,‡DPC ;START
2542 MOV #164774,BUFFER+2 ;CONS 1 L.P. INTR ENA
2543 INC ‡DPC ;RESUME
2544 MOV #175600,BUFFER+4 ;COLOR LEVEL
2545 INC ‡DPC ;RESUME
2546 MOV #BIT15,‡GDDAT ;EXPECT INTENSITY ENABLE CONSOLE 0 ONLY
2547 RESET ;CLR FLAGS
2548 MOV ‡CONS,‡BDADR ;READ FLAGS
2549 CMP ‡GDDAT,‡BDDAT ;IS REG 22 CORRECT?
2550 BEQ TST131 ;NEXT TEST IF SO
2551 ERROR 32 ;RESET FAILED TO SET UP REG 22
2552
2553

```

M04

```

2554
2555
2556
2557 015620 000004
2558 015622 012777 020000 164266
2559 015630 012777 160000 164176
2560 015636 012737 025252 001124
2561 015644 013777 002034 164212
2562 015652 013737 001124 027530
2563 015660 005277 164200
2564 015664 013737 002064 001122
2565 015672 017737 164166 001126
2566 015700 023737 001124 001126
2567 015706 001401
2568 015710 104010
2569
2570
2571
2572
2573
2574 015712 000004
2575 015714 012777 020000 164174
2576 015722 012777 160000 164104
2577 015730 012737 012524 001124
2578 015736 013777 002034 164120
2579 015744 013737 001124 027530
2580 015752 005277 164106
2581 015756 013737 002064 001122
2582 015764 017737 164074 001126
2583 015772 023737 001124 001126
2584 016000 001401
2585 016002 104010
2586
2587
2588
2589
2590
2591 016004 000004
2592 016006 012737 000020 001164
2593 016014 013737 025232 001124
2594 016022 012777 020000 164066
2595 016030 012737 160000 027526
2596 016036 012777 027526 164020
2597 016044 013737 001124 027530
2598 016052 005277 164006
2599 016056 013737 002064 001122
2600 016064 017737 163774 001126
2601 016072 023737 001124 001126
2602 016100 001401
2603 016102 104010
2604
2605 016104 162737 000002 001124
2606 016112 103351
2607
2608
2609

```

```

*****
*TEST 131 JUMP TEST TO LOC. 25252
*****
TST131: SCOPE
MOV #BIT13, @STKPT ;SET MAINT SWITCH #2
MOV #160000, @DBUF ;MOVE DJUMP INTO THE BUFFER
MOV #25252, $GDDAT ;MOVE 25252 INTO THE NEXT LOC.
MOV @DBUF, @DPC ;START THE DISPLAY
MOV $GDDAT, BUFFER+2 ;LD ADRS
INC @DPC ;SINGLE STEP THE DISPLAY
MOV DPC, $BDADR ;SET UP REG ADRS 00
MOV @DPC, $BDDAT ;READ THE DISPLAY P.C.
CMP $GDDAT, $BDDAT ;COMPARE EXPCT TO RCVD
BEQ TST132 ;;BR IF EQUAL
ERROR 10 ;DJUMP FAILED TO LOAD THE NEW
;DISPLAY P.C. PROPERLY

*****
*TEST 132 JUMP TEST TO LOC. 12524
*****
TST132: SCOPE
MOV #BIT13, @STKPT ;SET MAINT SWITCH #2
MOV #160000, @DBUF ;MOVE DJUMP INTO THE BUFFER
MOV #12524, $GDDAT ;MOVE 12524 INTO THE NEXT LOC.
MOV @DBUF, @DPC ;START THE DISPLAY
MOV $GDDAT, BUFFER+2 ;LD ADRS
INC @DPC ;SINGLE STEP THE DISPLAY
MOV DPC, $BDADR ;SET UP REG ADRS 00
MOV @DPC, $BDDAT ;READ THE DISPLAY P.C.
CMP $GDDAT, $BDDAT ;COMPARE EXPCT TO RCVD
BEQ TST133 ;;BR IF EQUAL
ERROR 10 ;DJUMP FAILED TO LOAD THE NEW
;DISPLAY P.C. PROPERLY

*****
*TEST 133 JUMP TO EACH EXISTING LOCATION IN LOWER BANK
*****
TST133: SCOPE
MOV #20, $TIMES ;;DO 20 ITERATIONS
MOV $LSTAD, $GDDAT ;LOAD JUMP DESTINATION
MOV #BIT13, @STKPT ;SET MAINT SWITCH #2
MOV #160000, @BUFFER ;LOAD "DISPLAY JUMP" INST
1$: MOV @BUFFER, @DPC ;START DISPLAY
MOV $GDDAT, BUFFER+2 ;SET UP ADRS
INC @DPC ;SINGLE STEP THE DISPLAY
MOV DPC, $BDADR ;SET UP REG ADRS 00
MOV @DPC, $BDDAT ;READ DESTINATION D.P.C.
CMP $GDDAT, $BDDAT ;TEST IF EXPECTED
BEQ 2$ ;;BR IF SAME
ERROR 10 ;JUMP TO LOCATION "N" FAILED
;N=ADDRESS IN $GDDAT
2$: SUB #2, $GDDAT ;ADJUST EXPECTED ADDRESS
BCC 1$ ;BR IF MORE LOCATIONS TO TEST

*****
*TEST 134 TEST 'JUMP RELATIVE' INSTR (BIT 8=0) AT LOC 'BUFFER' & VERIFY NEW DPC AD

```

```

2610
2611 016114 000004
2612 016116 012737 016140 001110
2613 016124 012777 020000 163764
2614 016132 012700 000400
2615 016136 000423
2616 016140 012737 161000 027526 1S:
2617 016146 050037 027526
2618 016152 012777 027526 163704
2619 016160 013737 002064 001122
2620 016166 017737 163672 001126
2621 016174 023737 001124 001126
2622 016202 001401
2623 016204 104011
2624 016206 006200 2S:
2625 016210 103410
2626 016212 010037 001124
2627 016216 006337 001124
2628 016222 062737 027530 001124
2629 016230 000743
2630
2631
2632 ;*****
;*TEST 135 TEST 'JUMP RELATIVE' INSTR (BIT 8=1) AT LOC 'BUFFER' & VERIFY NEW DPC
;*****
2633
2634 016232 000004
2635 016234 012737 016256 001110
2636 016242 012777 020000 163646
2637 016250 012700 000400
2638 016254 000423
2639 016256 012737 161400 027526 1S:
2640 016264 050037 027526
2641 016270 012777 027526 163566
2642 016276 013737 002064 001122
2643 016304 017737 163554 001126
2644 016312 023737 001124 001126
2645 016320 001401
2646 016322 104011
2647 016324 006200 2S:
2648 016326 103413
2649 016330 012737 027530 001124
2650 016336 010001
2651 016340 005401
2652 016342 006301
2653 016344 042701 177000
2654 016350 160137 001124
2655 016354 000740
2656
2657
2658 ;*****
;*TEST 136 TEST THAT JUMP ABS INSTR WILL SETUP DPC WITH A FLOATING ONE
;*****
2659
2660 016356 000004
2661 016360 012777 020000 163530
2662 016366 012737 160000 027526
2663 016374 005037 001124
2664 016400 012737 000001 027530
2665 016406 012777 027526 163450 1S:

```

B05

```

2666 016414 012737 016406 001110      MOV      #15,$LPERR      ;SET UP SCOPE LOOP ADRS
2667 016422 013737 002064 001122      MOV      DPC,$BDADR     ;SET UP REG ADRS 00
2668 016430 005277 163430          INC      @DPC           ;PICK UP ABS ADRS
2669 016434 000240          NOP                    ;ALLOW TIME FOR 2ND NPR
2670 016436 017737 163422 001126      MOV      @DPC,$BDDAT    ;GET ADRS JUMPED TO
2671 016444 023737 001124 001126      CMP      $GDDAT,$BDDAT  ;IS IT CORRECT?
2672 016452 001401          BEQ      25            ;BR IF OK
2673 016454 104010          ERROR   10           ;JUMP ABSOLUTE FAILED (FLOAT A 1 ADDRESS)
2674 016456 006337 027530          25:  ASL      BUFFER+2      ;ADVANCE ONE BIT LEFT
2675 016462 103404          BCS      TST137       ;ADVANCE TO NEXT TEST IF ALL BITS TESTED
2676 016464 013737 027530 001124      MOV      BUFFER+2,$GDDAT ;SET UP EXPECTED ADRS
2677 016472 000745          BR       15           ;TRY NEXT BIT

```

```

:*****
:*TEST 137 TEST THAT JUMP ABS INSTR WILL SETUP DPC WITH A FLOATING ZERO
:*****

```

```

2681 016474 000004          TST137: SCOPE
2682 016476 012737 016526 001110      MOV      #15,$LPERR     ;SET UP SCOPE LOOP ADRS
2683 016476 012737 016526 001110      MOV      #BIT13,@STKPT   ;SET MAINT SWITCH #2
2684 016504 012777 020000 163404      MOV      #160000,BUFFER  ;SET UP JUMP ABS INSTR AT 'BUFFER'
2685 016512 012737 160000 027526      MOV      #177774,$GDDAT  ;SET LSB TO ZERO
2686 016520 012737 177774 001124      MOV      #BUFFER,@DPC   ;START
2687 016526 012777 027526 163330      15:  MOV      $GDDAT,BUFFER+2 ;SET UP ADRS
2688 016534 013737 001124 027530      MOV      INC @DPC       ;PICK UP ABS ADRS
2689 016542 005277 163316          MOV      DPC,$BDADR     ;SET UP REG ADRS 00
2690 016546 013737 002064 001122      MOV      @DPC,$BDDAT    ;GET ADRS JUMPED TO
2691 016554 017737 163304 001126      CMP      $GDDAT,$BDDAT  ;IS IT CORRECT?
2692 016562 023737 001124 001126      BEQ      25            ;BR IF OK
2693 016570 001401          ERROR   10           ;JUMP ABSOLUTE FAILED (TO A FLOATING ZERO)
2694 016572 104010          25:  ASL      $GDDAT       ;SHIFT ZERO BIT LEFT
2695 016574 006337 001124          BCC      TST140       ;ADVANCE TO NEXT TEST IF ALL BITS TESTED
2696 016600 103004          ADD     #2,$GDDAT     ;REPLACE LSB
2697 016602 062737 000002 001124      BR       15           ;TRY NEXT BIT
2698 016610 000746

```

```

:*****
:*TEST 140 TEST THAT THE 'JSR ABSOLUTE' INSTR WILL SET 'JSR' BIT IN REG 32
:*****

```

```

2701 016612 000004          TST140: SCOPE
2702 016614 012777 020000 163274      MOV      #BIT13,@STKPT   ;SET MAINT SWITCH #2
2703 016622 012737 162000 027526      MOV      #162000,BUFFER  ;SET UP JSR ABS INSTR
2704 016630 005037 027530          CLR     BUFFER+2       ;CLR ABS ADRS LOC
2705 016634 012737 001000 001124      MOV      #BIT9,$GDDAT   ;EXPECT JSR BIT
2706 016642 012777 027526 163214      MOV      #BUFFER,@DPC   ;START
2707 016650 013737 002116 001122      MOV      STKPT,$BDADR   ;SET UP REG ADRS 32
2708 016656 032777 001000 163232      BIT     #BIT9,@STKPT    ;IS JSR BIT SET?
2709 016664 001003          BNE     TST141       ;ADVANCE TO NEXT TEST IF OK
2710 016666 005037 001126          CLR     $BDDAT        ;SHOW THAT IT WAS ZERO
2711 016672 104012          ERROR   12           ;CALL ERROR ROUTINE USING MSG 23

```

```

:*****
:*TEST 141 TEST 'JSR RELATIVE' INSTR (BIT 8=0) AT LOC 'BUFFER' & VERIFY NEW DPC ADR
:*****

```

```

2714 016674 000004          TST141: SCOPE
2715 016676 012777 020000 163212      MOV      #BIT13,@STKPT   ;SET MAINT SWITCH #2
2716 016704 012737 016720 001110      MOV      #15,$LPERR     ;SET UP SCOPE LOOP ADRS
2717 016712 012700 000400          MOV      #400,R0        ;SET UP MSB FIRST

```

C05

```

016716 000426          BH      25      ;GO CALCULATE ADRS
016720 052777 000040 163170 15:  BIS      #40,2STKPT ;RESET STK PTR
016726 012737 163000 027526      MOV      #163000,BUFFER ;SET UP JSR RELATIVE INSTR
016734 050037 027526      BIS      RO,BUFFER ;SET UP RELATIVE ADRS PATTERN
016740 012777 027526 163116      MOV      #BUFFER,2DPC ;START
016746 013737 002064 001122      MOV      DPC,$BDADR ;SET UP REG ADRS 00
016754 017737 163104 001126      MOV      2DPC,$BDDAT ;READ NEW DPC ADRS
016762 023737 001124 001126      CMP      $GDDAT,$BDDAT ;DOES IT EQUAL THE CALCULATED ADRS?
016770 001401          BEQ      25      ;BR IF OK
016772 104013          ERROR    13      ;JSR RELATIVE "POSITIVE" FAILED
016774 006200          ASR      RO      ;SHIFT RIGHT TO NEXT BIT
016776 103410          BCS     TST142 ;ADVANCE TO NEXT TEST IF DONE
017000 010037 001124      MOV      RO,$GDDAT ;SET UP OFFSET VALUE
017004 006337 001124      ASL     $GDDAT ;MAKE INTO WORD OFFSET
017010 062737 027530 001124      ADD     #BUFFER+2,$GDDAT ;ADD IN ADRS OF INSTR PLUS 2
017016 000740          BR      15      ;TRY THIS RELATIVE ADRS

```

```

*****
*TEST 142 TEST 'JSR RELATIVE' INSTR (BIT 8=1) AT LOC 'BUFFER' & VERIFY NEW DPC ADR
*****

```

```

TST142: SCOPE
017020 000004          MOV      #BIT13,2STKPT ;SET MAINT SWITCH #2
017022 012777 020000 163066      MOV      #15,$LPERR ;SET UP SCOPE LOOP ADRS
017030 012737 017044 001110      MOV      #400,RO ;SET UP MSB FIRST
017036 000400          BR      25      ;GO CALCULATE ADRS
017042 052777 000040 163044 15:  BIS      #40,2STKPT ;RESET STK PTR
017044 012737 163400 027526      MOV      #163400,BUFFER ;SET UP JSR RELATIVE INSTR
017052 050037 027526      BIS      RO,BUFFER ;SET UP RELATIVE ADRS PATTERN
017060 012777 027526 162772      MOV      #BUFFER,2DPC ;START
017072 013737 002064 001122      MOV      DPC,$BDADR ;SET UP REG ADRS 00
017100 017737 162760 001126      MOV      2DPC,$BDDAT ;READ NEW DPC ADRS
017106 023737 001124 001126      CMP      $GDDAT,$BDDAT ;DOES IT EQUAL THE CALCULATED VALUE?
017114 001401          BEQ      25      ;BR IF OK
017116 104013          ERROR    13      ;JSR RELATIVE NEGATIVE TO ADDRESS FAILED
017120 006200          ASR      RO      ;SHIFT RIGHT TO NEXT BIT
017122 103413          BCS     TST143 ;ADVANCE TO NEXT TEST IF DONE
017124 012737 027530 001124      MOV      #BUFFER+2,$GDDAT ;SET UP CURRENT ADRS
017132 010001      MOV      RO,R1 ;GET RELATIVE ADRS VALUE
017134 005401      NEG     R1 ;MAKE NEG
017136 006301      ASL     R1 ;CORRECT FOR WORD OFFSET
017140 042701 177000      BIC     #177000,R1 ;GET RID OF MSBS
017144 160137 001124      SUB     R1,$GDDAT ;SUB OFFSET FROM DPC VALUE
017150 000735          BR      15      ;TRY NEXT RELATIVE ADRS

```

```

*****
*TEST 143 TEST THAT 'JSR ABS' INSTR WILL SET UP DPC WITH A FLOATING ONE
*****

```

```

TST143: SCOPE
017152 000004          MOV      #BIT13,2STKPT ;SET MAINT SWITCH #2
017154 012777 020000 162734      MOV      #162000,BUFFER ;SET UP JSR ABSOLUTE INSTR AT 'BUFFER'
017162 012737 162000 027526      CLR     $GDDAT ;EXPECT ZEROS FIRST TIME
017170 005037 001124      MOV      #1,BUFFER+2 ;SET UP ABS ADRS TO 1
017174 012737 000001 027530 15:  BIS      #40,2STKPT ;RESET STK PTR
017202 052777 000040 162706      MOV      #BUFFER,2DPC ;START
017210 012777 027526 162646      MOV      #15,$LPERR ;SET UP SCOPE LOOP ADRS
017216 012737 017202 001110      INC     2DPC ;ADVANCE TO ABS ADRS
017224 005277 162634

```

```

017230 013737 002064 001122      MOV      DPC,$BDADR      ;SET UP REG ADRS 00
017236 017737 162622 001126      MOV      @DPC,$BDDAT    ;GET ADRS JUMPED TO
017244 023737 001124 001126      CMP      $GDDAT,$BDDAT  ;IS IT CORRECT?
017250 001401      BEQ      25             ;BR IF OK
017254 104012      ERROR    12            ;JSR ABSOLUTE FAILED (TO A FLOATING ONE)
017256 006337 027530 25:      ASL      BUFFER+2      ;ADVANCE ONE BIT LEFT
017262 103404      SCS      TST144        ;ADVANCE TO NEXT TEST IF ALL BITS TESTED
017264 013737 027530 001124      MOV      BUFFER+2,$GDDAT ;SET UP EXPECTED ADRS
017272 000743      BR       15            ;TRY NEXT BIT

```

```

*****
;*TEST 144 TEST THAT JSR ABS INSTR WILL SET UP DPC WITH A FLOATING ZERO
*****

```

```

TST144: SCOPE
017274 000004      MOV      @BIT13,@STKPT  ;SET MAINT SWITCH #2
017276 012777 020000 162612      MOV      @15,$LPERR    ;SET UP SCOPE LOOP ADRS
017304 012737 017326 001110      MOV      @162000,BUFFER ;SET UP JSR ABSOLUTE AT 'BUFFER'
017312 012737 162000 027526      MOV      @177774,$GDDAT ;SET LSB TO ZERO
017320 012737 177774 001124      MOV      @40,@STKPT    ;RESET STK PTR
017326 052777 000040 162562 15:      BIS      @BUFFER,@DPC  ;START
017334 012777 027526 162522      MOV      $GDDAT,BUFFER+2 ;SET UP ADRS
017342 013737 001124 027530      INC      @DPC          ;ADVANCE TO ABS ADRS
017350 005277 162510      MOV      DPC,$BDADR    ;SET UP REG ADRS 03
017354 013737 002064 001122      MOV      @DPC,$BDDAT   ;GET ADRS JUMPED TO
017362 017737 162476 001126      CMP      $GDDAT,$BDDAT ;IS IT CORRECT?
017370 023737 001124 001126      BEQ      25             ;BR IF OK
017376 001401      ERROR    12            ;JSR ABSOLUTE FAILED (TO A FLOATING ZERO)
017400 104012      ASL      $GDDAT        ;SHIFT ZERO BIT LEFT
017402 006337 001124 25:      BCC      TST145        ;ADVANCE TO NEXT TEST IF ALL BITS TESTED
017406 103004      ADD     @2,$GDDAT      ;REPLACE LSB
017410 062737 000002 001124      BR       15            ;TRY NEXT BIT
017416 000743

```

```

*****
;*TEST 145 TEST THAT GRAPHPLT X CAN LOAD X POSITION REGISTER WITH #1252
*****

```

```

TST145: SCOPE
017420 000004      MOV      XPOS,$BDADR   ;SET UP REG ADRS 04
017422 013737 002070 001122      MOV      @BIT13,@STKPT ;SET MAINT SWITCH #2
017430 012777 020000 162460      MOV      @BUFFER,R0    ;LOAD BUFFER POINTER
017436 012700 027526      MOV      @122000,(R0)+ ;LOW INTERSITY - SET GRAPH PLOT X MODE
017442 012720 122000      MOV      @1252,(R0)+  ;SET X POSITION
017446 012720 001252      MOV      @DBUF,@DPC    ;LOAD DISPLAY P.C.
017452 013777 002034 162404      MOV      @1252,$GDDAT ;LOAD EXPECTED
017460 012737 001252 001124      INC      @DPC          ;SINGLE STEP THE DISPLAY
017466 005277 162372      JSR      PC,DLAY       ;STALL
017472 004737 023170      MOV      @XPOS,$BDDAT  ;READ X POSITION
017476 017737 162366 001126      CMP      $GDDAT,$BDDAT ;COMPARE EXPCT TO RCVD
017504 023737 001124 001126      BEQ      TST146        ;BR IF EQUAL
017512 001401      ERROR    14            ;X POSITION REGISTER FAILED TO LOAD
017514 104014      ;PROPERLY USING GRAPH PLOT X MODE

```

```

*****
;*TEST 146 TEST THAT GRAPHPLT X CAN LOAD X POSITION REGISTER WITH #525
*****

```

```

TST146: SCOPE
017516 000004      MOV      XPOS,$BDADR   ;SET UP REG ADRS 04
017520 013737 002070 001122

```

E05

```

2834 017526 012777 020000 162362      MOV      #BIT13,STKPT      ;SET MAINT SWITCH #2
2835 017534 012700 027526          MOV      #BUFFER,RO      ;LOAD BUFFER POINTER
2836 017540 012720 122000          MOV      #122000,(RO)+   ;LOW INTENSITY - SET GRAPH PLOT X MODE
2837 017544 012720 000525          MOV      #525,(RO)+     ;SET X POSITION
2838 017550 012777 027526 162306      MOV      #BUFFER,JDPC    ;STAR DISPLAY
2839 017556 012737 000525 001124      MOV      #525,$GDDAT    ;LOAD EXPECTED
2840 017564 005277 162274          INC      JDPC           ;SINGLE STEP THE DISPLAY
2841 017570 004737 023170          JSR      PC,DLAY        ;STALL
2842 017574 017737 162270 001126      MOV      #YPOS,$BDDAT   ;READ X POSITION
2843 017602 023737 001124 001126      CMP      $GDDAT,$BDDAT  ;COMPARE EXPCT TO RCVD
2844 017610 001401          BEQ      TST147         ;;BR IF EQUAL
2845 017612 104014          ERROR    14           ;X POSITION REGISTER FAILED TO LOAD
                          ;PROPERLY USING GRAPH PLOT X MODE

;*****
;*TEST 147      TEST THAT GRAPH PLOT Y CAN LOAD Y POSITION REGISTER WITH #1252
;*****
TST147: SCOPE
2851 017614 000004          MOV      YPOS,$BDDADR   ;SET UP REG ADRS 06
2852 017616 013737 002072 001122      MOV      #BIT13,STKPT   ;SET MAINT SWITCH #2
2853 017624 012777 020000 162264      MOV      #BUFFER,RO     ;LOAD BUFFER POINTER
2854 017632 012700 027526          MOV      #126000,(RO)+  ;LOW INTENSITY - SET GRAPH PLOT Y
2855 017636 012720 126000          MOV      #1252,(RO)+   ;SET Y POSITION
2856 017642 012720 001252          MOV      #DBUF,JDPC    ;LOAD THE DISPLAY P.C.
2857 017646 013777 002034 162210      MOV      #1252,$GDDAT  ;LOAD EXPECTED
2858 017654 012737 001252 001124      INC      JDPC          ;SINGLE STEP THE DISPLAY
2859 017662 005277 162176          JSR      PC,DLAY        ;STALL
2860 017666 004737 023170          MOV      #YPOS,$BDDAT  ;READ Y POSITION
2861 017672 017737 162174 001126      CMP      $GDDAT,$BDDAT ;COMPARE EXPCT TO RCVD
2862 017700 023737 001124 001126      BEQ      TST150         ;;BR IF EQUAL
2863 017706 001401          ERROR    15           ;Y POSITION REGISTER FAILED TO LOAD
                          ;PROPERLY USING GRAPH PLOT Y MODE

;*****
;*TEST 150      TEST THAT GRAPH PLOT Y CAN LOAD Y POSITION REGISTER WITH #525
;*****
TST150: SCOPE
2870 017712 000004          MOV      YPOS,$BDDADR   ;SET UP REG ADRS 06
2871 017714 013737 002072 001122      MOV      #BIT13,STKPT   ;SET MAINT SWITCH #2
2872 017722 012777 020000 162166      MOV      #BUFFER,RO     ;LOAD BUFFER POINTER
2873 017730 012700 027526          MOV      #126000,(RO)+  ;LOW INTENSITY - SET GRAPH PLOT Y MODE
2874 017734 012720 126000          MOV      #525,(RO)+   ;SET Y POSITION
2875 017740 012720 000525          MOV      #DBUF,JDPC    ;LOAD THE DISPLAY P.C.
2876 017744 013777 002034 162112      MOV      #525,$GDDAT  ;LD EXPECTED
2877 017752 012737 000525 001124      INC      JDPC          ;SINGLE STEP THE DISPLAY
2878 017760 005277 162100          JSR      PC,DLAY        ;STALL
2879 017764 004737 023170          MOV      #YPOS,$BDDAT  ;READ Y POSITION
2880 017770 017737 162076 001126      CMP      $GDDAT,$BDDAT ;COMPARE EXPCT TO RCVD
2881 017776 023737 001124 001126      BEQ      TST151         ;;BR IF EQUAL
2882 020004 001401          ERROR    15           ;Y POSITION REGISTER FAILED TO LOAD
                          ;PROPERLY USING GRAPH PLOT Y MODE

;*****
;*TEST 151      TEST FOR PROPER SELECTION OF X AND Y REGISTERS
;*****

```

F05

```

2890 020010 000004          TST151: SCOPE
2891 020012 013737 002070 001122  MOV   XPOS,$B0ADR ;SET UP REG ADRS 04
2892 020020 012777 020000 162070  MOV   #BIT13,2STKPT ;SET MAINT SWITCH #2
2893 020026 012700 027526          MOV   #BUFFER,RO ;LOAD BUFFER POINTER
2894 020032 012720 122000          MOV   #122000,(RO)+ ;LOW INTENSITY - SET GRAPHPLOT X MODE
2895 020036 012720 001234          MOV   #1234,(RO)+ ;SET X POSITION
2896 020042 012720 126000          MOV   #126000,(RO)+ ;SET GRAPHPLOT Y MODE
2897 020046 012720 001432          MOV   #1432,(RO)+ ;SET Y POSITION
2898 020052 013777 002034 162004  MOV   DBUF,2DPC ;LOAD THE DISPLAY P.C.
2899 020060 012737 001234 001124  MOV   #1234,$GDDAT ;LD EXPECTED
2900 020066 005277 161772          INC   2DPC ;SINGLE STEP THE DISPLAY
2901 020072 004737 023170          JSR   PC,DLAY ;STALL
2902 020076 017737 161766 001126  MOV   2XPOS,$BDDAT ;READ X POSITION
2903 020104 023737 001124 001126  CMP   $GDDAT,$BDDAT ;COMPARE EXPCT TO RCVD
2904 020112 001402          BEQ   15 ;GRAPHPLOT X MODE FAILED TO SELECT
2905 020114 104014          ERROR 14 ;
2906 020116 000424          BR    TST152 ;
2907
2908 020120          15:
2909 020120 005277 161740          INC   2DPC ;SINGLE STEP THE DISPLAY
2910 020124 013737 002072 001122  MOV   YPOS,$B0ADR ;SET UP REG ADRS 06
2911 020132 005277 161726          INC   2DPC ;SINGLE STEP THE DISPLAY
2912 020136 012737 001432 001124  MOV   #1432,$GDDAT ;LOAD EXPECTED
2913 020144 004737 023170          JSR   PC,DLAY ;STALL
2914 020150 017737 161716 001126  MOV   2YPOS,$BDDAT ;READ Y POSITION
2915 020156 023737 001124 001126  CMP   $GDDAT,$BDDAT ;COMPARE EXPCT TO RCVD
2916 020164 001401          BEQ   TST152 ;BR IF EQUAL
2917 020166 104015          ERROR 15 ;Y POSITION REGISTER FAILED TO LOAD
2918 ;PROPERLY USING GRAPHPLOT Y MODE
2919
2920 ;*****
2921 ;*TEST 152 TEST THAT RESET CLEARS X AND Y POSITION REG.
2922 ;*****
2923 020170 000004          TST152: SCOPE
2924 020172 012737 000040 001164  MOV   #40,STIMES ;DO 40 ITERATIONS
2925 020200 013737 002070 001122  MOV   XPOS,$B0ADR ;SET UP REG ADRS 04
2926 020206 012737 114000 027526  MOV   #114000,BUFFER ;LOAD "POINT" INST.
2927 020214 012777 020000 161674  MOV   #BIT13,2STKPT ;SET MAINT SWITCH #2
2928 020222 012777 027526 161634  MOV   #BUFFER,2DPC ;START DISPLAY
2929 020230 012737 001777 027530  MOV   #1777,BUFFER+2 ;X=1777
2930 020236 005277 161622          INC   2DPC ;SINGLE STEP THE DISPLAY
2931 020242 012737 001777 027532  MOV   #1777,BUFFER+4 ;Y=1777
2932 020250 005277 161610          INC   2DPC ;SINGLE STEP THE DISPLAY
2933 020254 004737 023170          JSR   PC,DLAY ;STALL
2934
2935 020260 000005          RESET ;INITILIZE
2936 020262 005037 001124          CLR   $GDDAT ;CLEAR EXPECTED
2937 020266 017737 161576 001126  MOV   2XPOS,$BDDAT ;READ X POSITION
2938 020274 001401          BEQ   15 ;BR IF CLEARED
2939 020276 104032          ERROR 32 ;RESET FAILED TO CLEAR X POS. REGISTER
2940
2941 020300 013737 002072 001122  15:  MOV   YPOS,$B0ADR ;SET UP REG ADRS 06
2942 020306 017737 161560 001126  MOV   2YPOS,$BDDAT ;READ Y POSITION
2943 020314 001401          BEQ   TST153 ;BR IF CLARED
2944 020316 104032          ERROR 32 ;RESET FAILED TO CLEAR Y POS. REGISTER
2945 ;*****

```

G05

```
2946 :*TEST 153 TEST THAT DISPLAY BUSY WILL SET & THEN CLR ON INTERNAL STOP
2947 :*****
2948 020320 000004          †TST153: SCOPE
2949 020322 012777 020000 161566      MOV      #BIT13, †STKPT ;SET MAINT SW 2
2950 020330 012737 100000 001124      MOV      #100000, †SGDDAT ;EXPECT DISPLAY BUSY INITIALLY
2951 020336 012737 164000 027526      MOV      #164000, †BUFFER ;SET UP A NOP INSTR
2952 020344 012737 172000 027530      MOV      #172000, †BUFFER+2 ;SET UP LD STATUS A INSTR WITH STOP
2953 020352 012777 027526 161504      MOV      #BUFFER, †DPC ;START
2954 020360 013737 002076 001122      MOV      †SREG1, †SBDADR ;SET UP REG ADRS 12
2955 020366 005777 161504          TST      †SREG1 ;SEE THAT DISPLAY BUSY IS SET
2956 020372 100403          BMI      15 ;BR IF 50
2957 020374 005037 001126      CLR      †SBDAT ;INDICATE DISPLAY BUSY NOT SET
2958 020400 104044          ERROR   44 ;DISPLAY BUSY NOT SET
2959 020402 005277 161456      15: INC     †DPC ;ADVANCE TO STOP INSTR
2960 020406 005037 001124      CLR      †SGDDAT ;EXPECTE ZERO
2961 020412 005777 161460          TST      †SREG1 ;SEE THAT DISPLAY BUSY HAS CLEARED
2962 020416 100004          BPL     TST154 ;GO TO NEXT TEST IF OK
2963 020420 012737 100000 001126      MOV      #100000, †SBDAT ;INDICATE DISPLAY BUSY WAS STILL SET
2964 020426 104044          ERROR   44 ;DISPLAY BUSY FAILED TO CLR ON STOP INSTR
2965
2966
2967 :*****
2968 :*TEST 154 TEST THAT RESET WILL CLEAR DISPLAY BUSY
2969 :*****
2970 020430 000004          †TST154: SCOPE
2971 020432 012737 000040 001164      MOV      #40, †TIMES ;DO 40 ITERATIONS
2972 020440 013737 002076 001122      MOV      †SREG1, †SBDADR ;SET UP REG ADRS 12
2973 020446 012777 020000 161442      MOV      #BIT13, †STKPT ;SET MAINT SW 2
2974 020454 012737 164000 027526      MOV      #164000, †BUFFER ;SET UP NOP INSTR
2975 020462 012777 027526 161374      MOV      #BUFFER, †DPC ;START
2976 020470 012737 000404 001124      MOV      #404, †SGDDAT ;EXPECT ONLY CHAR & VECTOR SCALE BITS
2977 020476 000005          RESET   ;CLR BUSY
2978 020500 017737 161372 001126      MOV      †SREG1, †SBDAT ;READ REG 12
2979 020506 023737 001124 001126      CMP      †SGDDAT, †SBDAT ;IS IT CORRECT?
2980 020514 001401          BEQ     TST155 ;NEXT TEST IF OK
2981 020516 104044          ERROR   44 ;RESET FAILED TO CLR DISPLAY BUSY
2982
2983 :*****
2984 :*TEST 155 TEST THAT EXTERNAL STOP WILL SET & RESET WITH DISPLAY OFF
2985 :*****
2986 020520 000004          †TST155: SCOPE
2987 020522 012737 000040 001164      MOV      #40, †TIMES ;DO 40 ITERATIONS
2988 020530 012777 020000 161360      MOV      #BIT13, †STKPT ;SET MAINT SWITCH #2
2989 020536 012737 000340 177776      MOV      #340, †PSW ;SET PRIORITY TO HIGEST LEVEL
2990 020544 000005          RESET   ;MAKE SURE DISPLAY IS OFF
2991 020546 013737 002076 001122      MOV      †SREG1, †SBDADR ;SET UP REG 12 ADRS
2992 020554 012737 000200 001124      MOV      #200, †SGDDAT ;EXPECT EXTERNAL STOP TO BE SET
2993 020562 013777 001124 161306      MOV      †SGDDAT, †SREG1 ;SET EXTERNAL STOP
2994 020570 105777 161302          TSTB   †SREG1 ;SEE IF BIT SET
2995 020574 100403          RMC    15 ;BR IF IT DID SET
2996 020576 005037 001126      CLR      †SBDAT ;INDICATE IT WAS NOT SET
2997 020602 104045          ERROR   45 ;EXTERNAL STOP FAILED TO SET
2998 020604 012777 027526 161252      15: MOV     #BUFFER, †DPC ;CLR EXTERNAL STOP BIT
2999 020612 105777 161260          TSTB   †SREG1 ;CHECK THAT IT WAS CLEARED
3000 020616 100006          BPL     TST156 ;BR TO NEXT TEST IF OK
3001 020620 005037 001124      CLR      †SGDDAT ;EXPECTED ZERO
```

H05

```
3002 020624 012737 000200 001126      MOV      #200,$BDDAT      ;INDICATE THAT IT DID NOT CLR
3003 020632 104045                      ERROR     45              ;EXTERNAL STOP FAILED TO CLR ON START
3004
3005
3006
3007
3008 020634 000004                      ;*****
3009 020636 012777 020000 161252      MOV      #BIT13,2STKPT    ;SET MAINT SWITCH #2
3010 020644 012737 000340 177776      MOV      #340,PSW        ;SET PSW TO 7
3011 020652 012737 000200 001124      MOV      #200,$GDDAT     ;EXPECT EXTERNAL STOP TO BE SET INITIALLY
3012 020660 012737 164000 027526      MOV      #164000,BUFFER  ;SET UP A NOP INSTR
3013 020666 012777 027526 161170      MOV      #BUFFER,2DPC    ;START
3014 020674 012737 164000 027530      MOV      #164000,BUFFER+2 ;SET UP ANOTHER NOP
3015 020702 013777 001124 161166      MOV      $GDDAT,2SREG1   ;ISSUE A SOFTWARE STOP
3016 020710 005277 161150                      INC      2DPC            ;THIS RESUME SHOULD ALLOW EXTERNAL STOP TO SET
3017 020714 013737 002076 001122      MOV      SREG1,$BDADR    ;SET UP REG ADRS 12
3018 020722 032777 000200 161146      BIT      #200,2SREG1     ;SEE IF IT SET
3019 020730 001004                      BNE      15              ;BR IF 50
3020 020732 005037 001126      CLR      $BDDAT         ;INDICATE IT DID NOT SET
3021 020736 104045                      ERROR     45              ;EXTERNAL STOP FAILED TO SET WHEN DISPLAY BUSY
3022 020740 000415                      BR       TST157          ;GO LOOK FOR LOOP ON TEST SW
3023 020742 012777 027526 161114 15:    MOV      #BUFFER,2DPC    ;START - THIS SHOULD CLR EXT STOP
3024 020750 032777 000200 161120      BIT      #200,2SREG1     ;CHECK THAT IT DID CLEAR
3025 020756 001406                      BEQ      TST157          ;GO TO NEXT TEST IF EXTERNAL STOP CLEARED
3026 020760 005037 001124                      CLR      $GDDAT         ;EXPECTED ZERO
3027 020764 012737 000200 001126      MOV      #200,$BDDAT     ;INDICATE IT DID NOT CLR
3028 020772 104045                      ERROR     45              ;EXTERNAL STOP FAILED TO CLR ON START
3029
3030
3031
3032
3033 020774 000004                      ;*****
3034 020776 012737 000010 001164      MOV      #10,$TIMES     ;;DO 10 ITERATIONS
3035 021004 000005                      RESET                      ;MAKE SURE DISPLAY IS OFF
3036 021006 013737 002076 001122      MOV      SREG1,$BDADR    ;SET UP REG 12 ADRS
3037 021014 005037 001124                      CLR      $GDDAT         ;EXPECT RESET STATE
3038 021020 012777 000200 161050      MOV      #200,2SREG1     ;SET EXTERNAL STOP
3039 021026 000005                      RESET                      ;CLR IT
3040 021030 032777 000200 161040      BIT      #200,2SREG1     ;DID IT CLR?
3041 021036 001404                      BEQ      TST160          ;NEXT TEST IF 50
3042 021040 012737 000200 001126      MOV      #200,$BDDAT     ;SHOW THAT IT WAS STILL SET
3043 021046 104045                      ERROR     45              ;RESET FAILED TO CLR EXTERNAL STOP
3044
3045
3046
3047
3048 021050 000004                      ;*****
3049 021052 012737 000040 001164      MOV      #40,$TIMES     ;;DO 40 ITERATIONS
3050 021060 012777 020000 161030      MOV      #BIT13,2STKPT  ;SET MAINT SWITCH #2
3051 021066 000005                      RESET                      ;MAKE SURE DISPLAY IS OFF
3052 021070 012737 000340 177776      MOV      #340,PSW        ;SET PSW TO 7
3053 021076 013737 002076 001122      MOV      SREG1,$BDADR    ;SET UP REG 12 ADRS
3054 021104 012737 000200 001124      MOV      #200,$GDDAT     ;EXPECT EXTERNAL STOP FLAG
3055 021112 013701 001250                      MOV      $VECT1,R1      ;GET BASIC VECTOR ADRS
3056 021116 012711 021162                      MOV      #15,(R1)       ;SET UP STOP INT ADRS
3057 021122 012761 000340 000002      MOV      #340,2(R1)     ;SET UP PSW TO 7 ON INT
```

```

3058 021130 012777 000200 160740      MOV      #200,DSREG1      ;CAUSE AN EXTERNAL STOP INT
3059 021136 005037 177776      CLR      PSW              ;ALLOW INT TO OCCUR
3060 021142 017737 160730 001126      MOV      DSREG1,$BDDAT   ;READ REG 12
3061 021150 042737 177577 001126      BIC      #177577,$BDDAT  ;SAVE ONLY STOP FLAG
3062 021156 104033          ERROR    33              ;EXTERNAL STOP FAILED TO INTERRUPT
3063 021160 000410          BR       2$              ;GO LOOK FOR LOOP ON TEST SWITCH
3064 021162 022626          1$:    CMP      (R6)+,(R6)+ ;FIX STK SINCE NO RETURN
3065 021164 032777 000200 160704      BIT      #200,DSREG1    ;CK THAT STOP FLAG IS SET
3066 021172 001003          BNE     2$              ;BR IF SO
3067 021174 005037 001126      CLR      $BDDAT         ;INDICATE FLAG WAS NOT SET
3068 021200 104033          ERROR    33              ;EXTERNAL STOP INTERRUPTED BUT FLAG NOT SET
3069 021202 004737 023134      2$:    JSR      PC,RSTVEC   ;RESTORE EXTERNAL STOP VECTOR WITH HALT
3070
3071
3072
3073

```

```

;*****
;*TEST 161      TEST THAT TIME OUT WILL CAUSE AN INTERRUPT
;*****

```

```

3074 021206 000004          †ST161: SCOPE
3075 021210 012737 000040 001164      MOV      #40,$TIMES     ;;DO 40 ITERATIONS
3076 021216 012777 020000 160672      MOV      #BIT13,STKPT   ;SET MAINT SWITCH #2
3077 021224 012737 000340 177776      MOV      #340,PSW       ;SET PRIORITY TO HIGHEST LEVEL
3078 021232 013701 001250          MOV      $VECT1,R1      ;GET BASIC VECTOR ADRS
3079 021236 012761 021330 000010      MOV      #1$,10(R1)     ;SET UP TIMEOUT INT SERVICE ADRS
3080 021244 012761 000340 000012      MOV      #340,12(R1)    ;SET UP PRIORITY TO 7 ON INT
3081 021252 013737 002076 001122      MOV      SREG1,$SDADR   ;SET UP REG 12 ADRS
3082 021260 012737 004060 001124      MOV      #4060,$GDDAT   ;EXPECT TIMEOUT & DPC 16,17
3083 021266 012777 007600 160600      MOV      #7600,$RLO     ;RELOCATE TO ADRS 760000
3084 021274 005037 177776          CLR      PSW            ;ALLOW INTERRUPT TO OCCUR
3085 021300 012777 000000 160556      MOV      #0,$DPC        ;DISPLAY WILL TIMEOUT AT ADRS 760000
3086 021306 021616          CMP      (SP),(SP)      ;ALLOW TIME
3087 021310 017737 160562 001126      MOV      DSREG1,$BDDAT  ;READ REG 12
3088 021316 042737 173717 001126      BIC      #173717,$BDDAT ;SAVE ONLY TIMEOUT BIT & DPC 16,17
3089 021324 104046          ERROR    46              ;VS60 FAILED TO INTERRUPT ON TIME OUT
3090 021326 000414          BR       2$              ;GO LOOK FOR LOOP ON TEST SWITCH
3091 021330 022626          1$:    CMP      (R6)+,(R6)+ ;FIX STACK SINCE NO RETURN
3092 021332 017737 160540 001126      MOV      DSREG1,$BDDAT  ;READ STATUS
3093 021340 042737 173717 001126      BIC      #173717,$BDDAT ;SAVE TIMEOUT & DPC 16,17
3094 021346 023737 001124 001126      CMP      $GDDAT,$BDDAT ;CORRECT?
3095 021354 001401          BEQ     2$              ;BR IF SO
3096 021356 104046          ERROR    46              ;TIMEOUT BIT FAILED TO SET ON INTERRUPT
3097 021360 004737 023134      2$:    JSR      PC,RSTVEC   ;RESET TIMEOUT VECTOR WITH HALT
3098 021364 000005          RESET
3099
3100
3101
3102

```

```

;*****
;*TEST 162      TEST THAT RESET WILL CLEAR TIMEOUT FLAG
;*****

```

```

3103 021366 000004          †ST162: SCOPE
3104 021370 012737 000040 001164      MOV      #40,$TIMES     ;;DO 40 ITERATIONS
3105 021376 013737 002076 001122      MOV      SREG1,$SDADR   ;SET L.P. REG ADRS 12
3106 021404 012777 020000 160504      MOV      #BIT13,STKPT   ;SET MAINT SW 2
3107 021412 012737 000340 177776      MOV      #340,PSW       ;DON'T LET INTR OCCUR.
3108 021420 012737 000404 001124      MOV      #404,$GDDAT    ;EXPECT ONLY CHAR & VECTOR SCALE BITS
3109 021426 012777 007600 160440      MOV      #7600,$RLO     ;RELOCATE TO ADRS 760000
3110 021434 012777 000000 160422      MOV      #0,$DPC        ;DISPLAY SHOULD TIMEOUT AT ADRS 760000
3111 021442 021616          CMP      (SP),(SP)      ;ALLOW TIME
3112 021444 000005          RESET
3113 021446 017737 160424 001126      MOV      DSREG1,$BDDAT  ;READ REG 12

```

```

3114 021454 023737 001124 001126
3115 021462 001401
3116 021464 104032
3117
3118
3119
3120
3121 021466 000004
3122 021470 012737 000040 001164
3123 021476 012737 000340 177776
3124 021504 012777 020000 160404
3125 021512 012737 144000 001124
3126 021520 012737 164360 027526
3127 021526 012777 027526 160330
3128 021534 052777 040000 160354
3129 021542 012737 122000 027530
3130 021550 005277 160310
3131 021554 012737 001777 027532
3132 021562 005277 160276
3133 021566 013737 002106 001122
3134 021574 052777 040000 160304
3135 021602 017737 160300 001126
3136 021610 023737 001124 001126
3137 021616 001401
3138 021620 104034
3139 021622 000005
3140 021624 012737 100000 001124
3141 021632 017737 160250 001126
3142 021640 023737 001124 001126
3143 021646 001401
3144 021650 104032
3145
3146
3147
3148
3149 021652 000004
3150 021654 012737 000040 001164
3151 021662 012737 000340 177776
3152 021670 012777 020000 160220
3153 021676 012737 101440 001124
3154 021704 012737 164760 027526
3155 021712 012777 027526 160144
3156 021720 052777 040000 160170
3157 021726 012737 122000 027530
3158 021734 005277 160124
3159 021740 012737 001777 027532
3160 021746 005277 160112
3161 021752 013737 002106 001122
3162 021760 052777 000400 160120
3163 021766 017737 160114 001126
3164 021774 023737 001124 001126
3165 022002 001401
3166 022004 104034
3167 022006 000005
3168 022010 012737 100000 001124
3169 022016 017737 160064 001126

```

```

CMP $GDDAT,$BDDAT ;IS IT CORRECT?
BEQ TST163 ;NEXT TEST IF OK
ERROR 32 ;RESET FAILED TO CLR TIMEOUT

:*****
;*TEST 163 TEST THAT LIGHT PEN FLAG WILL SET THEN CLEAR ON RESET - CONS 0
:*****
TST163: SCOPE
MOV #40,$TIMES ;DO 40 ITERATIONS
MOV #340,PSW ;HIGHEST PRIORITY
MOV #20000,STKPT ;SET MAINT SW 2
MOV #144000,$GDDAT ;EXPECT L.P. FLAG INITIALLY
MOV #164360,BUFFER ;ENABLE L.P. FLG INTR
MOV #BUFFER,DPCC ;START
BIS #40000,STKPT ;SET MAINT SW 3
MOV #122000,BUFFER+2 ;GRAPH X MODE
INC DPCC ;RESUME
MOV #1777,BUFFER+4 ;X=1777
INC DPCC ;RESUME
MOV CONS,$BDDADR ;SET UP REG ADRS 22
BIS #40000,CONS ;SET L.P. FLAG
MOV CONS,$BDDAT ;READ FLAGS REG 22
CMP $GDDAT,$BDDAT ;IS IT CORRECT?
BEQ 15 ;BR IF SO
ERROR 34 ;LIGHT PEN FLAG FAILED TO SET - CONSOLE 0
RESET ;CLR FLAGS
MOV #100000,$GDDAT ;EXPECT ONLY INT ENA CONSOLE 0
MOV CONS,$BDDAT ;READ REG 22
CMP $GDDAT,$BDDAT ;DID REG 22 RESET?
BEQ TST164 ;NEXT TEST IF SO
ERROR 32 ;RESET FAILED TO CLEAR FLAGS - CONSOLE 0

:*****
;*TEST 164 TEST THAT LIGHT PEN FLAG WILL SET THEN CLEAR ON RESET - CONS 1
:*****
TST164: SCOPE
MOV #40,$TIMES ;DO 40 ITERATIONS
MOV #340,PSW ;HIGHEST PRIORITY
MOV #20000,STKPT ;SET MAINT SW 2
MOV #101440,$GDDAT ;EXPECT L.P. FLAG INITIALLY
MOV #164760,BUFFER ;ENABLE L.P. FLG
MOV #BUFFER,DPCC ;START
BIS #40000,STKPT ;SET MAINT SW 3
MOV #122000,BUFFER+2 ;GRAPH X MODE
INC DPCC ;RESUME
MOV #1777,BUFFER+4 ;X=1777
INC DPCC ;RESUME
MOV CONS,$BDDADR ;SET UP REG ADRS 22
BIS #400,CONS ;SET L.P. FLAG
MOV CONS,$BDDAT ;READ IT BACK
CMP $GDDAT,$BDDAT ;CORRECT?
BEQ 15 ;BR IF SO
ERROR 34 ;LIGHT PEN FLAG FAILED TO SET - CONSOLE 1
RESET ;CLR FLAGS
MOV #100000,$GDDAT ;EXPECT ONLY INT EN FOR CONSOLE 0
MOV CONS,$BDDAT ;READ REG 22

```

K05

```

3170 022024 023737 001124 001126      CMP      $GDDAT,$BDDAT      ;CORRECT?
3171 022032 001401                      BEQ      TST165             ;NEXT TEST IF 50
3172 022034 104032                      ERROR    32                 ;RESET FAILED TO CLR FLAGS - CONSOLE 1
3173
3174 ;*****
3175 ;*TEST 165      TEST THAT LIGHT PEN FLAG - CONSOLE 0 WILL CAUSE AN INTERRUPT
3176 ;*****
3177 †TST165: SCOPE
3178 022036 000004
3179 022040 012737 000100 001164      MOV      #100,$TIMES        ;DO 100 ITERATIONS
3180 022046 012737 000340 177776      MOV      #340,PSW          ;SET PRIORITY TO 7
3181 022054 013701 001250                      MOV      $VECT1,R1         ;GET BASIC VS60 VECTOR ADRS
3182 022060 012761 022230 000004      MOV      #35,4(R1)         ;SET UP INTR SERVICE ADRS
3183 022066 012761 000340 000006      MOV      #340,6(R1)        ;SET UP PRIORITY TO 7 ON INT
3184 022074 012737 144000 001124      MOV      #144000,$GDDAT    ;EXPECT INT EN, L.P. FLG, L.P. INT EN
3185 022102 012700 000001                      MOV      #1,R0             ;R0 WHEN NON-ZERO MEANS INT EXPECTED
3186 022106 012777 020040 160002      MOV      #20040,‡STKPT     ;SET MAINT SW 2
3187 022114 012737 164360 027526      MOV      #164360,BUFFER    ;SET UP TO ENAB L.P. INT & INTENSITY EN
3188 022122 012777 027526 157734      MOV      #BUFFER,‡DPC      ;START
3189 022130 052777 040000 157760      BIS      #40000,‡STKPT     ;SET MAINT SW 3
3190 022136 012737 122000 027530      MOV      #122000,BUFFER+2  ;GRAPH X MODE
3191 022144 005277 157714                      INC      ‡DPC              ;PICK UP GRAPH X INSTR - INT WILL OCCUR IN DATA MODE
3192 022150 012737 001777 027532      MOV      #1777,BUFFER+4    ;X=1777
3193 022156 005277 157702                      INC      ‡DPC              ;RESUME FOR X
3194 022162 013737 002106 001122      MOV      CONS,$BDADR       ;SET UP REG ADRS 22
3195 022170 052777 040000 157710      BIS      #40000,‡CONS     ;SET L.P. FLAG
3196 022176 005037 177776                      CLR      PSW               ;ALLOW INTR
3197 022202 012701 000040                      MOV      #40,R1           ;SET UP DELAY
3198 022206 005301 25: DEC      R1                ;COUNT AWAY
3199 022210 001376      BNE      25:              ;STAY UNTIL DONE
3200 022212 005700      TST      R0              ;DO WE EXPECT INT THIS PASS - NO IF ZERO
3201 022214 001451      BEQ      75:              ;BR IF NOT
3202 022216 017737 157664 001126      MOV      ‡CONS,$BDDAT     ;READ L.P. STATUS
3203 022224 104050      ERROR    50              ;LIGHT PEN CONSOLE 0 INTERRUPT FAILURE
3204 022226 000454      BR       85:              ;GO LOOK FOR LOOP ON TEST SW
3205 022230 022626 35: CMP      (R6)+,(R6)+       ;FIX STACK SINCE NO RTI
3206 022232 017737 157650 001126      MOV      ‡CONS,$BDDAT     ;GET CONTENTS OF REG 22
3207 022240 005700      TST      R0              ;DID WE EXPECT AN L.P. INTERRUPT THIS PASS?
3208 022242 001002      BNE      45:              ;BR IF 50
3209 022244 104050      ERROR    50              ;L.P. FLAG INTERRUPTED WHEN NOT ENABLED
3210 022246 000444      BR       85:              ;GO LOOK FOR LOOP ON TEST SW
3211 022250 023737 001124 001126 45: CMP      $GDDAT,$BDDAT    ;SEE IF STATUS IS CORRECT AFTER INT
3212 022256 001402      BEQ      55:              ;BR IF 50
3213 022260 104050      ERROR    50              ;INCORRECT INTERRUPT STATUS
3214 022262 000436      BR       85:              ;GO LOOK FOR LOOP ON TEST SW
3215 022264 032777 000200 157574 55: BIT      #200,‡SREGO       ;L.P. FLG SET IN REG 02 ALSO?
3216 022272 001012      BNE      65:              ;BR IF 50
3217 022274 013737 002066 001122      MOV      SREGO,$BDADR     ;SET UP REG ADRS 02
3218 022302 012737 000200 001124      MOV      #200,$GDDAT      ;EXPECTED L.P. FLAG
3219 022310 005037 001126      CLR      $BDDAT          ;SHOW IT WAS NOT SET
3220 022314 104050      ERROR    50              ;L.P. FLAG FAILED TO SET AT REG 02
3221 022316 000420      BR       85:              ;GO LOOK FOR LOOP ON TEST SW
3222 022320 012737 100000 001124 65: MOV      #100000,$GDDAT    ;SET UP EXPECTED WITH L.P. FLAG INT DISABLED
3223 022326 012737 164340 027526      MOV      #164340,BUFFER   ;SET UP INSTR WITH L.P. FLAG INT DISABLED
3224 022334 005000      CLR      R0              ;SET TEST SWITCH TO NO INTERRUPT EXPECTED
3225 022336 000671      BR       15:              ;GO REPEAT TEST WITH INT EY DISABLED
3225 022340 017737 157542 001126 75: MOV      ‡CONS,$BDDAT     ;READ REG 22

```

L05

```

3226 022346 023737 001126 001124      CMP      $BDDAT,$GDDAT      ;FLAG CLR ON START?
3227 022354 001401                      BEQ      8$                ;BR IF SO
3228 022356 104034                      ERROR   34                ;START FAILED TO CLR L.P. FLAG
3229 022360 004737 023134      8$:     JSR      PC,RSTVEC      ;RESET L.P. INTERRUPT VECTOR WITH HALT
3230
3231
3232 ;*****
3233 ;*TEST 166 TEST THAT LIGHT PEN FLAG - CONSOLE 1 WILL CAUSE AN INTERRUPT
3234 ;*****
3234 022364 000004      †ST166: SCOPE
3235 022366 012737 000100 001164      MOV      #100,$TIMES      ;DO 100 ITERATIONS
3236 022374 012737 000340 177776      MOV      #340,PSW        ;SET PRIORITY TO 7
3237 022402 013701 001250                      MOV      $VECT1,R1        ;GET BASIC VS60 VECTOR ADRS
3238 022406 012761 022556 000004      MOV      #3$,4(R1)        ;SET UP INTR SERVICE ADRS
3239 022414 012761 000340 000006      MOV      #340,6(R1)       ;SET UP PRIORITY TO 7 ON INT
3240 022422 012737 101440 001124      MOV      #101440,$GDDAT   ;EXPECT INT EN, L.P. FLG, L.P. INT EN
3241 022430 012700 000001                      MOV      #1,RO            ;RO WHEN NON-ZERO MEANS INT EXPECTED
3242 022434 012777 020040 157454      MOV      #20040,2STKPT    ;SET MAINT SW 2
3243 022442 012737 164760 027526      MOV      #164760,BUFFER   ;SET UP TO ENAB L.P. INT & INTENSITY EN
3244 022450 012777 027526 157406      1$:     MOV      #BUFFER,2DPC    ;START
3245 022456 052777 040000 157432      BIS      #40000,2STKPT    ;SET MAINT SW 3
3246 022464 012737 122000 027530      MOV      #122000,BUFFER+2 ;GRAPH X MODE
3247 022472 005277 157366                      INC      2DPC             ;PICK UP GRAPH X INSTR - INT WILL OCCUR IN DATA MODE
3248 022476 012737 001777 027532      MOV      #1777,BUFFER+4   ;X=1777
3249 022504 005277 157354                      INC      2DPC             ;RESUME FOR X
3250 022510 013737 002106 001122      MOV      CONS,$BDADR      ;SET UP REG ADRS 22
3251 022516 052777 000400 157362      BIS      #400,2CONS      ;SET L.P. FLAG
3252 022524 005037 177776                      CLR      PSW              ;ALLOW INTR
3253 022530 012701 000040                      MOV      #40,R1          ;SET UP DELAY
3254 022534 005301 2$:     DEC      R1               ;COUNT AWAY
3255 022536 001376                      BNE     2$               ;STAY UNTIL DONE
3256 022540 005700                      TST     RO               ;DO WE EXPECT INT THIS PASS - NO IF ZERO
3257 022542 001433                      BEQ     6$               ;BR IF NOT
3258 022544 017737 157336 001126      MOV      2CONS,$BDDAT     ;READ L.P. STATUS
3259 022552 104050                      ERROR   50               ;LIGHT PEN CONSOLE 1 INTERRUPT FAILURE
3260 022554 000436                      BR      7$               ;GO LOOK FOR LOOP ON TEST SW
3261 022556 022626 3$:     CMP      (R6)+,(R6)+      ;FIX STACK SINCE NO RTI
3262 022560 017737 157322 001126      MOV      2CONS,$BDDAT     ;GET L.P. STATUS
3263 022566 005700                      TST     RO               ;DID WE EXPECT AN L.P. INTERRUPT THIS PASS?
3264 022570 001002                      BNE     4$               ;BR IF SO
3265 022572 104050                      ERROR   50               ;L.P. FLAG INTERRUPTED WHEN NOT ENABLED
3266 022574 000426                      BR      7$               ;GO LOOK FOR LOOP ON TEST SW
3267 022576 023737 001124 001126      4$:     CMP      $GDDAT,$BDDAT ;SEE IF STATUS IS CORRECT AFTER INT
3268 022604 001402                      BEQ     5$               ;BR IF SO
3269 022606 104050                      ERROR   50               ;INCORRECT L.P. STATUS
3270 022610 000420                      BR      7$               ;GO LOOK FOR LOOP ON TEST SW
3271 022612 012737 101000 001124      5$:     MOV      #101000,$GDDAT ;SET UP TEST WITH L.P. FLAG INT DISABLED
3272 022620 012737 164740 027526      MOV      #164740,BUFFER   ;SET UP INSTR WITH L.P. FLAG DISABLED
3273 022626 005000                      CLR     RO               ;SET TEST SWITCH TO NO INTERRUPT EXPECTED
3274 022630 000707                      BR      1$               ;GO REPEAT TEST WITH INT EN DISABLED
3275 022632 017737 157250 001126      6$:     MOV      2CONS,$BDDAT     ;READ REG 22
3276 022640 023737 001124 001126      CMP      $GDDAT,$BDDAT    ;L.P. FLG CLR ON START?
3277 022646 001401                      BEQ     7$               ;BR IF SO
3278 022650 104034                      ERROR   34                ;START FAILED TO CLR L.P. FLAG
3279 022652 004737 023134      7$:     JSR      PC,RSTVEC      ;RESET L.P. INTERRUPT VECTOR WITH HALT
3280
3281 ;*****

```

M05

```

3282 ;*TEST 16? TEST THAT WORD 0,1 & 2 INDICATORS WILL SET
3283 ;*****
3284 022656 000004 TST16?: SCOPE
3285 022660 012777 020000 157230 MOV #BIT13,2STKPT ;SET MAINT SWITCH #2
3286 022666 012737 000100 001124 MOV #100,$GDDAT ;EXPECT WORD 0 INDICATION INITIALLY
3287 022674 012737 11C300 027526 MOV #110000,BUFFER ;SET UP LONG VECTOR INSTR
3288 022702 005037 027530 CLR BUFFER+2 ;CLR X,Y
3289 022706 005037 027532 CLR BUFFER+4
3290 022712 012777 027526 157144 MOV #BUFFER,2DPC ;START
3291 022720 000402 BR 2$ ;GO ALLOW TIME FOR INSTR NPR
3292 022722 005277 157136 1$: INC 2DPC ;PICK UP DATA WORD
3293 022726 013737 002116 001122 2$: MOV STKPT,$BDADR ;SET UP REG ADRS 32
3294 022734 017737 157156 001126 MOV 2STKPT,$BDDAT ;READ REG 32
3295 022742 042737 177077 001126 BIC #177077,$BDDAT ;SAVE ONLY WORD INDICATORS
3296 022750 023737 001124 001126 CMP $GDDAT,$BDDAT ;IS IT CORRECT?
3297 022756 001401 BEQ 3$ ;BR IF SO
3298 022760 104041 ERROR 41 ;WORD INDICATOR FAILURE
3299 022762 006337 001124 3$: ASL $GDDAT ;ADVANCE TO NEXT WORD
3300 022766 022737 001000 001124 CMP #1000,$GDDAT ;HAVE ALL BEEN TESTED?
3301 022774 001352 BNE 1$ ;BR IF NOT
3302
3303 ;*****
3304 ;*TEST 170 TEST FOR HALT AT END OF DIAGNOSTIC (SW12 SET)
3305 ;*****
3306 022776 000004 TST170: SCOPE
3307 023000 032777 010000 156130 BIT #BIT12,2SWR ;IS SW12 SET?
3308 023006 001401 BEQ $EOP ;BR IF NOT
3309 023010 000000 HALT ;COMPLETED PASS - SW12 SAYS HALT

```

```

3310 ;*****
3311
3312 .SBTTL END OF PASS ROUTINE
3313
3314 ;*INCREMENT THE PASS NUMBER ($PASS)
3315 ;*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
3316 ;*TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
3317 ;*IF THERES A MONITOR GO TO IT
3318 ;*IF THERE ISN'T JUMP TO RESTRT
3319
3320 $EOP:
3321 023012 000004 SCOPE
3322 023014 005037 001102 CLR $TSTNM ;; ZERO THE TEST NUMBER
3323 023020 005037 001164 CLR $TIMES ;; ZERO THE NUMBER OF ITERATIONS
3324 023024 005237 001206 INC $PASS ;; INCREMENT THE PASS NUMBER
3325 023030 042737 100000 001206 BIC #100000,$PASS ;; DON'T ALLOW A NEG. NUMBER
3326 023036 005327 DEC (PC)+ ;; LOOP?
3327 023040 000001 $EOPCT: .WORD 1
3328 023042 003022 BGT $DOAGN ;; YES
3329 023044 012737 MOV (PC)+,2(PC)+ ;; RESTORE COUNTER
3330 023046 000001 $ENDCT: .WORD 1
3331 023050 023040 $EOPCT
3332 023052 104400 023114 TYPE $ENDMG ;; TYPE "END PASS #"
3333 023056 013746 001206 MOV $PASS,-(SP) ;; SAVE $PASS FOR TYPEOUT
3334 023062 104404 TYPDS ;; GO TYPE--DECIMAL ASCII WITH SIGN
3335 023064 104400 023131 TYPE , $ENULL ;; TYPE A NULL CHARACTER
3336 023070
3337 $GET42:
3338 023070 013700 000042 MOV @#42,R0 ;; GET MONITOR ADDRESS
3339 023074 001405 BEQ $DOAGN ;; BRANCH IF NO MONITOR
3340 023076 000005 RESET ;; CLEAR THE WORLD
3341 023100 004710 $ENDAD: JSR PC,(R0) ;; GO TO MONITOR
3342 023102 000240 NOP ;; SAVE ROOM
3343 023104 000240 NOP ;; FOR
3344 023106 000240 NOP ;; ACT11
3345 023110
3346 023110 000137 002506 $DOAGN: JMP @#RESTRT ;; RETURN
3347 023114 005015 047105 020104 $ENDMG: .ASCIZ <15><12>/END PASS #/
3348 023122 040520 051523 021440
3349 023130 000
3350 023131 377 377 000 $ENULL: .BYTE -1,-1,0 ;; NULL CHARACTER STRING
3351

```

000000
000001
000002
000003
000004
000005
000006
000007
000008
000009
000010
000011
000012
000013
000014
000015
000016
000017
000018
000019
000020
000021
000022
000023
000024
000025
000026
000027
000028
000029
000030
000031
000032
000033
000034
000035
000036
000037
000038
000039
000040
000041
000042
000043
000044
000045
000046
000047
000048
000049
000050
000051
000052
000053
000054
000055
000056
000057
000058
000059
000060
000061
000062
000063
000064
000065
000066
000067
000068
000069
000070
000071
000072
000073
000074
000075
000076
000077
000078
000079
000080
000081
000082
000083
000084
000085
000086
000087
000088
000089
000090
000091
000092
000093
000094
000095
000096
000097
000098
000099
000100

0000134 012737 000340 177776
0000142 012701 000010
0000146 013700 001250
0000150 010010
0000154 062720 000002
0000160 005020
0000162 005300
0000164 001372
0000166 000207

```
*****  
: THIS CODE RESETS ALL VS60 VECTORS WITH HALTS  
*****  
RSTVEC: MOV #340,PSW :RESET PRIORITY TO HIGHEST LEVEL  
MOV #10,R1 :SET UP VECTOR LOCATION COUNT  
MOV $VECT1,R0 :GET 1ST VS60 VECTOR ADRS  
15: MOV R0,(R0) :SET UP ADRS TO HALT  
ADD #2,(R0)+ :POINT IT TO HALT  
CLR (R0)+ :SET UP HALT  
DEC R1 :COUNT THIS VECTOR RESTORE  
BNE 15 :BR IF MORE TO RESTORE  
RTS PC :RETURN IF ALL DONE
```

0000170 013700 002062
0000174 005300
0000176 001376
0000200 000207

```
*****  
: THIS IS A GENERAL PURPOSE DELAY ROUTINE (REG CNT OF 20)  
*****  
DLAY: MOV DFLAY,R0 :GET COUNT  
15: DEC R0 :COUNT AWAY  
BNE 15 :BR TILL DONE  
RTS PC :EXIT
```

023202
023202
023210
023212
023214
023220
023226
023232
023236
023240
023242
023246
023250
023250
023256
023260
023266
023270
023274
023276
023304
023306
023314
023316
023324
023326
023332
023336
023340
023346
023350
023354
023356
023362
023370
023372
023400
023406
023412

032777 040000 155726
001114
000416
013746 000004
012737 023240 000004
005737 177060
012637 000004
000463
022626
012637 000004
000423
032777 000400 155660
001404
127737 155652 001102
001465
105737 001103
001421
123737 001115 001103
101015
032777 001000 155622
001404
013737 001110 001106
000446
105037 001103
005037 001164
000415
032777 004000 155570
001011
005737 001206
001406
005237 001104
023737 001164 001104
002024
012737 000001 001104
013737 023456 001164
105237 001102
113737 001102 001204

```

*****
.SBTTL SCOPE HANDLER ROUTINE

*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY'7:0')
*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY'15:08'
*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
*SW14=1 LOOP ON TEST
*SW11=1 INHIBIT ITERATIONS
*SW09=1 LOOP ON ERROR
*SW08=1 LOOP ON TEST IN SWR'7:0'
*CALL
* SCOPE ::SCOPE=IOT

$SCOPE:
15: BIT #BIT14,$SWR ::LOOP ON PRESENT TEST?
BNE $OVER ::YES IF SW14=1
*****START OF CODE FOR THE XOR TESTER*****
$TSTR: BR 65 ::IF RUNNING ON THE "XOR" TESTER CHANGE
::THIS INSTRUCTION TO A "NOP" (NOP=240)
MOV $ERRVEC, -(SP) ::SAVE THE CONTENTS OF THE ERROR VECTOR
MOV $SS,$ERRVEC ::SET FOR TIMEOUT
TST $177060 ::TIME OUT ON XOR?
MOV (SP)+,$ERRVEC ::RESTORE THE ERROR VECTOR
BR $SVLAD ::GO TO THE NEXT TEST
55: *P (SP)+,(SP)+ ::CLEAR THE STACK AFTER A TIME OUT
MOV (SP)+,$ERRVEC ::RESTORE THE ERROR VECTOR
BR 75 ::LOOP ON THE PRESENT TEST
65: *****END OF CODE FOR THE XOR TESTER*****
BIT #BIT08,$SWR ::LOOP ON SPEC. TEST?
BEQ 75 ::BR IF NO
CMPB $SWR,$TSTNM ::ON THE RIGHT TEST? SWR'7:0'
BEQ $OVER ::BR IF YES
25: TSTB $ERFLG ::HAS AN ERROR OCCURRED?
BEQ 35 ::BR IF NO
CMPB $ERMAX,$ERFLG ::MAX. ERRORS FOR THIS TEST OCCURRED?
BHI 35 ::BR IF NO
BIT #BIT09,$SWR ::LOOP ON ERROR?
BEQ 45 ::BR IF NO
75: MOV $LPERR,$LPAOR ::SET LOOP ADDRESS TO LAST SCOPE
BR $OVER
45: CLRB $ERFLG ::ZERO THE ERROR FLAG
CLR $TIMES ::CLEAR THE NUMBER OF ITERATIONS TO MAKE
BR 15 ::ESCAPE TO THE NEXT TEST
35: BIT #BIT11,$SWR ::INHIBIT ITERATIONS?
BNE 15 ::BR IF YES
TST $PASS ::IF FIRST PASS OF PROGRAM
BEQ 15 ::INHIBIT ITERATIONS
INC $ICNT ::INCREMENT ITERATION COUNT
CMP $TIMES,$ICNT ::CHECK THE NUMBER OF ITERATIONS MADE
BGE $OVER ::BR IF MORE ITERATION REQUIRED
15: MOV #1,$ICNT ::REINITIALIZE THE ITERATION COUNTER
MOV $MXCNT,$TIMES ::SET NUMBER OF ITERATIONS TO DO
$SVLAD: INCB $TSTNM ::COUNT TEST NUMBERS
MOVB $TSTNM,$TESTN ::SET TEST NUMBER IN APT MAILBOX

```

```

023429 023430 023431 023432 023433 023434 023435 023436 023437 023438 023439 023440 023441 023442 023443 023444 023445 023446 023447 023448 023449 023450 023451 023452 023453 023454 023455 023456
023420 023424 023430 023434 023442 023450 023454 023456
011637 011637 005037 112737 013777 013716 000002 003720
001106 001110 001166 000001 001102 001106
001115 155470
SOVER:
SMXCNT: 2000.
*****

.SBTTL APT COMMUNICATIONS ROUTINE
SAT1: MOV B1,SFFLG ;TO REPORT FATAL ERROR
SATY3: MOV B1,SMFLG ;TO TYPE A MESSAGE
SATY4: BR SATYC
SATYC: MOV B1,SFFLG ;TO ONLY REPORT FATAL ERROR

MOV R0,-(SP) ;PUSH R0 ON STACK
MOV R1,-(SP) ;PUSH R1 ON STACK
TSTB SMFLG ;SHOULD TYPE A MESSAGE?
BEQ 55 ;IF NOT: BR
CMPB BAPTENV,SENV ;OPERATING UNDER APT?
BNE 35 ;IF NOT: BR
BITB BAPTPOOL,SENV ;SHOULD SPOOL MESSAGES?
BEQ 35 ;IF NOT: BR
MOV R4(SP),R0 ;GET MESSAGE ADDR.
ADD #2,4(SP) ;BUMP RETURN ADDR.
15: TST $MSGTYPE ;SEE IF DONE W/ LAST XMISSION?
BNE 15 ;IF NOT: WAIT
MOV R0,$MSGAD ;PUT ADDR IN MAILBOX
25: TSTB (R0)+ ;FIND END OF MESSAGE
BNE 25
SUB $MSGAD,R0 ;SUB START OF MESSAGE
ASR R0 ;GET MESSAGE LGTH IN WORDS
MOV R0,$MSGLGT ;PUT LENGTH IN MAILBOX
MOV #4,$MSGTYPE ;TELL APT TO TAKE MSG.
BR 55
35: MOV R4(SP),R4 ;PUT MSG ADDR IN JSR LINKAGE
ADD #2,4(SP) ;BUMP RETURN ADDRESS
MOV 177776,-(SP) ;PUSH 177776 ON STACK
JSR PC,$TYPE ;CALL TYPE MACRO
45: .WORD 0
55:
105: TSTB $FFLG ;SHOULD REPORT FATAL ERROR?
BEQ 125 ;IF NOT: BR
TST $SENV ;RUNNING UNDER APT?
BEQ 125 ;IF NOT: BR
115: TST $MSGTYPE ;FINISHED LAST MESSAGE?
BNE 115 ;IF NOT: WAIT
MOV R4(SP),$FATAL ;GET ERROR #
ADD #2,4(SP) ;BUMP RETURN ADDR.
INC $MSGTYPE ;TELL APT TO TAKE ERROR
125: CLRB $FFLG ;CLEAR FATAL FLAG
CLRB $LFLG ;CLEAR LOG FLAG
CLRB $MFLG ;CLEAR MESSAGE FLAG
MOV (SP)+,R1 ;POP STACK INTO R1
MOV (SP)+,R0 ;POP STACK INTO R0

```

```

3485 023720 000207
3486
3487 023722 000
3488 023723 000
3489 023724 000
3490 023726
3491 000200
3492 000001
3493 000100
3494 000040
3495
3496
3497
3498
3499
3500
3501
3502
3503
3504
3505
3506
3507
3508
3509
3510 023726
3511 023726 113737 001102 002060
3512 023734 105237 001103
3513 023740 001775
3514 023742 013777 001102 155170
3515 023750 032777 002000 155160
3516 023756 001402
3517 023760 104400 001170
3518 023764 005237 001112
3519 023770 011637 001116
3520 023774 162737 000002 001116
3521 024002 117737 155110 001114
3522 024010 032777 020000 155120
3523 024016 001004
3524 024020 004737 024130
3525 024024 104400 001175
3526 024030
3527 024030 122737 000001 001220
3528 024036 001007
3529 024040 113737 001114 024052
3530 024046 004737 023476
3531 024052 000
3532 024053 000
3533 024054 000777
3534 024056 005777 155054
3535 024062 100006
3536 024064 000000
3537 024066 022737 023100 000042
3538 024074 001001
3539 024076 000000
3540 024100 032777 001000 155030

```

```

R15 PC ;RETURN
$MFLG: .BYTE 0 ; MESSG. FLAG
$LFLG: .BYTE 0 ; LOG FLAG
$FFLG: .BYTE 0 ; FATAL FLAG
.EVEN
APTSIZE=200
APTENV=001
APTSPool=100
APTCSNP=040
;*****
.SBTL ERROR HANDLER ROUTINE
; *THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT.
; *SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
; *AND GO TO $ERRTYP ON ERROR
; *THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
; *$SW15=1 HALT ON ERROR
; *$SW13=1 INHIBIT ERROR TYPEOUTS
; *$SW10=1 BELL ON ERROR
; *$SW09=1 LOOP ON ERROR
; *CALL
; * ERROR N ;;ERROR=EMT AND N=ERROR ITEM NUMBER

$ERROR:
7$: MOVB $TSTNM,$TSTNUM
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 $SBELL ;; RING BELL
1$: 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,$ERRTYP ;; GO TO USER ERROR ROUTINE
TYPE $SCLF

20$: CMPB #APTENV,$ENV ;; RUNNING IN APT MODE
BNE 2$ ;; NO SKIP APT ERROR REPORT
MOVB $ITEMB,$15 ;; SET ITEM NUMBER AS ERROR NUMBER
JSR PC,$SATY4 ;; 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!
CMP #SENDAD,$42 ;; ACT-11 AUTO-ACCEPT?
BNE 3$ ;; BRANCH IF NO
HALT ;; YES
3$: BIT #BIT09,$SWR ;; LOOP ON ERROR SWITCH SET?

```

3541	024106	001402		BEG	4\$::BR IF NO
3542	024110	013716	001110	MOV	\$1 PERR.(SP)	::FUDGE RETURN FOR LOOPING
3543	024114	005737	001166	4\$: TST	\$ESCAPE	::CHECK FOR AN ESCAPE ADDRESS
3544	024120	001402		BEG	5\$::BR IF NONE
3545	024122	013716	001166	5\$: MOV	\$ESCAPE.(SP)	::FUDGE RETURN ADDRESS FOR ESCAPE
3546	024126					
3547	024126	000002		RTI		::RETURN

```

3548
3549
3550
3551
3552
3553
3554
3555
3556
3557 024130
3558 024130 104400 001175
3559 024134 010046
3560 024136 005000
3561 024140 152700 001114
3562 024144 001004
3563
3564 024146 013746 001116
3565
3566 024152 104401
3567 024154 000426
3568 024156 005300
3569 024160 006300
3570 024162 006300
3571 024164 006300
3572 024166 062700 001324
3573 024172 012037 024202
3574 024176 001404
3575 024200 104400
3576 024202 000000
3577 024204 104400 001175
3578 024210 012037 024220
3579 024214 001404
3580 024216 104400
3581 024220 000000
3582 024222 104400 001175
3583 024226 011000
3584 024230 001004
3585 024232 012600
3586 024234 104400 001175
3587 024240 000207
3588 024242
3589 024242 013046
3590 024244 104401
3591 024246 005710
3592 024250 001770
3593 024252 104400 024260
3594 024256 000771
3595 024260 020040 000
3596 024264

```

.SBTTL ERROR MESSAGE TYPEOUT ROUTINE

```

;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.

```

```

$ERRTYP:
      TYPE      $CRLF      ;; "CARRIAGE RETURN" & "LINE FEED"
      MOV      RO,-(SP)    ;; SAVE RO
      CLR      RO          ;; PICKUP THE ITEM INDEX
      BISB     2*$ITEMB,RO
      BNE      1$         ;; IF ITEM NUMBER IS ZERO, JUST
                          ;; TYPE THE PC OF THE ERROR
      MOV      $ERRPC,-(SP) ;; SAVE $ERRPC FOR TYPEOUT
                          ;; ERROR ADDRESS
                          ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
                          ;; GET OUT
      BR       6$         ;; ADJUST THE INDEX SO THAT IT WILL
                          ;; WORK FOR THE ERROR TABLE
1$:   DEC      RO
      ASL     RO
      ASL     RO
      ASL     RO
      ADD     2*$ERRTB,RO  ;; FORM TABLE POINTER
      MOV     (RO)+,2$    ;; PICKUP "ERROR MESSAGE" POINTER
      BEQ     3$         ;; SKIP TYPEOUT IF NO POINTER
      TYPE   0           ;; TYPE THE "ERROR MESSAGE"
                          ;; "ERROR MESSAGE" POINTER GOES HERE
2$:   .WORD   0          ;; "CARRIAGE RETURN" & "LINE FEED"
      TYPE   $CRLF      ;; PICKUP "DATA HEADER" POINTER
      MOV     (RO)+,4$   ;; SKIP TYPEOUT IF 0
      BEQ     5$         ;; TYPE THE "DATA HEADER"
                          ;; "DATA HEADER" POINTER GOES HERE
4$:   .WORD   0          ;; "CARRIAGE RETURN" & "LINE FEED"
      TYPE   $CRLF      ;; PICKUP "DATA TABLE" POINTER
      MOV     (RO),RO    ;; GO TYPE THE DATA
      BNE     7$         ;; RESTORE RO
5$:   MOV     (SP)+,RO   ;; "CARRIAGE RETURN" & "LINE FEED"
      TYPE   $CRLF      ;; RETURN
      RTS     PC
7$:   MOV     2(RO)+,-(SP) ;; SAVE 2(RO)+ FOR TYPEOUT
      TYPEOC ;; GO TYPE--OCTAL ASCII ALL DIGITS)
      TST    (RO)       ;; IS THERE ANOTHER NUMBER?
      BEQ    6$         ;; BR IF NO
      TYPE   8$         ;; TYPE TWO(2) SPACES
      BR     7$         ;; LOOP
8$:   .ASCIZ  / /      ;; TWO(2) SPACES
      .EVEN

```

```

3597
3598
3599
3600
3601
3602
3603
3604
3605
3606
3607
3608
3609
3610
3611
3612
3613
3614
3615
3616
3617
3618
3619
3620
3621
3622
3623
3624 024264 017646 000000
3625 024270 116637 000001 024507
3626 024276 112637 024511
3627 024302 062716 000002
3628 024306 000406
3629 024310 112737 000001 024507
3630 024316 112737 000006 024511
3631 024324 112737 000005 024506
3632 024332 010346
3633 024334 010446
3634 024336 010546
3635 024340 113704 024511
3636 024344 005404
3637 024346 062704 000006
3638 024352 110437 024510
3639 024356 113704 024507
3640 024362 016605 000012
3641 024366 005003
3642 024370 006105 1$:
3643 024372 000404 2$:
3644 024374 006105
3645 024376 006105
3646 024400 006105
3647 024402 010503
3648 024404 006103 3$:
3649 024406 105337 024510
3650 024412 100016
3651 024414 042703 177770
3652 024420 001002

```

```

*****
.SBTTL BINARY TO OCTAL (ASCII) AND TYPE
*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
*OCTAL (ASCII) NUMBER AND TYPE IT.
*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
*CALL:
*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
*      TYPOS    ;;CALL FOR TYPEOUT
*      .BYTE   N              ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
*      .BYTE   M              ;;M=1 OR 0
*                               ;;1=TYPE LEADING ZEROS
*                               ;;0=SUPPRESS LEADING ZEROS
*$TYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
*$TYPOS OR $TYPOC
*CALL:
*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
*      TYPON    ;;CALL FOR TYPEOUT
*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
*CALL:
*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
*      TYPOC    ;;CALL FOR TYPEOUT
*$TYPOS: MOV      2(SP),-(SP)    ;;PICKUP THE MODE
MOV      1(SP),SOFILL          ;;LOAD ZERO FILL SWITCH
MOV      (SP)+,SOMODE+1        ;;NUMBER OF DIGITS TO TYPE
ADD      #2,(SP)               ;;ADJUST RETURN ADDRESS
BR       $TYPON
*$TYPOC: MOV      #1,SOFILL      ;;SET THE ZERO FILL SWITCH
MOV      #6,SOMODE+1           ;;SET FOR SIX(6) DIGITS
*$TYPON: MOV      #5,SOCNT       ;;SET THE ITERATION COUNT
MOV      R3,-(SP)              ;;SAVE R3
MOV      R4,-(SP)              ;;SAVE R4
MOV      R5,-(SP)              ;;SAVE R5
MOV      SOMODE+1,R4           ;;GET THE NUMBER OF DIGITS TO TYPE
NEG      R4
ADD      #6,R4                 ;;SUBTRACT IT FOR MAX. ALLOWED
MOV      R4,SOMODE             ;;SAVE IT FOR USE
MOV      SOFILL,R4            ;;GET THE ZERO FILL SWITCH
MOV      12(SP),R5             ;;PICKUP THE INPUT NUMBER
CLR      R3                    ;;CLEAR THE OUTPUT WORD
1$:     ROL      R5              ;;ROTATE MSB INTO "C"
BR       3$                    ;;GO DO MSB
2$:     ROL      R5              ;;FORM THIS DIGIT
ROL      R5
ROL      R5
ROL      R5
3$:     MOV      R5,R3
ROL      R3                    ;;GET LSB OF THIS DIGIT
DECB    SOMODE                 ;;TYPE THIS DIGIT?
BPL     7$                      ;;BR IF NO
BIC     #177770,R3             ;;GET RID OF JUNK
BNE     4$                      ;;TEST FOR 0

```

```

3653 024422 005704          TST      R4          ;;SUPPRESS THIS 0?
3654 024424 001403          BEQ     SS          ;;BR IF YES
3655 024426 005204          4S:    INC     R4          ;;DON'T SUPPRESS ANYMORE 0'S
3656 024430 052703 000060      BIS     #'0,R3       ;;MAKE THIS DIGIT ASCII
3657 024434 052703 000040      5S:    BIS     #' ,R3       ;;MAKE ASCII IF NOT ALREADY
3658 024440 110337 024504      MOVB   R3,8S        ;;SAVE FOR TYPING
3659 024444 104400 024504      TYPE   8S          ;;GO TYPE THIS DIGIT
3660 024450 105337 024506      7S:    DECB   $SOCNT    ;;COUNT BY 1
3661 024454 003347          BGT     2S          ;;BR IF MORE TO DO
3662 024456 002402          BLT     6S          ;;BR IF DONE
3663 024460 005204          INC     R4          ;;INSURE LAST DIGIT ISN'T A BLANK
3664 024462 000744          BR      2S          ;;GO DO THE LAST DIGIT
3665 024464 012605          6S:    MOV     (SP)+,R5  ;;RESTORE R5
3666 024466 012604          MOV     (SP)+,R4      ;;RESTORE R4
3667 024470 012603          MOV     (SP)+,R3      ;;RESTORE R3
3668 024472 016666 000002 000004      MOV     2(SP),4(SP)   ;;SET THE STACK FOR RETURNING
3669 024500 012616          MOV     (SP)+,(SP)
3670 024502 000002          RTI                    ;;RETURN
3671 024504          8S:    .BYTE   0          ;;STORAGE FOR ASCII DIGIT
3672 024505          .BYTE   0          ;;TERMINATOR FOR TYPE ROUTINE
3673 024506          .BYTE   0          ;;OCTAL DIGIT COUNTER
3674 024507          .BYTE   0          ;;ZERO FILL SWITCH
3675 024510 000000      $OMODE: .WORD   0      ;;NUMBER OF DIGITS TO TYPE
;*****
.SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
; *THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
; *SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
; *NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
; *BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
; *REPLACED WITH SPACES.
; *CALL:
; *   MOV     NUM,-(SP)      ;;PUT THE BINARY NUMBER ON THE STACK
; *   TYPDS                    ;;GO TO THE ROUTINE
$TYPDS:
MOV     R0,-(SP)          ;;PUSH R0 ON STACK
MOV     R1,-(SP)          ;;PUSH R1 ON STACK
MOV     R2,-(SP)          ;;PUSH R2 ON STACK
MOV     R3,-(SP)          ;;PUSH R3 ON STACK
MOV     R5,-(SP)          ;;PUSH R5 ON STACK
MOV     #20200,-(SP)      ;;SET BLANK SWITCH AND SIGN
MOV     20(SP),R5         ;;GET THE INPUT NUMBER
BPL     1S                ;;BR IF INPUT IS POS.
NEG     R5                ;;MAKE THE BINARY NUMBER POS.
MOVB   #'-,1(SP)         ;;MAKE THE ASCII NUMBER NEG.
1S:    CLR     R0          ;;ZERO THE CONSTANTS INDEX
MOV     $DBLK,R3         ;;SETUP THE OUTPUT POINTER
MOVB   #' ,(R3)+         ;;SET THE FIRST CHARACTER TO A BLANK
2S:    CLR     R2          ;;CLEAR THE BCD NUMBER
MOV     $DTBL(R0),R1     ;;GET THE CONSTANT
3S:    SUB     R1,R5       ;;FORM THIS BCD DIGIT
BLT     4S                ;;BR IF DONE
INC     R2                ;;INCREASE THE BCD DIGIT : 1
BR      3S

```

3709	024576	060105		4\$:	AUD	R1,R5	::	ADD BACK THE CONSTANT
3710	024600	005702			TST	R2	::	CHECK IF BCD DIGIT=0
3711	024602	001002			BNE	5\$::	FALL THROUGH IF 0
3712	024604	105716			TSTB	(SP)	::	STILL DOING LEADING 0'S?
3713	024606	100407			BMI	7\$::	BR IF YES
3714	024610	106316		5\$:	ASLB	(SP)	::	MSD?
3715	024612	103003			BCC	6\$::	BR IF NO
3716	024614	116663	000001 177777		MOVB	1(SP),-1(R3)	::	YES--SET THE SIGN
3717	024622	052702	000060	6\$:	BIS	#'0,R2	::	MAKE THE BCD DIGIT ASCII
3718	024626	052702	000040	7\$:	BIS	#' R2	::	MAKE IT A SPACE IF NOT ALREADY A DIGIT
3719	024632	110223			MOVB	R2,(R3)+	::	PUT THIS CHARACTER IN THE OUTPUT BUFFER
3720	024634	005720			TST	(R0)+	::	JUST INCREMENTING
3721	024636	020027	000010		CMP	R0,#10	::	CHECK THE TABLE INDEX
3722	024642	002746			BLT	2\$::	GO DO THE NEXT DIGIT
3723	024644	003002			BGT	8\$::	GO TO EXIT
3724	024646	010502			MOV	R5,R2	::	GET THE LSD
3725	024650	000764			BR	6\$::	GO CHANGE TO ASCII
3726	024652	105726		8\$:	TSTB	(SP)+	::	WAS THE LSD THE FIRST NON-ZERO?
3727	024654	100003			BPL	9\$::	BR IF NO
3728	024656	116663	177777 177776		MOVB	-1(SP),-2(R3)	::	YES--SET THE SIGN FOR TYPING
3729	024664	105013		9\$:	CLAB	(R3)	::	SET THE TERMINATOR
3730	024666	012605			MOV	(SP)+,R5	::	POP STACK INTO R5
3731	024670	012603			MOV	(SP)+,R3	::	POP STACK INTO R3
3732	024672	012602			MOV	(SP)+,R2	::	POP STACK INTO R2
3733	024674	012601			MOV	(SP)+,R1	::	POP STACK INTO R1
3734	024676	012600			MOV	(SP)+,R0	::	POP STACK INTO R0
3735	024700	104400	024726		TYPE	\$DBLK	::	NOW TYPE THE NUMBER
3736	024704	016666	000002 000004		MOV	2(SP),4(SP)	::	ADJUST THE STACK
3737	024712	012616			MOV	(SP)+,(SP)		
3738	024714	000002			RTI		::	RETURN TO USER
3739	024716	023420		\$DTBL:	1000.			
3740	024720	001750			1000.			
3741	024722	000144			100.			
3742	024724	000012			10.			
3743	024726	000004		\$DBLK:	.9LKW	4		

3744
3745
3746
3747
3748
3749
3750
3751
3752
3753
3754
3755
3756
3757
3758
3759
3760
3761
3762
3763
3764
3765
3766
3767
3768
3769
3770
3771
3772
3773
3774
3775
3776
3777
3778
3779
3780
3781
3782
3783
3784
3785
3786
3787
3788
3789
3790
3791

024736 012737 025064 000024
024744 012737 000340 000026
024752 010046
024754 010146
024756 010246
024760 010346
024762 010446
024764 010546
024766 010637 025070
024772 012737 025004 000024
025000 000000
025002 000776

025004 013706 025070
025010 005037 025070
025014 005237 025070
025020 001375
025022 012605
025024 012604
025026 012603
025030 012602
025032 012601
025034 012600
025036 012737 024736 000024
025044 012737 000340 000026
025052 104400
025054 025072
025056 012716
025060 002144
025062 000002
025064 000000
025066 000776
025070 000000
025072 005015 042522 052123
025100 051101 044524 043516
025106 040440 052106 051105
025114 040440 050040 053517
025122 051105 043040 044501
025130 052514 042522 005015
025136 000
025140

```
*****  
.SBTTL POWER DOWN AND UP ROUTINES  
:POWER DOWN ROUTINE  
$PWRDN: MOV $SILLUP, @#PWRVEC ;; SET FOR FAST UP  
MOV #340, @#PWRVEC+2 ;; PRIORITY  
MOV R0, -(SP) ;; PUSH R0 ON STACK  
MOV R1, -(SP) ;; PUSH R1 ON STACK  
MOV R2, -(SP) ;; PUSH R2 ON STACK  
MOV R3, -(SP) ;; PUSH R3 ON STACK  
MOV R4, -(SP) ;; PUSH R4 ON STACK  
MOV R5, -(SP) ;; PUSH R5 ON STACK  
MOV SP, $SAVR6 ;; SAVE SP  
MOV #PWRUP, @#PWRVEC ;; SET UP VECTOR  
HALT  
BR .-2 ;; HANG UP  
  
:POWER UP ROUTINE  
$PWRUP: MOV $SAVR6, SP ;; GET SP  
CLR $SAVR6 ;; WAIT LOOP FOR THE TTY  
IS: INC $SAVR6 ;; WAIT FOR THE INC  
BNE IS OF WORD  
MOV (SP)+, R5 ;; POP STACK INTO R5  
MOV (SP)+, R4 ;; POP STACK INTO R4  
MOV (SP)+, R3 ;; POP STACK INTO R3  
MOV (SP)+, R2 ;; POP STACK INTO R2  
MOV (SP)+, R1 ;; POP STACK INTO R1  
MOV (SP)+, R0 ;; POP STACK INTO R0  
MOV #PWRDN, @#PWRVEC ;; SET UP THE POWER DOWN VECTOR  
MOV #340, @#PWRVEC+2 ;; PRIORITY  
TYPE PWRMSG ;; REPORT THE POWER FAILURE  
$PWRMG: .WORD PWRMSG ;; POWER FAIL MESSAGE POINTER  
MOV (PC)+, (SP) ;; RESTART AT BEGIN  
$PWRAD: .WORD BEGIN ;; RESTART ADDRESS  
RTI  
$SILLUP: HALT ;; THE POWER UP SEQUENCE WAS STARTED  
BR .-2 ;; BEFORE THE POWER DOWN WAS COMPLETE  
$SAVR6: 0 ;; PUT THE SP HERE  
PWRMSG: .ASCIZ '<15><12>/RESTARTING AFTER A POWER FAILURE<<15><12>'  
  
.EVEN
```

```

3792 ;*****
3793
3794 .SBTTL ROUTINE TO SIZE MEMORY
3795
3796 ;*CALL:
3797 ;* JSR PC,$SIZE
3798 ;* RETURN
3799 ;*$LSTAD WILL CONTAIN THE LAST AVAILABLE MEMORY LOCATION
3800
3801 $SIZE: MOV RO,-(SP) ;;SAVE RO ON THE STACK
3802 MOV R1,-(SP) ;;SAVE R1 ON THE STACK
3803 MOV @#ERRVEC,-(SP) ;;SAVE PRESENT ERROR VECTOR PS & PC
3804 MOV @#ERRVEC+2,-(SP)
3805 MOV SP,RO ;;SAVE THE STACK POINTER
3806 MOV @#PS,@#ERRVEC+2 ;;SET ERRVEC PS TO PRESENT PS
3807 MOV #2$,@#ERRVEC ;;SET FOR TIMEOUT
3808 MOV #20000,R1 ;;FIRST ADDRESS
3809 1$: TST (R1)+ ;;TEST THIS ADDRESS
3810 BR 1$ ;;TRY ANOTHER
3811 2$: SUB #4,R1 ;;DROP BACK
3812 MOV RO,SP ;;RESTORE THE STACK
3813 MOV (SP)+,@#ERRVEC+2 ;;RESTORE ERROR VECTOR
3814 MOV (SP)+,@#ERRVEC
3815 MOV R1,$LSTAD ;;LAST ADDRESS
3816 MOV (SP)+,R1 ;;RESTORE R1
3817 MOV (SP)+,RO ;;RESTORE RO
3818 RTS PC
3819 $LSTAD: .WORD 0 ;;CONTAINS THE LAST ADDRESS

```

```

3820
3821 ;*****
3822
3823 .SBTTL TYPE ROUTINE
3824
3825 ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
3826 ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
3827 ;*NOTE1: $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
3828 ;*NOTE2: $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
3829 ;*NOTE3: $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
3830
3831 ;*
3832 ;*CALL:
3833 ;*1) USING A TRAP INSTRUCTION
3834 ;* TYPE ,MESADR ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
3835 ;*OR
3836 ;* TYPE
3837 ;* MESADR
3838 ;*2) USING A JSR INSTRUCTION
3839 ;* MOV PS,-(SP) ;;PUSH PROCESSOR STATUS WORD ON THE STACK
3840 ;* JSR PC,$TYPE ;;CALL TYPE ROUTINE
3841 ;* MESADDR ;;FIRST ADRESS OF MESSAGE
3842
3843 025234 105737 001155 $TYPE: TSTB $TPFLG ;; IS THERE A TERMINAL?
3844 025240 100002 BPL 1$ ;; BR IF YES
3845 025242 000000 HALT ;; HALT HERE IF NO TERMINAL
3846 025244 000430 BR 3$ ;; LEAVE
3847 025246 010046 1$: MOV RO,-(SP) ;; SAVE RO
3848 025250 017600 000002 MOV @2(SP),RO ;; GET ADDRESS OF ASCIZ STRING
3849 025254 122737 000001 001220 CMPB #APTENV,$ENV ;; RUNNING IN APT MODE
3850 025262 001011 BNE 62$ ;; NO GO CHECK FOR APT CONSOLE
3851 025264 132737 000100 001221 BITB #APTSPool,$ENVM ;; SPOOL MESSAGE TO APT
3852 025272 001405 BEQ 62$ ;; NO GO CHECK FOR CONSOLE
3853 025274 010037 025304 MOV RO,61$ ;; SETUP MESSAGE ADDRESS FOR APT
3854 025300 004737 023466 JSR PC,$ATY3 ;; SPOOL MESSAGE TO APT
3855 025304 000000 61$: .WORD 0 ;; MESSAGE ADDRESS
3856 025306 132737 000040 001221 62$: BITB #APTCSUP,$ENVM ;; APT CONSOLE SUPPRESSED
3857 025314 001003 BNE 60$ ;; YES, SKIP TYPE OUT
3858 025316 112046 2$: MOVB (RO)+,-(SP) ;; PUSH CHARACTER TO BE TYPED ONTO STACK
3859 025320 001005 BNE 4$ ;; BR IF IT ISN'T THE TERMINATOR
3860 025322 005726 TST (SP)+ ;; IF TERMINATOR POP IT OFF THE STACK
3861 025324 012600 60$: MOV (SP)+,RO ;; RESTORE RO
3862 025326 062716 000002 3$: ADD #2,(SP) ;; ADJUST RETURN PC
3863 025332 000002 RTI ;; RETURN
3864 025334 122716 000011 4$: CMPB #THT,(SP) ;; BRANCH IF <HT>
3865 025340 001431 BEQ 8$
3866 025342 122716 000200 CMPB #TCRLF,(SP) ;; BRANCH IF NOT <CRLF>
3867 025346 001007 BNE 5$
3868 025350 005726 TST (SP)+ ;; POP <CR><LF> EQUIV
3869 025352 013746 177776 MOV PS,-(SP) ;; TYPE CR AND LF
3870 025356 004737 025234 JSR PC,$TYPE
3871 025362 001175 $CRLF
3872 025364 000754 BR 2$ ;; GET NEXT CHARACTER
3873 025366 004737 025450 5$: JSR PC,$TYPEC ;; GO TYPE THIS CHARACTER
3874 025372 123726 001154 6$: CMPB $FILLC,(SP)+ ;; IS IT TIME FOR FILLER CHARS.?
3875 025376 001347 BNE 2$ ;; IF NO GO GET NEXT CHAR.

```

```

3876 025400 013746 001152      MOV      $NULL,-(SP)      ;;GET # OF FILLER CHARS. NEEDED
3877                                ;;AND THE NULL CHAR.
3878 025404 105366 000001      7$:     DECB      1(SP)      ;;DOES A NULL NEED TO BE TYPED?
3879 025410 002770                BLT      6$                ;;BR IF NO--GO POP THE NULL OFF OF STACK
3880 025412 004737 025450        JSR      PC,$TYPEC        ;;GO TYPE A NULL
3881 025416 105337 025514        DECB      $CHARCNT        ;;DO NOT COUNT AS A COUNT
3882 025422 000770                BR       7$                ;;LOOP
3883
3884                                ;HORIZONTAL TAB PROCESSOR
3885
3886 025424 112716 000040        8$:     MOVB      #40,(SP)      ;;REPLACE TAB WITH SPACE
3887 025430 004737 025450        9$:     JSR      PC,$TYPEC        ;;TYPE A SPACE
3888 025434 132737 000007 025514    BITB      #7,$CHARCNT        ;;BRANCH IF NOT AT
3889 025442 001372                BNE      9$                ;;TAB STOP
3890 025444 005726                TST      (SP)+            ;;POP SPACE OFF STACK
3891 025446 000723                BR       2$                ;;GET NEXT CHARACTER
3892 025450 105777 153472        $TYPEC: TSTB      @STPS        ;;WAIT UNTIL PRINTER IS READY
3893 025454 100375                BPL      $TYPEC
3894 025456 116677 000002 153464    MOVB      2(SP),@STPB        ;;LOAD CHAR TO BE TYPED INTO DATA REG.
3895 025464 122766 000015 000002    CMPB      #15,2(SP)        ;;BRANCH IF
3896 025472 001003                BNE      1$                ;;NOT <CR>
3897 025474 105037 025514        CLRB      $CHARCNT
3898 025500 000406                BR       $TYPEX
3899 025502 122766 000012 000002    1$:     CMPB      #12,2(SP)    ;;EXIT
3900 025510 002002                BGE      $TYPEX            ;;BRANCH IF
3901 025512 105227                INCB      (PC)+            ;;<LF>
3902 025514 000000                $CHARCNT: .WORD 0          ;;INC SPACE
3903 025516 000207                $TYPEX: RTS PC            ;;COUNT
3904
3905                                ;;
3906                                EQUATES
3907                                THT=11
3908                                TCRLF=200

```


3941						
3942	025554	005015	051412	040524	HEADER: .SBTTL	ASCII MESSAGES
3943	025556	052122	047440	020106	.ASCIZ	(15)(12)(12)/START OF MD-11-DZVSA-A VS60 INSTRUCTION TEST PART I//15//1
3944	025570	042115	030455	026461		
3945	025576	055104	051526	026501		
3946	025604	020101	053040	033123		
3947	025612	020060	047111	052123		
3948	025620	052522	052103	047511		
3949	025626	020116	042524	052123		
3950	025634	050040	051101	020124		
3951	025642	006511	000012			
3952						
3953	025646	040515	047111	027124	EM1: .ASCIZ	/MAINT. SWITCH REG
3954	025654	051440	044527	041524		
3955	025662	020110	042522	000107		
3956	025670	047514	042101	047111	EM2: .ASCIZ	/LOADING D.P.C. REG (STATIC)/
3957	025676	020107	027104	027120		
3958	025704	027103	051040	043505		
3959	025712	036040	052123	052101		
3960	025720	041511	000076			
3961	025724	047514	042101	047111	EM3: .ASCIZ	/LOADING MODE REG/
3962	025732	020107	047515	042504		
3963	025740	051040	043505	000		
3964	025745	123	050505	042525	EM4: .ASCIZ	/SEQUENCING DPC REG/
3965	025752	041516	047111	020107		
3966	025760	050104	020103	042522		
3967	025766	000107				
3968	025770	047514	042101	047111	EM5: .ASCIZ	/LOADING LINE TYPE REG/
3969	025776	020107	044514	042516		
3970	026004	052040	050131	020105		
3971	026012	042522	000107			
3972	026016	047514	042101	047111	EM6: .ASCIZ	/LOADING INTENSITY LEVEL REG/
3973	026024	020107	047111	042524		
3974	026032	051516	052111	020131		
3975	026040	042514	042526	020114		
3976	026046	042522	000107			
3977	026052	047514	042101	047111	EM7: .ASCIZ	/LOADING GRAPHPLOT INC REG/
3978	026060	020107	051107	050101		
3979	026066	050110	047514	020124		
3980	026074	047111	020103	042522		
3981	026102	000107				
3982	026104	044504	050123	040514	EM10: .ASCIZ	/DISPLAY JUMP ABS TO AN ADRS/
3983	026112	020131	052512	050115		
3984	026120	040440	051502	052040		
3985	026126	020117	047101	040440		
3986	026134	051104	000123			
3987	026140	044504	050123	040514	EM11: .ASCIZ	/DISPLAY JUMP REL TO AN ADRS/
3988	026146	020131	052512	050115		
3989	026154	051040	046105	052040		
3990	026162	020117	047101	040440		
3991	026170	051104	000123			
3992	026174	044504	050123	040514	EM12: .ASCIZ	/DISPLAY JSR ABS TO AN ADRS/
3993	026202	020131	051512	020122		
3994	026210	041101	020123	047524		
3995	026216	040440	020116	042101		
3996	026224	051522	000			

3997	026227	104	051511	046120	EM13:	.ASCIZ	/DISPLAY JSR REL TO AN ACRS/
3998	026234	054501	045040	051123			
3999	026242	051040	046105	052040			
4000	026250	020117	047101	040440			
4001	026256	051104	000123				
4002	026262	047514	042101	047111	EM14:	.ASCIZ	/LOADING X POS REG/
4003	026270	020107	020130	047520			
4004	026276	020123	042522	000107			
4005	026304	047514	042101	047111	EM15:	.ASCIZ	/LOADING Y POS REG/
4006	026312	020107	020131	047520			
4007	026320	020123	042522	000107			
4008	026326	052502	020123	044524	EM16:	.ASCIZ	/BUS TIME-OUT ER/
4009	026334	042515	047455	052125			
4010	026342	042440	000122				
4011	026346	044103	051101	052040	EM17:	.ASCIZ	/CHAR TERMINATE ER/
4012	026354	051105	044515	040516			
4013	026362	042524	042440	000122			
4014	026370	020131	054504	040516	EM20:	.ASCIZ	/Y DYNAMIC OFFSET ER/
4015	026376	044515	020103	043117			
4016	026404	051506	052105	042440			
4017	026412	000122					
4018	026414	020130	054504	040516	EM21:	.ASCIZ	/X DYNAMIC OFFSET ER/
4019	026422	044515	020103	043117			
4020	026430	051506	052105	042440			
4021	026436	000122					
4022	026440	042522	047514	040503	EM22:	.ASCIZ	/RELOCATE REG ER/
4023	026446	042524	051040	043505			
4024	026454	042440	000122				
4025	026460	027104	027120	027125	EM23:	.ASCIZ	/D.P.U. NAME REG ER/
4026	026466	047040	046501	020105			
4027	026474	042522	020107	051105			
4028	026502	000					
4029	026503	123	040505	041522	EM24:	.ASCIZ	/SEARCH CODE OF ASSOC. NAME REG ER/
4030	026510	020110	047503	042504			
4031	026516	047440	020106	051501			
4032	026524	047523	027103	047040			
4033	026532	046501	020105	042522			
4034	026540	020107	051105	000			
4035	026545	126	041505	047524	EM25:	.ASCIZ	/VECTOR SCALE ER/
4036	026552	020122	041523	046101			
4037	026560	020105	051105	000			
4038	026565	103	040510	020122	EM26:	.ASCIZ	/CHAR SCALE ER/
4039	026572	041523	046101	020105			
4040	026600	051105	000				
4041	026603	102	044514	045516	EM27:	.ASCIZ	/BLINK LOGIC ER/
4042	026610	046040	043517	041511			
4043	026616	042440	000122				
4044	026622	052111	046101	041511	EM30:	.ASCIZ	/ITALIC LOGIC ER/
4045	026630	046040	043517	041511			
4046	026636	042440	000122				
4047	026642	042515	052516	046040	EM31:	.ASCIZ	/MENU LOGIC ER/
4048	026650	043517	041511	042440			
4049	026656	000122					
4050	026660	042522	042523	020124	EM32:	.ASCIZ	/RESET FAILED TO CLEAR A REG/
4051	026666	040506	046111	042105			
4052	026674	052040	020117	046103			

4053	026702	040505	020122	020101			
4054	026710	042522	000107				
4055	026714	052123	050117	044440	EM33:	.ASCIZ	/STOP INTR ER/
4056	026722	052116	020122	051105			
4057	026730	000					
4058	026731	114	050056	020056	EM34:	.ASCIZ	/L.P. FLAG ER/
4059	026736	046106	043501	042440			
4060	026744	000122					
4061	026746	047503	047514	020122	EM35:	.ASCIZ	/COLOR LEVEL ER/
4062	026754	042514	042526	020114			
4063	026762	051105	000				
4064	026755	111	042524	051516	EM36:	.ASCIZ	/ITENSITY ENABLE - CONS 0 + 1 ER/
4065	026772	052111	020131	047105			
4066	027000	041101	042514	026440			
4067	027006	041440	047117	020123			
4068	027014	020060	020053	020061			
4069	027022	051105	000				
4070	027025	111	052116	051105	EM37:	.ASCIZ	/INTERNAL STOP FLAG FAILED TO SET/
4071	027032	040516	020114	052123			
4072	027040	050117	043040	040514			
4073	027046	020107	040506	046111			
4074	027054	042105	052040	020117			
4075	027062	042523	000124				
4076	027066	047111	042524	047122	EM40:	.ASCIZ	/INTERNAL STOP FLAG FAILED TO CLEAR/
4077	027074	046101	051440	047524			
4078	027102	020120	046106	043501			
4079	027110	043040	044501	042514			
4080	027116	020104	047524	041440			
4081	027124	042514	051101	000			
4082	027131	127	051117	020104	EM41:	.ASCIZ	/WORD 0-1-2 INDICATOR ER/
4083	027136	026460	026461	020062			
4084	027144	047111	044504	040503			
4085	027152	047524	020122	051105			
4086	027160	000					
4087	027161	114	050056	020056	EM42:	.ASCIZ	/L.P. INTR ENABLE - CONS 0 + 1 ER/
4088	027166	047111	051124	042440			
4089	027174	040516	046102	020105			
4090	027202	020055	047503	051516			
4091	027210	030040	025440	030440			
4092	027216	042440	000122				
4093	027222	027114	027120	051440	EM43:	.ASCIZ	/L.P. SWITCH INTR ENABLE - CONS 0 + 1 ER/
4094	027230	044527	041524	020110			
4095	027236	047111	051124	042440			
4096	027244	040516	046102	020105			
4097	027252	020055	047503	051516			
4098	027260	030040	025440	030440			
4099	027266	042440	000122				
4100	027272	044504	050123	040514	EM44:	.ASCIZ	/DISPLAY BUSY ER/
4101	027300	020131	052502	054523			
4102	027306	042440	000122				
4103	027312	054105	042524	047122	EM45:	.ASCIZ	/EXTERNAL STOP LOGIC ER/
4104	027320	046101	051440	047524			
4105	027326	020120	047514	044507			
4106	027334	020103	051105	000			
4107	027341	124	046511	026505	EM46:	.ASCIZ	/TIME-OUT INTR ER/
4108	027346	052517	020124	047111			

```

4109 027354 051124 042440 000122
4110 027362 040516 042515 046440 EM47: .ASCIZ /NAME MATCH INTR ER/
4111 027370 052101 044103 044440
4112 027376 052116 020122 051105
4113 027404 000
4114 027405 114 050056 020056 EMS0: .ASCIZ /L.P. FLAG INTR ER/
4115 027412 046106 043501 044440
4116 027420 052116 020122 051105
4117 027426 000
4118 027427 123 040524 045503 EMS1: .ASCIZ /STACK PTR ER/
4119 027434 050040 051124 042440
4120 027442 000122
4121 027444 051105 050122 020103 DH1: .ASCIZ /ERRPC TSTNUM BUSADR EXPCT RCVD/
4122 027452 020040 051524 047124
4123 027460 046525 020040 052502
4124 027466 040523 051104 020040
4125 027474 054105 041520 020124
4126 027502 020040 041522 042126
4127 027510 000
4128 027512
4129 027512 001116 002060 001122 DT1: .EVEN $ERRPC, TSTNUM, $BDADR, $GDDAT, $BDDAT, 0
4130 027520 001124 001126 000000
4131
4132 ;*****
4133 ;THIS IS THE WORKING AREA FOR ALL VS60 NPR'S (INSTRS & DATA)
4134 ;FROM HERE TO THE END OF 28K
4135 ;*****
4136 027526 000000 BUFFER: 0
4137
4138 000001 .END

```


RESVEC= 000010	321#													
RLO 002074	845#	1108	1110*	1111	1123	1127*	1128	1134*	1157	1159*	1161	1304*	3083*	
RSTVEC 023134	3109*													
RO =%000000	918	1729	2044	3069	3097	3229	3279	3355#						
	242#	906*	910	912*	916	938*	941*	956*	957*	958	973*	974	979*	
	1019*	1020	1022	1028*	1054*	1055	1057	1063*	1073*	1075	1082*	1091*	1093	
	1124*	1125	1127	1132*	1303*	1304	1311*	1398*	1399*	1400*	1401*	1402*	1485*	
	1486	1497*	1522*	1523	1529*	1571*	1574	1588*	1605*	1608	1622*	1638*	1641	
	1656*	1709*	1712	1724*	2018*	2019*	2020*	2026	2033	2042*	2614*	2617	2624*	
	2626	2637*	2640	2647*	2650	2721*	2725	2732*	2734	2745*	2749	2756*	2759	
	2816*	2817*	2818*	2835*	2836*	2837*	2854*	2855*	2856*	2873*	2874*	2875*	2893*	
	2894*	2895*	2896*	2897*	3184*	3199	3206	3223*	3241*	3256	3263	3273*	3338*	
	3341	3357*	3358*	3359*	3360*	3368*	3369*	3445	3453*	3457	3458	3460*	3461*	
	3462	3484*	3559	3560*	3561*	3568*	3569*	3570*	3571*	3572*	3573	3578	3583*	
	3585*	3589	3591	3690	3700*	3704	3720	3721	3734*	3752	3773*	3801	3805*	
	3812	3817*	3847	3848*	3853	3858	3861*	3918	3919*	3920	3921*	3922*	3923*	
	3924*													
R1 =%000001	243#	907*	908	909*	913*	914	915*	1703*	1704*	1705*	2650*	2651*	2652*	
	2653*	2654	2759*	2760*	2761*	2762*	2763	3055*	3056*	3057*	3078*	3079*	3080*	
	3180*	3181*	3182*	3196*	3197*	3237*	3238*	3239*	3253*	3254*	3356*	3361*	3446	
	3483*	3691	3704*	3705	3709	3733*	3753	3772*	3802	3808*	3809	3811*	3815	
	3816*													
R2 =%000002	244#	1440*	1441*	1442	1446*	1447	1455	3692	3703*	3707*	3710	3717*	3718*	
	3719	3724*	3732*	3754	3771*									
R3 =%000003	245#	3632	3641*	3647*	3648*	3651*	3656*	3657*	3658	3667*	3693	3701*	3702*	
	3716*	3719*	3728*	3729*	3731*	3755	3770*							
R4 =%000004	246#	3633	3635*	3636*	3637*	3638	3639*	3653	3655*	3663*	3666*	3756	3769*	
R5 =%000005	247#	3634	3640*	3642*	3644*	3645*	3646*	3647	3665*	3694	3696*	3698*	3705*	
	3709*	3724	3730*	3757	3768*									
R6 =%000006	248#	250	872*	873*	874	944	1720	3064	3091	3204	3261			
R7 =%000007	249#	251												
SETUP1 002400	906#													
SETUP2 002410	908#	911												
SETUP3 002434	914#	917												
SIZE 002054	834#	920*	921*	1442	1446									
SP =%000006	250#	876*	891*	892*	898	899	900	927*	2025	2031	3086	3111	3333*	
	3394*	3397	3399	3400	3429	3430	3434*	3445*	3446*	3453	3454*	3465	3466*	
	3467*	3477	3478*	3483	3484	3519	3542*	3545*	3559*	3564*	3585	3589*	3624*	
	3625	3626	3627*	3632*	3633*	3634*	3640	3665	3666	3667	3668*	3669*	3690*	
	3691*	3692*	3693*	3694*	3695*	3696	3699*	3712	3714*	3716	3726	3728	3730	
	3731	3732	3733	3734	3736*	3737*	3752*	3753*	3754*	3755*	3756*	3757*	3758	
	3764*	3768	3769	3770	3771	3772	3773	3778*	3801*	3802*	3803*	3804*	3805	
	3812*	3813	3814	3816	3817	3847*	3848	3858*	3850	3861	3862*	3864	3865	
	3868	3869*	3874	3876*	3878*	3886*	3890	3894	3895	3899	3918*	3919		
SREG0 002066	842#	1336	1341	1352	1357	1367	1372	1382	1387	1739	1740	1754	1755	
	1772	1773	1788	1789	1802	1803	1820	1821	1836	1837	1850	1851	1864	
	1865	1879	1880	1894	1895	1912	1913	1928	1929	1944	1945	1962	1963	
	1974	1979	1985	2002	2004	2016	2028	2032	3214	3216				
SREG1 002076	846#	2054	2055	2070	2071	2082	2089	2106	2107	2125	2126	2141	2142	
	2163	2164	2179	2180	2200	2201	2219	2221	2954	2955	2961	2972	2978	
	2991	2993*	2994	2999	3015*	3017	3018	3024	3036	3038*	3040	3053	3058*	
	3060	3065	3081	3087	3092	3105	3113							
STACK = 001100	231#	876	927											
STKLMT= 177774	236#													
STKPT 002116	854#	1140	1143*	1144	1225	1227*	1229	1239	1241*	1242	1255	1257*	1258*	
	1259	1265*	1267	1283*	1299*	1320*	1337*	1353*	1368*	1383*	1397*	1439*	1464*	

TST104	013040	2167	2173#	
TST105	013154	2186	2193#	
TST106	013254	2204	2211#	
TST107	013356	2223	2231#	
TST11	003422	1070#		
TST110	013464	2250#		
TST111	013562	2261	2268#	
TST112	013642	2277	2283#	
TST113	013756	2296	2303#	
TST114	014056	2314	2320#	
TST115	014172	2333	2340#	
TST116	014260	2349	2355#	
TST117	014374	2369	2376#	
TST12	003516	1088#		
TST120	014464	2385	2392#	
TST121	014600	2405	2412#	
TST122	014670	2421	2428#	
TST123	015004	2441	2448#	
TST124	015074	2457	2464#	
TST125	015210	2477	2484#	
TST126	015300	2493	2500#	
TST127	015414	2513	2520#	
TST13	003614	1106#		
TST130	015504	2529	2537#	
TST131	015620	2551	2557#	
TST132	015712	2567	2574#	
TST133	016004	2584	2591#	
TST134	016114	2611#		
TST135	016232	2625	2634#	
TST136	016356	2648	2660#	
TST137	016474	2675	2682#	
TST14	003674	1121#		
TST140	016612	2696	2703#	
TST141	016674	2711	2718#	
TST142	017020	2733	2742#	
TST143	017152	2757	2769#	
TST144	017274	2784	2791#	
TST145	017420	2806	2813#	
TST146	017516	2825	2832#	
TST147	017614	2844	2851#	
TST15	003764	1139#		
TST150	017712	2863	2870#	
TST151	020010	2882	2890#	
TST152	020170	2906	2916	2923#
TST153	020320	2943	2948#	
TST154	020430	2962	2970#	
TST155	020520	2980	2986#	
TST156	020634	3000	3008#	
TST157	020774	3022	3025	3033#
TST16	004054	1155#		
TST160	021050	3041	3048#	
TST161	021206	3074#		
TST162	021366	3103#		
TST163	021466	3115	3121#	
TST164	021652	3143	3149#	
TST165	022036	3171	3177#	

TST166	022364	3234#			
TST167	022656	3284#			
TST17	004120	1162	1168#		
TST170	022776	3306#			
TST2	002622	952#			
TST20	004164	1175	1181#		
TST21	004230	1188	1194#		
TST22	004316	1204	1210#		
TST23	004362	1217	1223#		
TST24	004434	1231	1237#		
TST25	004522	1253#			
TST26	004646	1269	1279#		
TST27	004736	1289	1296#		
TST3	002710	969#			
TST30	005046	1310	1317#		
TST31	005120	1325	1335#		
TST32	005206	1344	1351#		
TST33	005274	1360	1366#		
TST34	005362	1375	1381#		
TST35	005450	1390	1395#		
TST36	005672	1409	1416	1424	1430
TST37	006022	1453	1456	1463#	1436#
TST4	003002	987#			
TST40	006132	1482#			
TST41	006250	1503#			
TST42	006330	1512	1518#		
TST43	006426	1536#			
TST44	006504	1544	1550#		
TST45	006550	1557	1563#		
TST46	006766	1597#			
TST47	007204	1631#			
TST5	003052	994	1000#		
TST50	007412	1665#			
TST51	007500	1675	1681#		
TST52	007602	1692	1698#		
TST53	010016	1734#			
TST54	010104	1743	1749#		
TST55	010172	1758	1765#		
TST56	010274	1776	1783#		
TST57	010360	1792	1797#		
TST6	003140	1016#			
TST60	010446	1806	1813#		
TST61	010550	1824	1831#		
TST62	010630	1839	1845#		
TST63	010710	1853	1859#		
TST64	010776	1868	1874#		
TST65	011064	1883	1889#		
TST66	011152	1898	1905#		
TST67	011252	1916	1923#		
TST7	003236	1035#			
TST70	011336	1932	1939#		
TST71	011424	1948	1955#		
TST72	011524	1966	1973#		
TST73	011630	1986	1992#		
TST74	011744	2006	2012#		
TST75	012142	2049#			

VS60 INSTRUCTION TEST PART 1
CROSS REFERENCE TABLE -- USER SYMBOLS

TSY16	012226	2058	2065											
TSY17	012231	2074	2081											
TYPE	104404	3334	3339											
	104400	924	3332	3335	3517	3525	3558	3575	3577	3580	3592	3596	3593	3659
		3335	3336	3935										
TYP00	104401	3566	3590	3936										
TYP01	104403	3938												
TYP05	104402	3937												
XOFF	002100	847	1037	1039*	1040	1053	1057*	1058	1065*	1093*	1170	1172*	1174	1199*
XPOS	002070	843	2237	2238	2251	2258	2270	2275	2814	2823	2833	2842	2891	2902
		2925	2937											
YOFF	002102	848	1002	1004*	1005	1018	1022*	1023	1030*	1075*	1183	1185*	1187	1199*
YPOS	002072	844	2652	2861	2871	2880	2910	2914	2941	2942				
		856	1072	1076	1090	1094	1196	1201						
		955												
STAT	001000	383	389											
STAT	*****	3471	3497											
STAT	023504	3442	3444											
STAT	023460	3440												
STAT	023466	3441	3854											
STAT	023476	3443	3530											
BASE	001254	499	907	937										
ADDR	001122	422	937*	939	940*	954*	971*	988*	1002*	1018*	1037*	1053*	1072*	1090*
		1108*	1123*	1140*	1157*	1170*	1183*	1196*	1212*	1225*	1233*	1255*	1291*	1300*
		1319*	1336*	1352*	1367*	1382*	1396*	1438*	1470*	1491*	1509*	1520*	1537*	1552*
		1576*	1610*	1643*	1672*	1684*	1714*	1739*	1754*	1772*	1788*	1802*	1820*	1836*
		1850*	1864*	1879*	1894*	1912*	1928*	1944*	1962*	1974*	2002*	2016*	2054*	2070*
		2082*	2106*	2125*	2141*	2163*	2179*	2200*	2219*	2237*	2251*	2270*	2289*	2310*
		2326*	2347*	2362*	2383*	2398*	2419*	2434*	2455*	2470*	2491*	2506*	2527*	2539*
		2564*	2581*	2599*	2619*	2642*	2667*	2690*	2709*	2727*	2751*	2778*	2900*	2814*
		2833*	2852*	2871*	2891*	2910*	2925*	2941*	2954*	2972*	2991*	3017*	3036*	3053*
		3081*	3105*	3133*	3161*	3193*	3210*	3250*	3293*	4129				
SBDDAT	001126	424	874	959*	960	975*	976	992*	993	1005*	1006*	1007	1023*	1024*
		1025	1040*	1041*	1042	1058*	1059*	1060	1076*	1077*	1078	1094*	1095*	1096
		1111*	1112	1128*	1129	1144*	1145*	1146	1161*	1174*	1187*	1201*	1202*	1203
		1216*	1229*	1230*	1242*	1243*	1244	1259*	1260*	1261	1267*	1268	1285*	1286
		1305*	1306	1324*	1341*	1342*	1343	1357*	1358*	1359	1372*	1373*	1374	1387*
		1388*	1389	1405*	1406	1412*	1413	1420*	1421	1428*	1429	1449*	1450	1471*
		1472*	1473	1492*	1493*	1494	1511*	1524*	1525*	1526	1541*	1542*	1543	1556*
		1577*	1579	1583*	1584	1586*	1611*	1613	1617*	1618	1620*	1645*	1647	1651*
		1652	1654*	1674*	1690*	1691	1716*	1721*	1740*	1741*	1742	1755*	1756*	1757
		1773*	1774*	1775	1789*	1790*	1791	1803*	1804*	1805	1821*	1822*	1823	1837*
		1838	1851*	1852	1865*	1866*	1867	1880*	1881*	1882	1895*	1896*	1897	1913*
		1914*	1915	1929*	1930*	1931	1945*	1946*	1947	1963*	1964*	1965	1979*	1985*
		2004*	2005	2028*	2032*	2037	2055*	2056*	2057	2071*	2072*	2073	2089*	2090*
		2091	2107*	2108*	2109	2126*	2127*	2128	2142*	2143*	2144	2164*	2165*	2166
		2180*	2181*	2182	2201*	2202*	2203	2221*	2222	2238*	2239*	2240	2258*	2259*
		2260	2276*	2290*	2291*	2292	2311*	2312*	2313	2327*	2329*	2329	2350*	2363*
		2364*	2365	2386*	2399*	2400*	2401	2422*	2435*	2436*	2437	2458*	2471*	2472*
		2473	2491*	2507*	2508*	2509	2530*	2549*	2550	2565*	2566	2582*	2583	2600*
		2601	2620*	2621	2643*	2644	2670*	2671	2691*	2692	2712*	2728*	2729	2752*
		2753	2779*	2780	2801*	2802	2823*	2824	2842*	2843	2861*	2862	2880*	2891
		2902*	2903	2914*	2915	2937*	2942*	2957*	2963*	2978*	2979	2996*	3002*	3020*
		3027*	3042*	3060*	3061*	3067*	3087*	3098*	3092*	3093*	3094	3113*	3114	3135*
		3136	3141*	3142	3163*	3164	3169*	3170	3201*	3205*	3210	3219*	3225*	3226
		3258*	3262*	3267	3275*	3276	3294*	3295*	3296	4129				

\$BELL	001170	444#	3517	3548																
\$CDW1	001260	501#																		
\$CDW2	001262	502#																		
\$CHAPC	025514	3981#	3898	3897#	3902#															
\$CHTAG	001100	410#	872	879	895	886	887													
\$CM1	000002	440#	441#	442#																
\$CM2	000004	440#	441#	442#																
\$CM3	000002	438#	440																	
\$CPUOP	001226	470#																		
\$CRLF	001175	446#	3525	3548	3558	3577	3582	3586	3871	3908										
\$CBLK	024726	3701	3735	3743#																
\$CCH0	001264	503#																		
\$COW1	001256	504#																		
\$COW10	001310	513#																		
\$COW11	001312	514#																		
\$COW12	001314	515#																		
\$COW13	001316	516#																		
\$COW14	001320	517#																		
\$COW15	001322	518#																		
\$COW2	001270	505#																		
\$COW3	001272	506#																		
\$COW4	001274	507#																		
\$COW5	001276	508#																		
\$COW6	001300	509#																		
\$COW7	001302	510#																		
\$COW8	001304	511#																		
\$COW9	001306	512#																		
\$DEVCT	001210	461#																		
\$DEVN	001256	500#																		
\$COAGN	023110	3328	3339	3345#																
\$CTBL	024716	3704	3739#																	
\$ENORD	023100	368	3341#	3537																
\$ENDCT	023046	885	3330#																	
\$ENOMG	023114	3332	3347#																	
\$ENULL	023131	3335	3350#																	
\$ENV	001220	466#	3449	3473	3527	3849														
\$ENVN	001221	467#	902	3451	3851	3856														
\$EOP	023012	3308	3320#																	
\$EOPCT	023040	885#	3327#	3331																
\$ERFLG	001103	413#	3379	3407	3409	3415#	3437	3512#	3548											
\$ERMAX	001115	419#	888#	3409	3432#	3437														
\$ERFOR	023726	879	3510#																	
\$ERPPC	001116	420#	3519#	3520#	3521	3548	3564	4129												
\$ERPTB	001324	538#	3572																	
\$ERPTY	024130	3524	3557#																	
\$ERTTL	001112	417#	3518#	3548																
\$ESCAP	001166	443#	887#	3431#	3543	3545	3548													
\$ETABL	001220	465#																		
\$ETEND	001324	395	521#																	
\$FATAL	001202	458#	3477#																	
\$FFLG	023724	3440#	3443#	3471	3480#	3489#														
\$FILLC	001154	436#	3874	3908																
\$FILLS	001153	435#	3908																	
\$GOADR	001120	421#																		
\$GCCAT	001124	423#	955#	956	960	963#	972#	976	980#	981#	989#	993	1003#	1004						
		1007	1010#	1020#	1021#	1025	1038#	1039	1042	1045#	1055#	1056#	1060	1074#						

1078	1081*	1092*	1096	1099*	1100*	1109*	1110	1112	1115*	1125*	1126*	1129
1142*	1143	1146	1149*	1158*	1171*	1184*	1197*	1203	1213*	1226*	1240*	1241
1244	1247*	1256*	1261	1264*	1268	1282*	1284	1286	1290*	1302*	1306	1309*
1321*	1340*	1343	1356*	1359	1371*	1374	1386*	1389	1404*	1406	1411*	1413
1419*	1421	1427*	1429	1445*	1448*	1450	1455	1466*	1468	1473	1476*	1486*
1487*	1489	1494	1506*	1521*	1526	1530*	1538*	1543	1553*	1568*	1569*	1570
1572*	1579	1581*	1584	1590*	1591	1602*	1603*	1604	1606*	1613	1615*	1618
1624*	1625	1635*	1636*	1637	1639*	1647	1649*	1652	1658*	1659	1670*	1683*
1691	1706*	1707*	1708	1710*	1726*	1727	1737*	1742	1752*	1757	1770*	1775
1786*	1791	1800*	1805	1817*	1823	1833*	1834	1847*	1848	1862*	1867	1877*
1882	1892*	1897	1911*	1915	1926*	1931	1942*	1947	1960*	1965	1978*	1984*
1998*	2005	2017*	2037	2052*	2057	2068*	2073	2088*	2091	2102*	2104	2109
2112*	2121*	2128	2139*	2144	2148*	2159*	2166	2177*	2182	2185*	2196*	2203
2217*	2222	2235*	2240	2244*	2257*	2260	2274*	2287*	2292	2295*	2306*	2313
2323*	2329	2332*	2342*	2359*	2365	2368*	2378*	2395*	2401	2404*	2414*	2421*
2437	2440*	2450*	2467*	2473	2476*	2486*	2503*	2509	2512*	2522*	2547*	2550
2560*	2562	2566	2577*	2579	2583	2593*	2597	2601	2605*	2621	2626*	2627*
2628*	2644	2649*	2654*	2663*	2671	2676*	2686*	2688	2692	2695*	2697*	2707*
2729	2734*	2735*	2736*	2753	2758*	2763*	2772*	2780	2785*	2795*	2798	2802
2805*	2807*	2820*	2824	2839*	2843	2858*	2862	2877*	2881	2899*	2903	2912*
2915	2936*	2950*	2960*	2976*	2979	2992*	2993	3001*	3011*	3015	3026*	3037*
3054*	3082*	3094	3108*	3114	3125*	3136	3140*	3142	3153*	3164	3168*	3170
3183*	3210	3217*	3221*	3226	3240*	3267	3271*	3276	3286*	3296	3299*	3300
4129												
3336*												
227												
390*												
414*	3422*	3473	3425*	3436								
3750	3781*											
418*	3521*	3529	3548	3561								
447*	3548	3908										
3481*	3488*											
415*	889*	3413*	3429*	3434	3436							
416*	890*	934*	953*	970*	1001*	1017*	1036*	1052*	1071*	1099*	1107*	1122*
1141*	1238*	1298*	1465*	1484*	1519*	1567*	1601*	1634*	1702*	2101*	2137*	2175*
2232*	2285*	2322*	2358*	2394*	2430*	2466*	2502*	2612*	2635*	2666*	2683*	2720*
2744*	2776*	2793*	3413	3430*	3436	3542						
920	2593	3815*	3819*									
483*												
487*												
490*												
493*												
391	395	456*	901	2014	3428	3527	3849					
477*												
485*												
488*												
491*												
391*												
3441*	3447	3482*	3487*									
463*	3457*	3460										
464*	3462*											
457*	3455	3463*	3475	3479*								
478*												
486*												
489*												
492*												

SGET42 023070
 SHO = 000000
 SHIBTS 001000
 SICNT 001104
 SILLUP 025064
 SITEMB 001114
 SLF 001176
 SLFLG 023723
 SLPADR 001106
 SLPERR 001110

SLSTAD 025232
 SMAOR1 001232
 SMAOR2 001236
 SMAOR3 001242
 SMAOR4 001246
 SMAIL 001200
 SMAMS1 001230
 SMAMS2 001234
 SMAMS3 001240
 SMAMS4 001244
 SMBADR 001002
 SMFLG 023722
 MSGAD 001214
 MSGLG 001216
 MSGTY 001200
 MTYPE1 001231
 MTYPE2 001235
 MTYPE3 001241
 MTYPE4 001245

MAINDEC-11-DZVSA-A VS60 INSTRUCTION TEST PART 1
DZVSA.P11 CROSS REFERENCE TABLE -- MACRO NAMES

.SCATA	217#	402
.SEOP	217#	3310
.SERRO	217#	3495
.SERRT	217#	3549
.SPARM	217#	
.SPOWE	217#	3745
.SREAD	217#	
.SSAVE	217#	
.SSCOP	217#	3373
.SSIZE	217#	3792
.SSPAC	217#	
.SEWDO	217#	
.STRAP	217#	3909
.STYPD	217#	3676
.STYPE	217#	3821
.STYPO	217#	3598

ADD	909	915	940	1309	1411	1419	1427	1448	2244	2629	2697	2736	2807	3359	3454
	3466	3478	3572	3627	3637	3709	3862								
ASL	1290	2627	2652	2674	2695	2735	2761	2783	2805	3299	3569	3570	3571	3922	
ASLB	3714														
ASR	963	979	980	1010	1028	1045	1063	1115	1132	1476	1497	2624	2647	2732	2756
	3461														
BCC	1291	1477	2245	2606	2696	2806	3715								
BCS	982	1029	1064	1133	1498	2625	2648	2675	2733	2757	2784				
BEJ	903	961	977	994	1008	1026	1043	1061	1079	1097	1113	1130	1147	1162	1175
	1188	1204	1217	1231	1245	1262	1269	1287	1307	1310	1325	1344	1360	1375	1390
	1407	1414	1422	1430	1451	1456	1474	1495	1512	1527	1544	1557	1585	1619	1653
	1675	1692	1743	1758	1776	1792	1806	1824	1839	1853	1868	1883	1898	1916	1932
	1948	1966	2006	2027	2038	2058	2074	2092	2110	2129	2145	2146	2167	2183	2204
	2223	2241	2261	2277	2293	2314	2330	2366	2402	2438	2474	2510	2551	2567	2584
	2602	2622	2645	2672	2693	2730	2754	2781	2803	2825	2844	2863	2882	2904	2916
	2938	2943	2980	3025	3041	3095	3115	3137	3143	3165	3171	3200	3211	3227	3257
	3268	3277	3297	3308	3339	3404	3406	3408	3412	3421	3448	3452	3472	3474	3513
	3516	3541	3544	3574	3579	3592	3654	3852	3865						
BGE	3424	3900													
BGT	3328	3661	3723												
BHI	3410														
BIC	1006	1021	1024	1041	1056	1059	1077	1095	1126	1145	1202	1230	1243	1260	1342
	1358	1373	1388	1472	1487	1493	1525	1542	1572	1583	1606	1617	1639	1651	1710
	1741	1756	1774	1790	1804	1822	1866	1881	1896	1914	1930	1946	1964	2056	2072
	2090	2108	2127	2143	2165	2181	2202	2239	2259	2291	2312	2328	2334	2364	2370
	2400	2406	2436	2442	2472	2478	2508	2514	2653	2762	3061	3088	3093	3295	3325
	3651														
BIS	957	981	1468	1489	1569	1574	1581	1586	1603	1608	1615	1620	1636	1641	1649
	1654	1707	1712	2104	2617	2640	2723	2725	2747	2749	2774	2796	3128	3134	3156
	3162	3188	3194	3245	3251	3656	3657	3717	3718						
BISB	3561														
BIT	1838	1852	2384	2420	2456	2492	2528	2710	3018	3024	3040	3065	3214	3307	3389
	3403	3411	3418	3515	3522	3540									
BITB	902	3451	3851	3856	3888										
BLT	3662	3706	3722	3879											
BMI	1578	1612	1646	1722	1980	2149	2186	2296	2333	2349	2369	2405	2441	2477	2513
	2956	2995	3713												
BNE	875	911	917	923	942	964	1011	1046	1116	1443	1580	1592	1614	1626	1648
	1660	1728	2034	2385	2421	2457	2493	2529	2711	3019	3066	3198	3207	3215	3255
	3264	3301	3362	3370	3390	3419	3450	3456	3459	3476	3523	3528	3538	3562	3584
	3552	3711	3767	3850	3857	3859	3867	3875	3889	3896					
BPL	1083	1101	1150	1248	1531	1589	1623	1657	1717	1725	1986	2113	2962	3000	3535
	3650	3697	3727	3844	3893										
BR	895	943	1289	1312	1409	1416	1424	1453	1458	1719	2030	2036	2040	2043	2151
	2188	2298	2335	2371	2407	2443	2479	2515	2615	2629	2638	2655	2677	2698	2722
	2737	2746	2764	2786	2808	2906	3022	3063	3090	3203	3209	3213	3220	3224	3260
	3266	3270	3274	3291	3392	3399	3401	3414	3417	3442	3464	3533	3567	3594	3628
	3643	3664	3708	3725	3761	3782	3810	3846	3872	3882	3891	3898			
CLR	873	886	887	901	947	991	1030	1065	1081	1099	1134	1158	1171	1184	1197
	1213	1226	1301	1321	1506	1540	1553	1670	1715	1786	1926	1984	2024	2042	2052
	2088	2235	2257	2274	2350	2386	2422	2458	2494	2530	2663	2706	2712	2772	2936
	2957	2960	2996	3001	3020	3026	3037	3059	3067	3084	3195	3218	3223	3252	3273
	3288	3289	3322	3323	3360	3416	3431	3560	3641	3700	3703	3765			
CLRB	929	3415	3480	3481	3482	3729	3897								
CMP	874	898	910	916	944	960	976	993	1007	1025	1042	1060	1078	1096	1112
	1129	1146	1203	1244	1261	1268	1286	1306	1343	1359	1374	1389	1406	1413	1421

	1429	1442	1447	1450	1455	1473	1494	1526	1543	1579	1584	1591	1613	1618	1625
	1647	1652	1659	1691	1720	1727	1742	1757	1775	1791	1805	1823	1867	1882	1897
	1915	1931	1947	1965	2005	2025	2031	2037	2057	2073	2091	2109	2128	2144	2166
	2182	2203	2222	2240	2260	2292	2313	2329	2365	2401	2437	2473	2509	2550	2566
	2583	2601	2621	2644	2671	2692	2729	2753	2780	2802	2824	2843	2862	2881	2903
	2915	2979	3064	3086	3091	3094	3111	3114	3136	3142	3164	3170	3204	3210	3226
	3261	3267	3276	3296	3300	3399	3423	3537	3721						
CMPB	3405	3409	3449	3527	3849	3864	3866	3874	3895	3899					
DEC	941	1247	1656	1658	1724	1726	2112	3197	3254	3326	3361	3369	3568		
DECB	3649	3660	3878	3881											
EMT	232														
HALT	356	3309	3536	3539	3760	3781	3845								
INC	1311	1410	1418	1426	1457	1910	1961	1999	2001	2087	2124	2162	2199	2218	2243
	2256	2309	2346	2382	2418	2454	2490	2526	2544	2546	2563	2580	2598	2668	2689
	2777	2799	2821	2840	2859	2878	2900	2909	2911	2930	2932	2959	3016	3130	3132
	3158	3160	3190	3192	3247	3249	3292	3324	3422	3479	3518	3655	3663	3707	3766
INCB	3427	3512	3901												
IOT	233														
JMP	361	400	3346												
JSR	918	919	1729	2044	2822	2841	2860	2879	2901	2913	2933	3069	3097	3229	3279
	3341	3468	3524	3530	3854	3870	3873	3880	3887						
MOV	872	876	877	878	879	880	881	882	883	884	885	889	890	891	892
	893	896	897	899	900	904	906	907	908	912	913	914	920	926	927
	934	935	936	937	938	946	953	954	955	956	958	959	970	971	972
	973	974	975	988	989	990	992	1001	1002	1003	1004	1005	1017	1018	1019
	1020	1022	1023	1036	1037	1038	1039	1040	1052	1053	1054	1055	1057	1058	1071
	1072	1073	1074	1075	1076	1089	1090	1091	1092	1093	1094	1107	1108	1109	1110
	1111	1122	1123	1124	1125	1127	1128	1140	1141	1142	1143	1144	1156	1157	1159
	1161	1169	1170	1172	1174	1182	1183	1185	1187	1195	1196	1198	1199	1201	1211
	1212	1214	1216	1224	1225	1227	1229	1238	1239	1240	1241	1242	1254	1255	1256
	1257	1258	1259	1264	1265	1267	1280	1281	1282	1283	1284	1285	1297	1298	1299
	1300	1302	1303	1304	1305	1318	1319	1320	1322	1324	1336	1337	1338	1339	1340
	1341	1352	1353	1354	1355	1356	1357	1367	1368	1369	1370	1371	1372	1382	1383
	1384	1385	1386	1387	1396	1397	1398	1399	1400	1401	1402	1403	1404	1405	1412
	1420	1428	1437	1438	1439	1440	1441	1444	1445	1446	1449	1464	1465	1466	1467
	1469	1470	1471	1483	1484	1485	1486	1488	1490	1491	1492	1504	1505	1507	1508
	1509	1511	1519	1520	1521	1522	1523	1524	1537	1538	1539	1541	1551	1552	1554
	1556	1564	1565	1566	1567	1568	1570	1571	1573	1575	1576	1577	1598	1599	1600
	1601	1602	1604	1605	1607	1609	1610	1611	1632	1633	1634	1635	1637	1638	1640
	1642	1643	1645	1666	1667	1668	1669	1671	1672	1674	1682	1683	1684	1685	1686
	1687	1688	1689	1690	1699	1700	1701	1702	1703	1704	1705	1706	1708	1709	1711
	1713	1714	1716	1721	1735	1736	1737	1738	1739	1740	1750	1751	1752	1753	1754
	1755	1766	1767	1768	1769	1770	1771	1772	1773	1784	1785	1787	1788	1789	1798
	1799	1800	1801	1802	1803	1814	1815	1816	1817	1818	1819	1820	1821	1832	1833
	1834	1835	1836	1837	1846	1847	1848	1849	1850	1851	1860	1861	1862	1863	1864
	1865	1875	1876	1877	1878	1879	1880	1890	1891	1892	1893	1894	1895	1906	1907
	1908	1909	1911	1912	1913	1924	1925	1927	1928	1929	1940	1941	1942	1943	1944
	1945	1956	1957	1958	1959	1960	1962	1963	1974	1975	1976	1977	1978	1979	1982
	1983	1985	1993	1994	1995	1996	1997	1998	2000	2002	2004	2013	2014	2015	2016
	2017	2018	2019	2020	2021	2022	2023	2028	2032	2041	2050	2051	2053	2054	2055
	2066	2067	2068	2069	2070	2071	2082	2083	2084	2085	2086	2089	2100	2101	2102
	2103	2105	2106	2107	2119	2120	2121	2122	2123	2125	2126	2136	2137	2138	2139
	2140	2141	2142	2157	2158	2159	2160	2161	2163	2164	2174	2175	2176	2177	2178
	2179	2180	2194	2195	2196	2197	2198	2200	2201	2212	2213	2214	2215	2216	2217
	2219	2221	2232	2233	2234	2236	2237	2238	2251	2252	2253	2254	2255	2258	2269
	2270	2271	2272	2273	2276	2284	2285	2286	2287	2288	2289	2290	2304	2305	2306

	2307	2308	2310	2311	2321	2322	2323	2324	2325	2326	2327	2341	2342	2343	2344
	2345	2347	2357	2358	2359	2360	2361	2362	2363	2377	2378	2379	2380	2381	2383
	2393	2394	2395	2396	2397	2398	2399	2413	2414	2415	2416	2417	2419	2429	2430
	2431	2432	2433	2434	2435	2449	2450	2451	2452	2453	2455	2465	2466	2467	2468
	2469	2470	2471	2485	2486	2487	2488	2489	2491	2501	2502	2503	2504	2505	2506
	2507	2521	2522	2523	2524	2525	2527	2538	2539	2540	2541	2542	2543	2545	2547
	2549	2558	2559	2560	2561	2562	2564	2565	2575	2576	2577	2578	2579	2581	2582
	2592	2593	2594	2595	2596	2597	2599	2600	2612	2613	2614	2616	2618	2619	2620
	2626	2635	2636	2637	2639	2641	2642	2643	2649	2650	2661	2662	2664	2665	2666
	2667	2670	2676	2683	2684	2685	2686	2687	2688	2690	2691	2704	2705	2707	2708
	2709	2719	2720	2721	2724	2726	2727	2728	2734	2743	2744	2745	2748	2750	2751
	2752	2758	2759	2770	2771	2773	2775	2776	2778	2779	2785	2792	2793	2794	2795
	2797	2798	2800	2801	2814	2815	2816	2817	2818	2819	2820	2823	2833	2834	2835
	2836	2837	2838	2839	2842	2852	2853	2854	2855	2856	2857	2858	2861	2871	2872
	2873	2874	2875	2876	2877	2880	2891	2892	2893	2894	2895	2896	2897	2898	2899
	2902	2910	2912	2914	2924	2925	2926	2927	2928	2929	2931	2937	2941	2942	2949
	2950	2951	2952	2953	2954	2963	2971	2972	2973	2974	2975	2976	2978	2987	2988
	2989	2991	2992	2993	2998	3002	3009	3010	3011	3012	3013	3014	3015	3017	3023
	3027	3034	3036	3038	3042	3049	3050	3052	3053	3054	3055	3056	3057	3058	3060
	3075	3076	3077	3078	3079	3080	3081	3082	3083	3085	3087	3092	3104	3105	3106
	3107	3108	3109	3110	3113	3122	3123	3124	3125	3126	3127	3129	3131	3133	3135
	3140	3141	3150	3151	3152	3153	3154	3155	3157	3159	3161	3163	3168	3169	3178
	3179	3180	3181	3182	3183	3184	3185	3185	3187	3189	3191	3193	3196	3201	3205
	3216	3217	3221	3222	3225	3235	3236	3237	3238	3239	3240	3241	3242	3243	3244
	3246	3248	3250	3253	3258	3262	3271	3272	3275	3285	3286	3287	3290	3293	3294
	3329	3333	3338	3355	3356	3357	3358	3368	3394	3395	3397	3400	3413	3425	3426
	3429	3430	3433	3434	3445	3446	3453	3457	3462	3463	3465	3467	3477	3483	3484
	3514	3519	3542	3545	3559	3564	3573	3578	3583	3585	3589	3624	3632	3633	3634
	3640	3647	3665	3666	3667	3668	3669	3690	3691	3692	3693	3694	3695	3696	3701
	3704	3724	3730	3731	3732	3733	3734	3736	3737	3750	3751	3752	3753	3754	3755
	3756	3757	3758	3759	3764	3768	3769	3770	3771	3772	3773	3774	3775	3778	3801
	3802	3803	3804	3805	3806	3807	3808	3812	3813	3814	3815	3816	3817	3847	3848
	3853	3861	3869	3876	3918	3919	3923	3511	3521	3529	3625	3626	3629	3630	3635
MOV8	888	3428	3432	3440	3441	3443	3716	3719	3728	3858	3886	3894	3921	3631	3635
NEG	2651	2760	3636	3698											
NOP	1644	2669	3342	3343	3344										
RESET	928	1160	1173	1186	1200	1215	1228	1266	1323	1510	1555	1673	2003	2220	2275
	2548	2935	2977	2990	3035	3039	3051	3098	3112	3139	3167	3340			
ROL	3642	3644	3645	3646	3648										
RTI	3435	3547	3670	3738	3780	3863									
RTS	3363	3371	3485	3587	3818	3903	3924								
SUB	921	1082	1100	1149	1529	1530	1588	1590	1622	1624	2148	2150	2185	2187	2295
	2297	2332	2368	2404	2440	2476	2512	2605	2654	2763	3460	3520	3705	3811	
TRAP	3926	3936	3937	3938	3939										
TST	894	922	939	2026	2033	2348	2955	2961	3199	3206	3256	3263	3396	3420	3455
	3473	3475	3534	3543	3591	3653	3710	3720	3809	3860	3868	3890	3920		
TSTB	2994	2999	3407	3447	3458	3471	3712	3726	3843	3892					
.ASCII	445	446													
.ASCIZ	444	447	3347	3595	3784	3942	3953	3956	3961	3964	3968	3972	3977	3982	3987
	3992	3997	4002	4005	4008	4011	4014	4018	4022	4025	4029	4035	4038	4041	4044
	4047	4050	4055	4058	4061	4064	4070	4076	4082	4087	4093	4100	4103	4107	4110
	4114	4118	4121												
.BLKW	3743														
.BYTE	412	413	418	419	434	435	436	437	466	467	477	478	485	486	488
	489	491	492	494	495	496	497	3350	3487	3488	3489	3531	3532	3671	3672

CROSS REFERENCE TABLE -- PERMANENT SYMBOLS

415
417

498	498	3490	3596	3791	4128	345	346	347	348	359	362	367	369	374
499	499	499	499	499	499	442	443	444	448	449	477	485	488	491
500	500	500	500	500	500	502	503	504	505	506	507	508	509	510
501	501	501	501	501	501	517	518	519	523	524	525	526	527	528
502	502	502	502	502	502	901	902	903	904	905	906	907	908	909
503	503	503	503	503	503	999	1001	1013	1015	1017	1032	1034	1036	1048
504	504	504	504	504	504	1087	1089	1103	1105	1107	1116	1120	1122	1136
505	505	505	505	505	505	1162	1165	1167	1169	1170	1175	1178	1180	1182
506	506	506	506	506	506	1204	1207	1209	1211	1212	1217	1220	1222	1224
507	507	507	507	507	507	1252	1254	1255	1259	1269	1278	1280	1281	1287
508	508	508	508	508	508	1314	1316	1318	1319	1325	1332	1334	1336	1344
509	509	509	509	509	509	1367	1375	1378	1380	1382	1390	1392	1394	1396
510	510	510	510	510	510	1437	1438	1453	1456	1458	1460	1462	1464	1479
511	511	511	511	511	511	1512	1515	1517	1519	1523	1535	1537	1544	1547
512	512	512	512	512	512	1564	1565	1594	1596	1598	1599	1628	1630	1632
513	513	513	513	513	513	1678	1680	1682	1692	1695	1697	1699	1700	1731
514	514	514	514	514	514	1759	1762	1764	1692	1695	1697	1699	1700	1731
515	515	515	515	515	515	1814	1824	1828	1830	1832	1839	1842	1844	1846
516	516	516	516	516	516	1873	1875	1883	1886	1899	1890	1898	1902	1904
517	517	517	517	517	517	1936	1938	1940	1948	1952	1954	1956	1956	1970
518	518	518	518	518	518	1993	1994	2006	2009	2011	2013	2014	2046	2048
519	519	519	519	519	519	2078	2080	2082	2092	2096	2098	2100	2115	2117
520	520	520	520	520	520	2149	2153	2155	2157	2167	2170	2172	2174	2186
521	521	521	521	521	521	2212	2213	2223	2228	2230	2232	2247	2249	2251
522	522	522	522	522	522	2280	2282	2284	2296	2300	2302	2304	2314	2317
523	523	523	523	523	523	2349	2353	2355	2357	2369	2373	2375	2377	2385
524	524	524	524	524	524	2413	2421	2425	2427	2429	2441	2445	2447	2449
525	525	525	525	525	525	2483	2485	2493	2497	2499	2501	2513	2517	2519
526	526	526	526	526	526	2551	2554	2556	2558	2567	2571	2573	2575	2584
527	527	527	527	527	527	2610	2612	2625	2631	2633	2635	2648	2657	2659
528	528	528	528	528	528	2700	2702	2704	2711	2715	2717	2719	2733	2739
529	529	529	529	529	529	2784	2788	2790	2792	2806	2810	2812	2814	2825
530	530	530	530	530	530	2852	2863	2867	2869	2871	2882	2887	2889	2891
531	531	531	531	531	531	2938	2943	2945	2947	2949	2962	2967	2969	2971
532	532	532	532	532	532	3000	3005	3007	3009	3022	3025	3030	3032	3034
533	533	533	533	533	533	3071	3073	3075	3076	3100	3102	3104	3105	3115
534	534	534	534	534	534	3148	3150	3151	3171	3174	3176	3178	3179	3231
535	535	535	535	535	535	3203	3205	3207	3210	3214	3215	3216	3217	3219
536	536	536	536	536	536	3237	3246	3247	3252	3254	3255	3267	3273	3279
537	537	537	537	537	537	3407	3408	3409	3418	3420	3423	3430	3435	3436
538	538	538	538	538	538	3501	3511	3515	3522	3524	3525	3527	3534	3537
539	539	539	539	539	539	3598	3583	3583	3582	3588	3588	3588	3588	3588
540	540	540	540	540	540	3718	3766	3745	3758	3768	3776	3779	3780	3794
541	541	541	541	541	541	3922	3922	3926	3936	3937	3938	3939	3940	4132

415
417

240	240	343	346	347	348	363	369	371	375	379	386	403	409	412
438	449	450	876	921	932	933	934	950	951	952	953	967	968	969
970	985	986	987	988	995	998	999	1000	1001	1014	1015	1016	1017	1033
1034	1035	1036	1049	1050	1051	1052	1068	1069	1070	1071	1086	1087	1099	1099
1104	1105	1106	1107	1119	1120	1121	1122	1137	1138	1139	1140	1153	1154	1155
1156	1163	1166	1167	1168	1169	1176	1179	1180	1181	1182	1189	1192	1193	1194
1195	1205	1208	1209	1210	1211	1218	1221	1222	1223	1224	1232	1235	1236	1237
1238	1251	1252	1253	1254	1270	1277	1278	1279	1280	1287	1290	1294	1295	1296
1297	1311	1315	1316	1317	1318	1326	1333	1334	1335	1336	1345	1349	1350	1351
1352	1361	1364	1365	1366	1367	1376	1379	1380	1381	1382	1391	1393	1394	1395
1396	1410	1417	1425	1431	1434	1435	1436	1437	1454	1457	1458	1461	1462	1463

3925	3926	3927	3928	3929	3930	3931	3932	3933	3934	3935	3936	3937	3938	3939	3940	3941	3942	3943	3944	3945	3946	3947	3948	3949	3950	3951	3952	3953	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	3996	3997	3998	3999	4000	4001	4002	4003	4004	4005	4006	4007	4008	4009	4010	4011	4012	4013	4014	4015	4016	4017	4018	4019	4020	4021	4022	4023	4024	4025	4026	4027	4028	4029	4030	4031	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	4087	4088	4089	4090	4091	4092	4093	4094	4095	4096	4097	4098	4099	4100	4101	4102	4103	4104	4105	4106	4107	4108	4109	4110	4111	4112	4113	4114	4115	4116	4117	4118	4119	4120	4121	4122	4123	4124	4125	4126	4127	4128	4129	4130	4131	4132	4133	4134	4135	4136	4137	4138	4139	4140	4141	4142	4143	4144	4145	4146	4147	4148	4149	4150	4151	4152	4153	4154	4155	4156	4157	4158	4159	4160	4161	4162	4163	4164	4165	4166	4167	4168	4169	4170	4171	4172	4173	4174	4175	4176	4177	4178	4179	4180	4181	4182	4183	4184	4185	4186	4187	4188	4189	4190	4191	4192	4193	4194	4195	4196	4197	4198	4199	4200	4201	4202	4203	4204	4205	4206	4207	4208	4209	4210	4211	4212	4213	4214	4215	4216	4217	4218	4219	4220	4221	4222	4223	4224	4225	4226	4227	4228	4229	4230	4231	4232	4233	4234	4235	4236	4237	4238	4239	4240	4241	4242	4243	4244	4245	4246	4247	4248	4249	4250	4251	4252	4253	4254	4255	4256	4257	4258	4259	4260	4261	4262	4263	4264	4265	4266	4267	4268	4269	4270	4271	4272	4273	4274	4275	4276	4277	4278	4279	4280	4281	4282	4283	4284	4285	4286	4287	4288	4289	4290	4291	4292	4293	4294	4295	4296	4297	4298	4299	4300	4301	4302	4303	4304	4305	4306	4307	4308	4309	4310	4311	4312	4313	4314	4315	4316	4317	4318	4319	4320	4321	4322	4323	4324	4325	4326	4327	4328	4329	4330	4331	4332	4333	4334	4335	4336	4337	4338	4339	4340	4341	4342	4343	4344	4345	4346	4347	4348	4349	4350	4351	4352	4353	4354	4355	4356	4357	4358	4359	4360	4361	4362	4363	4364	4365	4366	4367	4368	4369	4370	4371	4372	4373	4374	4375	4376	4377	4378	4379	4380	4381	4382	4383	4384	4385	4386	4387	4388	4389	4390	4391	4392	4393	4394	4395	4396	4397	4398	4399	4400	4401	4402	4403	4404	4405	4406	4407	4408	4409	4410	4411	4412	4413	4414	4415	4416	4417	4418	4419	4420	4421	4422	4423	4424	4425	4426	4427	4428	4429	4430	4431	4432	4433	4434	4435	4436	4437	4438	4439	4440	4441	4442	4443	4444	4445	4446	4447	4448	4449	4450	4451	4452	4453	4454	4455	4456	4457	4458	4459	4460	4461	4462	4463	4464	4465	4466	4467	4468	4469	4470	4471	4472	4473	4474	4475	4476	4477	4478	4479	4480	4481	4482	4483	4484	4485	4486	4487	4488	4489	4490	4491	4492	4493	4494	4495	4496	4497	4498	4499	4500	4501	4502	4503	4504	4505	4506	4507	4508	4509	4510	4511	4512	4513	4514	4515	4516	4517	4518	4519	4520	4521	4522	4523	4524	4525	4526	4527	4528	4529	4530	4531	4532	4533	4534	4535	4536	4537	4538	4539	4540	4541	4542	4543	4544	4545	4546	4547	4548	4549	4550	4551	4552	4553	4554	4555	4556	4557	4558	4559	4560	4561	4562	4563	4564	4565	4566	4567	4568	4569	4570	4571	4572	4573	4574	4575	4576	4577	4578	4579	4580	4581	4582	4583	4584	4585	4586	4587	4588	4589	4590	4591	4592	4593	4594	4595	4596	4597	4598	4599	4600	4601	4602	4603	4604	4605	4606	4607	4608	4609	4610	4611	4612	4613	4614	4615	4616	4617	4618	4619	4620	4621	4622	4623	4624	4625	4626	4627	4628	4629	4630	4631	4632	4633	4634	4635	4636	4637	4638	4639	4640	4641	4642	4643	4644	4645	4646	4647	4648	4649	4650	4651	4652	4653	4654	4655	4656	4657	4658	4659	4660	4661	4662	4663	4664	4665	4666	4667	4668	4669	4670	4671	4672	4673	4674	4675	4676	4677	4678	4679	4680	4681	4682	4683	4684	4685	4686	4687	4688	4689	4690	4691	4692	4693	4694	4695	4696	4697	4698	4699	4700	4701	4702	4703	4704	4705	4706	4707	4708	4709	4710	4711	4712	4713	4714	4715	4716	4717	4718	4719	4720	4721	4722	4723	4724	4725	4726	4727	4728	4729	4730	4731	4732	4733	4734	4735	4736	4737	4738	4739	4740	4741	4742	4743	4744	4745	4746	4747	4748	4749	4750	4751	4752	4753	4754	4755	4756	4757	4758	4759	4760	4761	4762	4763	4764	4765	4766	4767	4768	4769	4770	4771	4772	4773	4774	4775	4776	4777	4778	4779	4780	4781	4782	4783	4784	4785	4786	4787	4788	4789	4790	4791	4792	4793	4794	4795	4796	4797	4798	4799	4800	4801	4802	4803	4804	4805	4806	4807	4808	4809	4810	4811	4812	4813	4814	4815	4816	4817	4818	4819	4820	4821	4822	4823	4824	4825	4826	4827	4828	4829	4830	4831	4832	4833	4834	4835	4836	4837	4838	4839	4840	4841	4842	4843	4844	4845	4846	4847	4848	4849	4850	4851	4852	4853	4854	4855	4856	4857	4858	4859	4860	4861	4862	4863	4864	4865	4866	4867	4868	4869	4870	4871	4872	4873	4874	4875	4876	4877	4878	4879	4880	4881	4882	4883	4884	4885	4886	4887	4888	4889	4890	4891	4892	4893	4894	4895	4896	4897	4898	4899	4900	4901	4902	4903	4904	4905	4906	4907	4908	4909	4910	4911	4912	4913	4914	4915	4916	4917	4918	4919	4920	4921	4922	4923	4924	4925	4926	4927	4928	4929	4930	4931	4932	4933	4934	4935	4936	4937	4938	4939	4940	4941	4942	4943	4944	4945	4946	4947	4948	4949	4950	4951	4952	4953	4954	4955	4956	4957	4958	4959	4960	4961	4962	4963	4964	4965	4966	4967	4968	4969	4970	4971	4972	4973	4974	4975	4976	4977	4978	4979	4980	4981	4982	4983	4984	4985	4986	4987	4988	4989	4990	4991	4992	4993	4994	4995	4996	4997	4998	4999	5000	5001	5002	5003	5004	5005	5006	5007	5008	5009	5010	5011	5012	5013	5014	5015	5016	5017	5018	5019	5020	5021	5022	5023	5024	5025	5026	5027	5028	5029	5030	5031	5032	5033	5034	5035	5036	5037	5038	5039	5040	5041	5042	5043	5044	5045	5046	5047	5048	5049	5050	5051	5052	5053	5054	5055	5056	5057	5058	5059	5060	5061	5062	5063	5064	5065	5066	5067	5068	5069	5070	5071	5072	5073	5074	5075	5076	5077	5078	5079	5080	5081	5082	5083	5084	5085	5086	5087	5088	5089	5090	5091	5092	5093	5094	5095	5096	5097	5098	5099	5100	5101	5102	5103	5104	5105	5106	5107	5108	5109	5110	5111	5112	5113	5114	5115	5116	5117	5118	5119	5120	5121	5122	5123	5124	5125	5126	5127	5128	5129	5130	5131	5132	5133	5134	5135	5136	5137	5138	5139	5140	5141	5142	5143	5
------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	---

E09

MA:ND00-11-DZVSA-A VS60 INSTRUCTION TEST PART 1 MACY11 27(732) 20-SEP-76 09:57 PAGE 111
DZVSA.P11 CROSS REFERENCE TABLE -- PERMANENT SYMBOLS

* DZVSA SOL CRF PAGNUM=DZVSA
RUN-TIME: 75 57 11 SECONDS
RUN-TIME RATIO: 486/144=3.3
CORE USED: 24K (47 PAGES)

