

NC11

GAMMA 11 EXERCISER
MD-11-DZNCB-C

EP-DZNCB-C-DL-A
COPYRIGHT © 73-76
FICHE 1 OF 1

DEC 1977
digital
MADE IN USA

This microfiche card contains a grid of frames. The left side of the card features a vertical column of frames, each containing a small table or data set. The right side of the card is mostly blank, with some faint markings and a small number '2' visible near the top center.

B01

.REM %

IDENTIFICATION

PRODUCT CODE:	MAINDEC-11-DZNCB-C-D
PRODUCT NAME:	GAMMA 11 EXERCISER
DATE CREATED:	MAY 1976
MAINTAINER:	DIAGNOSTIC ENGINEERING
AUTHOR:	R. MOORE

COPYRIGHT (C) 1973, 1974 & 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.

0.0 TABLE OF CONTENTS

1.0 ABSTRACT

2.0 REQUIREMENTS

- 2.1 EQUIPMENT
- 2.2 EQUIPMENT/DOCUMENTATION - OPTIONAL
- 2.3 PRELIMINARY PROGRAMS
- 2.4 STORAGE

3.0 LOADING PROCEDURE

4.0 STARTING PROCEDURE

5.0 OPERATOR OPTIONS

- 5.1 OPERATOR KEYBOARD OPTIONS
- 5.2 OPERATOR SWITCH REGISTER OPTION(S)

6.0 RECOMMENDED OPERATOR ACTION

- 6.1 CAMERA/ADC SETUP
- 6.2 JOYSTICK SETUP
- 6.2.1 H306/AR11
- 6.2.2 H306/NC11

7.0 MISCELLANEOUS

- 7.1 DEVICE BUS ADDRESS MODIFICATIONS
- 7.2 FREE RUN MODE (TYPE "F")
- 7.3 ERROR REPORTING
- 7.4 EXECUTION TIME

8.0 PROGRAM DESCRIPTION

- 8.1 DATA CONNECTION & DISPLAY
- 8.2 JOYSTICK MODES
- 8.2.1 H306/AR11
- 8.2.2 H306/NC11

9.0 LISTING

11-DZNCB-0
MACY11
GAMMA 11 EXERCISER MACY11 27(732) 21-SEP-76 15:32 PAGE 2

4.0 STARTING PROCEDURE

LOADING ADDRESS 200 AND STARTING WILL INITIALIZE THE SYSTEM AND AWAIT A KEYBOARD COMMAND.

5.0 OPERATOR OPTIONS

5.1 OPERATOR KEYBOARD OPTIONS

TYPE "N"- COLLECT NEW DATA FROM CAMERA - CLEARS CORE MATRIX FIRST
 TYPE "C"- COLLECT DATA FROM CAMERA - STARTS NC11, PREVIOUS CORE MATRIX DATA NOT CLEARED
 TYPE "S"- STOP NC11 DATA COLLECTION - ALSO TERMINATES FREE RUN MODE
 TYPE "Z"- ZOOM - SET NC11 TO GAIN 1
 TYPE "R"- REGULAR - SET NC11 TO GAIN 2 (DEFAULT COND)
 TYPE "V"- SELECT VTO1 DISPLAY
 TYPE "W"- SELECT VSVO1 DISPLAY (DEFAULT COND USING 1ST BIT MAP)
 TYPE "M"- SELECT VSVO1 DISPLAY AND USE 2ND BIT MAP
 TYPE "E"- ERASE THE SELECTED DISPLAY
 TYPE "A"- DISPLAY ISOTOPE A (DEFAULT COND)
 TYPE "B"- DISPLAY ISOTOPE B (B GAMMA)
 TYPE "D"- DISPLAY DATA - DISPLAY SELECTED ISOTOPE
 TYPE "L"- GET LOWER THRESHOLD FROM KEYBOARD (DEFAULT=0) - ALL MATRIX VALUES LESS THAN TYPED VALUE ARE NOT DISPLAYED
 TYPE "U"- GET UPPER THRESHOLD FROM KEYBOARD (DEFAULT=177777) - ALL MATRIX VALUES GREATER THAN TYPED VALUE ARE NOT DISPLAYED
 TYPE "F"- FREE RUN MODE - COLLECT AND DISPLAY NEW DATA CONTINUOUSLY - TYPE "S" TO TERMINATE THIS MODE
 TYPE "G"- INITIALIZE EVERYTHING - START FRESH
 TYPE "J"- JOYSTICK CALIBRATION USING AR11 - USE "CNTRL C" TO TERMINATE MODE
 TYPE "K"- JOYSTICK CALIBRATION USING NC11 - USE "CNTRL C" TO TERMINATE MODE - USE "CNTRL G" TO TURN OFF/ON BELL (NO CONVERT INDICATION)
 TYPE "T"- DISPLAY INTENSITY TEST PATTERN ON SELECTED DISPLAY - THIS FEATURE IS FOR DISPLAY VERIFICATION ONLY

 TYPE "CNTRL & C" - ABORT WHATEVER - BACK TO KEYBOARD MONITOR
 TYPE "CNTRL & G" - TURN OFF/ON PRINTER BELL UPON A NO CONVERT - NC11 JOYSTICK CAL ONLY

5.2 OPERATOR SWITCH REGISTER OPTION(S)

NONE

110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200

304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360

7.3 ERROR REPORTING

1. INCORRECT KEYBOARD COMMANDS ARE RESPONDED WITH '?'
2. ANY SELECTED DEVICE (SEE SECTION 7.1) WHICH DOES NOT RESPOND ON THE UNIBUS WILL BE INDICATED.

7.4 EXECUTION TIME

THE EXECUTION TIME IS COMPLETELY DEPENDENT UPON THE USER FOR WHATEVER OPTION SELECTED.

8.0 PROGRAM DESCRIPTION

8.1 DATA COLLECTION & DISPLAY

THE USER MAY COLLECT DATA FROM THE GAMMA CAMERA SYSTEM VIA THE NC11 INTERFACE AND DISPLAY THIS DATA ON A VTO1 STORAGE SCOPE OR THE VSVO1 (BITMAP) DISPLAY. ALL FUNCTIONS ARE ENTERED THRU THE KEYBOARD AS DEFINED IN SECTION 5.1. WHEN THE DATA COLLECTION MODE IS SELECTED THE NC11 IS RECEIVING X & Y DATA FROM THE GAMMA CAMERA (LOOKING A RADIOACTIVE SOURCE). THIS INFORMATION IS TRANSFORMED VIA THE ADDRESS MAKER LOGIC WHICH FORMS A UNIQUE ADDRESS WITHIN A DEFINED MATRIX RELATIVE TO THE SCAN POSITION OF THE CAMERA. THE NC11 PREFORMS AN NPR INCREMENT TO MEMORY TO THIS UNIQUE ADDRESS. THE PROGRAM SELECTS THE ADDRESS MATRIX 64X64X15 (RESOLUTION 2) OFFSET TO ADDRESS 20000(8) FOR THE 'A' ISOTOPE, AND FOR THE 'B' ISOTOPE, (CAMERAS EQUIPPED WITH THE DUAL ISOTOPE ATTACHMENT) INFORMATION VIA B GAMMA LOGIC IS STORED STARTING AT ADDRESS 40000 (8). THE INTENSITY OF EACH CELL (MEMORY LOCATION) OF THE IMAGE IN MEMORY IS ADJUSTED WITH THE UPPER AND LOWER THRESHOLDS AND SCALED TO AN INTENSITY LEVEL (32 FOR VTO1 AND 16 FOR VSVO1). THE CELL WITH THE HIGHEST COUNT REPRESENTS THE HIGHEST INTENSITY LEVEL, THEREFORE APPEARS BRIGHTEST ON THE DISPLAY. AT THE COMPLETION OF THE IMAGE DISPLAY, THE FOLLOWING PARAMETERS ARE DISPLAYED:

- | | |
|------------------|--|
| LOWER THRESHOLD- | ALL VALUES GREATER THAN THIS VALUE ARE DISPLAYED |
| UPPER THRESHOLD- | ALL VALUES ABOVE THIS VALUE ARE OMITTED |
| Z COUNT- | 32 BIT COUNTER OF A/D START PULSES |
| MATRIX COUNT- | TOTAL COUNT OF THE CONTENTS OF EACH CELL IN THE 64X64 MATRIX |
| ZOOM- | DISPLAYED IF GAIN WAS SELECTED |
| B-GAMMA- | DISPLAYED IF 'B' ISOTOPE SELECTED |

057
058
059
060
061
062
063
064
065
066
067
068
069
070
071
072
073
074
075
076
077
078
079
080
081
082
083
084
085
086
087
088
089
090
091
092
093
094
095
096
097
098
099
100
101
102
103
104
105
106
107
108
109
110
111
112

8.2 JOYSTICK MODES

8.2.1 H306/AR11

SELECTING THIS MODE WILL PROVIDE THE USER WITH A CHECK OF THE ABILITY OF THE H306 WHEN CONNECTED TO THE AR11 TO DISPLAY A X AND Y INDICATION ON THE SELECTED DISPLAY. THIS INDICATION SHOULD AGREE WITH THE PROCEDURE DEFINED IN SECTION 6.2.1. THE AR11 A/D CHANNELS HAVE THE FOLLOWING RELATIONSHIP: CHAN 0 = Y DATA, CHAN 1 = X DATA AND CHAN 2 IS THE INTERRUPT BAR INDICATOR (0=DEPRESSED COND.).

8.2.2 H306/NC11

SELECTING THE NC11 CONFIGURATION TAKES ADVANTAGE OF THE JOYSTICK MODE PROVIDED BY THE NC11. THIS MODE SELECTS THE EXTERNAL INPUTS AND THE NC11 PERFORMS LIKE A A/D WITH THE X & Y CONVERTED VALUES AVAILABLE IN X-Y HOLDING REGISTER. THE X AND Y DATA IS APPLIED TO THE SELECTED DISPLAY AND SHOULD CONFORM TO THE REQUIREMENTS DEFINED IN SECTION 6.2.2.

NOTE: WHEN USING THE VTO1 DISPLAY THE WRITE - THRU MODE IS USED WHICH DISPLAYS A SMALL CIRCLE OR DOT. THE VSV01 USES THE X & Y CROSS HAIRS AS THE DISPLAY INDICATOR.

9.0 LISTING

```

%
.TITLE MAINDEC-11-DZNCB-C GAMMA 11 EXERCISER
.*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-DZGAC-CO),MAR 21, 1976.
.*
$TN=1
$SWR=160000      ::HALT ON ERROR, LOOP ON TEST, INHIBIT ERROR TYP0UT
.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
```

000001
160000

001100

413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468

000011
000012
000015
000200
177776

177774
177772
177570
177570

000000
000001
000002
000003
000004
000005
000006
000007

000000
000040
000100
000140
000200
000240
000300
000340

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

;*MISCELLANEOUS DEFINITIONS

HT= 11 ;: CODE FOR HORIZONTAL TAB
LF= 12 ;: CODE FOR LINE FEED
CR= 15 ;: CODE FOR CARRIAGE RETURN
CRLF= 200 ;: CODE FOR CARRIAGE RETURN-LINE FEED
PS= 177776 ;: PROCESSOR STATUS WORD
.EQUIV PS,PSW
STKLMT= 177774 ;: STACK LIMIT REGISTER
PIRQ= 177772 ;: PROGRAM INTERRUPT REQUEST REGISTER
DSWR= 177570 ;: HARDWARE SWITCH REGISTER
DDISP= 177570 ;: HARDWARE DISPLAY REGISTER

;*GENERAL PURPOSE REGISTER DEFINITIONS

R0= %0 ;: GENERAL REGISTER
R1= %1 ;: GENERAL REGISTER
R2= %2 ;: GENERAL REGISTER
R3= %3 ;: GENERAL REGISTER
R4= %4 ;: GENERAL REGISTER
R5= %5 ;: GENERAL REGISTER
R6= %6 ;: GENERAL REGISTER
R7= %7 ;: GENERAL REGISTER
.EQUIV R6,SP ;: STACK POINTER
.EQUIV R7,PC ;: PROGRAM COUNTER

;*PRIORITY LEVEL DEFINITIONS

PR0= 0 ;: PRIORITY LEVEL 0
PR1= 40 ;: PRIORITY LEVEL 1
PR2= 100 ;: PRIORITY LEVEL 2
PR3= 140 ;: PRIORITY LEVEL 3
PR4= 200 ;: PRIORITY LEVEL 4
PR5= 240 ;: PRIORITY LEVEL 5
PR6= 300 ;: PRIORITY LEVEL 6
PR7= 340 ;: PRIORITY LEVEL 7

;*SWITCH REGISTER SWITCH DEFINITIONS

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

469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
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
259

```

525
526      000000
527
528
529
530      000174
531 000174 000000
532 000176 000000
533
534 000200 000137 001356

```

```

      .=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)
JMP     3#START ;;JUMP TO STARTING ADDRESS OF PROGRAM

```

.SBTTL COMMON TAGS

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

535
536
537
538
539
540
541 001100
542 001100
543 001100 000000
544 001102 000
545 001103 000
546 001104 000000
547 001106 000000
548 001110 000000
549 001112 000000
550 001114 000
551 001115 001
552 001116 000000
553 001120 000000
554 001122 000000
555 001124 000000
556 001126 000000
557 001130 000000
558 001132 000000
559 001134 000
560 001135 000
561 001136 000000
562 001140 177570
563 001142 177570
564 001144 177560
565 001146 177562
566 001150 177564
567 001152 177566
568 001154 000
569 001155 002
570 001156 012
571 001157 000
572 001160 077
573 001161 015
574 001162 000012
575

.=1100
\$CMTAG: .WORD 0
\$PASS: .WORD 0
\$STNM: .BYTE 0
\$ERFLG: .BYTE 0
\$ICNT: .WORD 0
\$LPADR: .WORD 0
\$LPERR: .WORD 0
\$ERTTL: .WORD 0
\$ITEMB: .BYTE 0
\$ERMAX: .BYTE 1
\$ERRPC: .WORD 0
\$GDADR: .WORD 0
\$BDADR: .WORD 0
\$GDDAT: .WORD 0
\$BDDAT: .WORD 0
\$SAUTOB: .BYTE 0
\$INTAG: .BYTE 0
\$SWR: .WORD DSWR
\$DISPLAY: .WORD DDISP
\$TKS: 177560
\$TKB: 177562
\$TPS: 177564
\$TPB: 177566
\$NULL: .BYTE 0
\$FILLS: .BYTE 2
\$FILLC: .BYTE 12
\$TPFLG: .BYTE 0
\$QUES: .ASCII /?/
\$CRLF: .ASCII <15>
\$LF: .ASCIZ <12>

:: START OF COMMON TAGS
:: CONTAINS PASS COUNT
:: CONTAINS THE TEST NUMBER
:: CONTAINS ERROR FLAG
:: CONTAINS SUBTEST ITERATION COUNT
:: CONTAINS SCOPE LOOP ADDRESS
:: CONTAINS SCOPE RETURN FOR ERRORS
:: CONTAINS TOTAL ERRORS DETECTED
:: CONTAINS ITEM CONTROL BYTE
:: CONTAINS MAX. ERRORS PER TEST
:: CONTAINS PC OF LAST ERROR INSTRUCTION
:: CONTAINS ADDRESS OF 'GOOD' DATA
:: CONTAINS ADDRESS OF 'BAD' DATA
:: CONTAINS 'GOOD' DATA
:: CONTAINS 'BAD' DATA
:: RESERVED--NOT TO BE USED
:: AUTOMATIC MODE INDICATOR
:: INTERRUPT MODE INDICATOR
:: ADDRESS OF SWITCH REGISTER
:: ADDRESS OF DISPLAY REGISTER
:: TTY KBD STATUS
:: TTY KBD BUFFER
:: TTY PRINTER STATUS REG. ADDRESS
:: TTY PRINTER BUFFER REG. ADDRESS
:: CONTAINS NULL CHARACTER FOR FILLS
:: CONTAINS # OF FILLER CHARACTERS REQUIRED
:: INSERT FILL CHARS. AFTER A "LINE FEED"
:: "TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
:: QUESTION MARK
:: CARRIAGE RETURN
:: LINE FEED

::*****

576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618

.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'SITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
;*NOTE1: IF \$ITEMB IS 0 THE ONLY PERTINENT DATA IS (\$ERRPC).
;*NOTE2: EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:

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

001164

\$ERRTB:
;THIS PROGRAM DOES NOT USE THE ABOVE ERROR TABLE

;NC11 BUS ADRS ASSIGNMENTS - MODS ARE TO BE MADE HERE

001164 164000

NCADR: 164000

;VS01 BUS ADRS ASSIGNMENTS (CHAR GEN) - MODS ARE TO BE MADE HERE

001166 172600

VTVADR: 172600

;VS01 BUS ADRS ASSIGNMENTS (BIT MAP) - MODS ARE TO BE MADE HERE

001170 172620

VTMADR: 172620

;VT01 BUS ADRS ASSIGNMENTS - MODS ARE TO BE MADE HERE

001172 176756

VTADR: 176756

;AR11 BUS ADRS ASSIGNMENTS - MODS ARE TO BE MADE HERE

001174 170400

ARADR: 170400

:COMMON PROGRAM TAGS AND STORAGE LOCATIONS

001246	0000000
001250	0000000
001252	0000000
001254	0000000
001256	0000020
001260	0000000
001262	0064000
001264	0000000
001266	0000000
001270	0000003
001272	0000000
001274	0000000
001276	0000000
001300	0000000
001302	0000000
001304	0000000
001306	0000000
001310	0000000
001312	0000000
001314	1777777
001316	0000000
001320	0000000
001322	0100000
001324	0000000
001326	0000000
001330	0000000
001332	0000006
001334	0000000
001336	0000000
001340	0200000
001342	0000000
001344	0000100
001346	0000000
001350	0200000
001352	0000000
001354	0000000

TEMPO: 0
 TEMP1: 000
 TEMP2: 0000
 DTYPE: 0000
 INTENS: 0000
 INTLUT: 0000
 VTSVAV: 6400
 MRXADR: 0000
 MAPADR: 0000
 PIXCNT: 0000
 PIXASM: 0000
 HORIZ: 0000
 VERT: 0000
 XREF: 0000
 YREF: 0000
 KBDBUF: 0000
 KBUFF: 0000
 TTYOUT: 0000
 THLO: 0000
 THHI: 177777
 COMSAV: 0000
 GAIN: 0000
 TOTSIZ: 4096
 CHARCT: 0000
 BASX: 0000
 BASY: 0000
 INCXY: 0000
 CPERL: 0000
 ROWCNT: 0000
 TABLEX: MATRIX
 FREERN: 0000
 TIME: 10
 JTYPE: 0000
 ARCHAN: 020000
 MSELECT: 000000
 BELLEN: 0

:COMMON UTILITY LOC
 :COMMON UTILITY LOC
 :COMMON UTILITY LOC
 :0=VSVO1 DISPLAY, -1=VTO1 DISPLAY
 :TOTAL # OF INTENSITY LEVELS, 16 OR 32
 :INITIAL INTENSITY SHADE
 :VSVO1 SET UP - FULL SCREEN, MONO(BLK/WHT), ENA DISPLAY
 :CURRENT CORE ADRS OF CELL BEING DISPLAYED
 :CURRENT ADRS OF BIT MAP LD
 :CURRENT PIXEL BYTE COUNT
 :4 PIXELS ASSEMBLED HERE BEFORE BIT MAP LD
 :COUNTS 64 DISPLAYED POSITIONS EACH ROW (VTO1)
 :COUNTS 64 ROWS FOR VTO1 DISPLAY
 :X POS FOR VTO1 DISPLAY
 :Y POS FOR VTO1 DISPLAY
 :CONTAINS ASCII CHAR THAT NEEDS CONVERSION FOR VTO1 DISP
 :CONTAINS KEYBOARD CHAR TYPED
 :OUTPUT CHAR TO PRINTER
 :LOW THRESHOLD VALUE APPLIED TO MATRIX CELLS
 :HIGH THRESHOLD VALUE APPLIED TO MATRIX CELLS
 :SAVED NC11 CSR CONTENTS WHEN DISPLAYING
 :GAIN 2=0, GAIN 1=40
 :TOTAL # OF CELLS IN MATRIX (ISOTOPE A OR B)
 :COUNTS TOTAL # OF CELLS IN MATRIX (ISOTOPE A OR B)
 :CONTAINS CURRENT X DAC POSITION (VTO1)
 :CONTAINS CURRENT Y DAC POSITION (VTO1)
 :CONTAINS CHAR SIZE OFFSET (VTO1)
 :CONTAINS LARGEST CELL OF MATRIX WITHIN THRESHOLDS
 :COUNTS 64 CELLS EACH ROW OF CORE MATRIX
 :CONTAINS STARTING ADRS OF MATRIX OF SELECTED ISOTOPE
 :NON-ZERO SAYS COLLECT NEW DATA & DISPLAY IN FREE RUN MO
 :HIGH ORDER COUNT DELAY FOR COLLECTING DATA-FREE RUN MOD
 :0=NC11 JOYSTICK SOURCE, -1=AR11 JOYSTICK SOURCE
 :CONTAINS CURRENT AR11 CHAN & UNIPOLAR BIT
 :0 SAYS 1ST BIT MAP, -1 SAYS 2ND BIT MAP(VSVO1)
 :0 SAYS RING BELL ON NO CONVERT, -1 SAYS DON'T


```

        .SBTTL MAIN PROGRAM

START:
.SBTTL INITIALIZE THE COMMON TAGS
::CLEAR THE COMMON TAGS ($CMTAG) AREA
MOV     #SCMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
CLR     (R6)+           ;;CLEAR MEMORY LOCATION
CMP     #SWR,R6        ;;DONE?
BNE     -6              ;;LOOP BACK IF NO
MOV     #STACK,SP      ;;SETUP THE STACK POINTER
::INITIALIZE A FEW VECTORS
MOV     #STRAP,3#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
MOV     #340,3#TRAPVEC+2;LEVEL 7
MOV     #SPWRDN,3#PWRVEC ;;POWER FAILURE VECTOR
MOV     #340,3#PWRVEC+2 ;;LEVEL 7
::SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
::EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
MOV     3#ERRVEC, -(SP) ;;SAVE ERROR VECTOR
MOV     #64$,3#ERRVEC   ;;SET UP ERROR VECTOR
MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
CMP     #-1,3#SWR       ;;TRY TO REFERENCE HARDWARE SWR
BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                        ;;AND THE HARDWARE SWR IS NOT = -1
BR      65$             ;;BRANCH IF NO TIMEOUT
MOV     #65$, (SP)      ;;SET UP FOR TRAP RETURN
RTI
MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
MOV     #DISPREG,DISPLAY
MOV     (SP)+,3#ERRVEC ;;RESTORE ERROR VECTOR

TYPE    MSG1            :GO IDENTIFY PROGRAM
JSR     PC,SELCTA       :DEFAULT TO VSV01 DISPLAY IF THERE
BCC     3$              :BR IF DISPLAY VSV01 IS THERE
JSR     PC,SELCTB       :IF NOT GO LOOK FOR VTO1 DISPLAY
BCC     3$              :BR IF AT LEAST THERE IS A VTO1 DISPLAY
TYPE    .MSG3           :GO TYPE 'BUS TIMEOUT ER - VSV01 DISPLAY'
TYPE    .MSG4           :GO TYPE 'BUS TIMEOUT ER - VTO1 DISPLAY'
HALT    :NO DISPLAY SEEN AT ASSIGNED BUS ADRS'S
3$:     MOV     NCADR,R0 :GET NC11 BASE ADRS
        MOV     #NCCSR,R1 :GET POINTER ADRS
NCSET:  MOV     R0,(R1)+  :SET UP NC11 REG ADRS PTRS
        ADD     #2,R0     :BUMP REG ADRS
        CMP     #VTVCRG,R1 :ALL SET UP?
        BNE     NCSET    :BR IF NOT
        MOV     #1$,ERRVEC :SET UP TIMEOUT ADRS
        TST     3#NCCSR  :WILL TRAP TO LOC 4 IF NOT THERE
        BR      START1   :BR IF THERE
1$:     CMP     (SP)+,(SP)+ :FIX STACK SINCE NO RTI
        TYPE    .MSG5     :GO TYPE 'BUS TIMEOUT ER - NC11'
        HALT    :NC11 NOT SEEN AT ASSIGNED BUS ADRS
START1: MOV     #ERRVEC+2,3#ERRVEC :RESTORE ERR TRAP LOC TO PT TO 5
        RESET   :CLR WORLD
        JSR     PC,NCSTP1 :GO STOP NC11 & CLR CORE MATRIX
        MOV     #MATRIX,3#NCOFF :SET OFFSET(ADRS 20000)
        CLR    GAIN      :REGULAR GAIN

```

74	001644	012737	020000	001340
75	001652	002037	001312	
76	001656	012737	177777	001314
77	001664	012737	006122	000060
78	001672	012737	000240	000060
79	001700	012777	000100	177266
80	001706	104400	012643	

```

MOV #MATRIX.TABLEX
CLR THLO
MOV #177777,THHI
MOV #KBINT,TKVEC
MOV #340,TKVEC+2
MOV #100,3$TKS
TYPE .MSG6

```

```

: ISOTOPE A
: CLEAR LOWER THRESHOLD
: CEILING THRESHOLD
: SET KEYBOARD INTR RETURN ADDR
: SET UP NEW PRIORITY ON INTR
: SET INTR ENABLE
: GO ASK FOR KEYBOARD COMMAND(S)

```

:THIS IS A LIST OF KEYBOARD COMMANDS FOR THE NC11 DISPLAY/JOYSTICK

```

:D      DISPLAY DATA
:E      ERASE THE SCOPE
:L      GET LOWER THRESHOLD FROM KEYBOARD & DISPLAY DATA
:U      GET UPPER THRESHOLD FROM KEYBOARD & DISPLAY DATA
:N      COLLECT NEW DATA FROM CAMERA (CLEAR CORE MATRIX)
:C      COLLECT DATA FROM CAMERA
:S      STOP COLLECTION
:G      INITIALIZE EVERYTHING
:Z      ZOOM - SET GAIN TO 1
:R      REGULAR - SET GAIN TO 2
:A      DISPLAY ISOTOPE A
:B      DISPLAY ISOTOPE B
:V      SELECT VTO1 DISPLAY
:W      SELECT VSVO1 DISPLAY (DEFAULT USING 1ST BIT MAP)
:M      SELECT VSVO1 DISPLAY AND USE 2ND BIT MAP
:F      FREE RUN MODE - COLLECT & DISPLAY NEW DATA CONTINUALLY
:J      JOYSTICK CALIBRATION USING AR11
:K      JOYSTICK CALIBRATION USING NC11
:T      DISPLAY INTENSITY TEST IMAGE ON SELECTED DISPLAY
:CNTRL C  ABORT WHATEVER - BACK TO KEYBOARD MONITOR
:CNTRL G  TURN ON/OFF BELL (NO CONVERT INDICATION) - IN NC11
:JOYSTICK CALIBRATION ONLY
    
```

:THIS CODE WILL DISPATCH PROGRAM TO THE PROPER ROUTINE VIA THE DISPATCH TABLE 'RTABLE'

```

LISEN: CLR    FREERN      :KNOCK DOWN FREE RUN MODE IF SET
        MOV    #STACK,SP  :RESET STACK PTR
        CLR    KBUFF      :INSURE NO KEYBOARD GARBAGE
        CLR    PSW        :ALLOW KEYBOARD INTR
LISN:  MOV    KBUFF,RO    :LOOK FOR CHAR
        BEQ    LISN       :WAIT FOR ONE
        CLR    KBUFF      :
        CMP    RO,#101    :WAS IT A CHAR?
        BCS    B00B00     :NOT A GOOD CHAR
        BIC    #177740,RO :ELIMINATE LOWER CASE
        CMP    RO,#33     :IS IT AN ALPHA CHAR?
        BCC    B00B00     :BR IF NOT
        ASL    RO         :MAKE UP WORD OFFSET
        TST   RTABLE-2(RO) :IS IT A LEGAL COMMAND?
        BEQ    B00B00     :BR IF NOT
        JMP    @RTABLE-2(RO) :GO DO IT
    
```

:KEYBOARD DISPATCH TABLE

```

RTABLE: ROUTA  :A DISPLAY ISOTOPE A
        ROUTB  :B DISPLAY ISOTOPE B
        ROUTC  :C COLLECT DATA
        ROUTD  :D DISPLAY DATA
        ROUTE  :E ERASE SCOPE
        ROUTF  :F FREE RUN MODE(COLLECT NEW DATA & DISPLAY CONTINUOUSL
        ROUTG  :G INITIALIZE EVERYTHING
        O      :H B00B00
        O      :I B00B00
        ROUTJ  :J GO TO AR11 JOYSTICK CALIBRATION
    
```

```

7705
7706
7707
7708
7709
7710
7711
7712
7713
7714
7715
7716
7717
7718
7719
7720
7721
7722
7723
7724
7725
7726
7727
7728
7729
7730
7731
7732
7733
7734
7735
7736
7737
7738
7739
7740
7741
7742
7743
7744
7745
7746
7747
7748
7749
7750
7751
7752
7753
7754
7755
7756
7757
7758
7759
7760
7761
7762
7763
7764
7765
7766
7767
7768
7769
7770
7771
7772
7773
7774
7775
7776
7777
7778
7779
7780
7781
7782 001712 005037 001342
7783 001716 012706 001100
7784 001722 005037 001306
7785 001726 005037 177776
7786 001732 013700 001306
7787 001736 001775
7788 001740 005037 001306
7789 001744 020027 000101
7790 001750 103445
7791 001752 042700 177740
7792 001756 020027 000033
7793 001762 103040
7794 001764 006300
7795 001766 005760 001776
7796 001772 001434
7797 001774 000170 001776
7798
7799
8000
8001 002000 002104
8002 002002 002116
8003 002004 002130
8004 002006 002140
8005 002010 002144
8006 002012 002154
8007 002014 002250
8008 002016 000000
8009 002020 000000
8010 002022 002260
    
```

011	002024	002302			ROUTK		:'K'	GO TO NC11 JOYSTICK CALIBRATION
012	002026	002312			ROUTL		:'L'	GET LOWER THRESHOLD FROM KEYBOARD
013	002030	002342			ROUTM		:'M'	SELECT VSVO1 DISPLAY AND USE 2ND BIT MAP
014	002032	002402			ROUTN		:'N'	GET NEW DATA FROM CAMERA
015	002034	000000			O		:'O'	BOOB00
016	002036	000000			O		:'P'	BOOB00
017	002040	000000			O		:'Q'	BOOB00
018	002042	002420			ROUTR		:'R'	REGULAR GAIN
019	002044	002436			ROUTS		:'S'	STOP COLLECTION
020	002046	002454			ROUTT		:'T'	DISPLAY INTENSITY TEST IMAGE ON SELECTED DISPLAY
021	002050	002470			ROUTU		:'U'	GET UPPER THRESHOLD FROM KEYBOARD
022	002052	002520			ROUTV		:'V'	SELECT VTO1 DISPLAY
023	002054	002540			ROUTW		:'W'	SELECT VSVO1 DISPLAY (DEFAULT COND USING 1ST BIT MAP
024	002056	000000			O		:'X'	BOOB00
025	002060	000000			O		:'Y'	BOOB00
026	002062	002576			ROUTZ		:'Z'	ZOOM - SET GAIN TO 1

:REPORTS ILLEGAL KEYBOARD CHARACTERS

030	002064	012737	000077	001310	BOOB00:	MOV	#77,TTYOUT	:SET UP '??'
031	002072	004737	006174			JSR	PC,TYPE	:TYPE IT
032	002076	004737	006154			JSR	PC,TYPECR	:DO A 'CR'
033	002102	000713				BR	LISN	:GO LOOK FOR GOOD CHAR
035	002104	012737	020000	001340	ROUTA:	MOV	#MATRIX, TABLEX	:WILL DISPLAY ISOTOPE A
036	002112	000137	002616			JMP	CHANGE	:GO DO IT
037	002116	012737	040000	001340	ROUTB:	MOV	#MATRIX+20000, TABLEX	:WILL DISPLAY ISOTOPE B
038	002124	000137	002616			JMP	CHANGE	:GO DO IT
039	002130	004737	005440		ROUTC:	JSR	PC,NCSTRT	:GO START NC11
040	002134	000137	001732			JMP	LISN	:COLLECT DATA UNTIL KEYBRD COMMAND
041	002140	000137	002616		ROUTD:	JMP	CHANGE	:GO DISPLAY DATA
042	002144	004737	005536		ROUTE:	JSR	PC,ERASE	:GO ERASE SCOPE
043	002150	000137	001732			JMP	LISN	:GO WAIT ON NEXT COMMAND
044	002154	012737	177777	001342	ROUTF:	MOV	#-1,FREERN	:SELECT FREE RUN MODE
045	002162	004737	005506			JSR	PC,NCSTP1	:GO STOP NC11 & CLR CORE MATRIX AREA
046	002166	004737	005440			JSR	PC,NCSTRT	:GO START NC11
047	002172	005000				CLR	RO	:SET UP TIMER
048	002174	013701	001344			MOV	TIME,R1	:SET UP GROSS TIMER VALUE
049	002200	005300		1\$:		DEC	RO	:COUNT
050	002202	001376				BNE	1\$:WAIT FOR ZERO
051	002204	042737	000040	001306		BIC	#BITS,KBUFF	:LOOK FOR POSSIBLE STOP KEY - RID LOWER CASE
052	002212	022737	000123	001306		CMP	#123,KBUFF	:STOP BEEN STRUCK?
053	002220	001007				BNE	2\$:BR IF NOT
054	002222	042777	000003	176746		BIC	#3,ANCCSR	:STOP NC11
055	002230	005077	176742			CLR	ANCCSR	:ZERO NC CSR
056	002234	000137	001712			JMP	LISEN	:GO AWAIT NEXT COMMAND
057	002240	005301		2\$:		DEC	R1	:DO LOOP AGAIN?
058	002242	001356				BNE	1\$:BR IF SO
059	002244	000137	002616			JMP	CHANGE	:NOW GO DISPLAY DATA JUST COLLECTED
060	002250	004737	005536		ROUTG:	JSR	PC,ERASE	:GO ERASE SELECTED DISPLAY
061	002254	000137	001616			JMP	START1	:INITIALIZE AND START FRESH
062	002260	004737	005356		ROUTJ:	JSR	PC,SELCTC	:GO SELECT AR11 FOR JOYSTICK CAL
063	002264	103402				BCS	1\$:BR IF BUS ER FROM AR11
064	002266	000137	004146			JMP	TJOY	:GO TO IT
065	002272	104400	013577	1\$:		TYPE	,MSG7	:GO TYPE 'BUS TIMEOUT ER - AR11'
066	002276	000137	001732			JMP	LISEN	:GO BACK TO KEYBOARD LISEN

857	002302	005037	001346		ROUTK:	CLR	JTYPE	:JTYPE=0 SAYS NC11 FOR JOYSTICK CAL
858	002306	000137	004146			JMP	TJOY	:GO TO IT
859	002312	012737	000340	177776	ROUTL:	MOV	#PR7,PSW	:DON'T WANT ANY INTR'S NOW
870	002320	104400	013331			TYPE	,MSG2	:ASK FOR OCTAL DATA
871	002324	104410				RDOCT		:GO GET IT
872	002326	012637	001312			MOV	(SP)+,THLO	:SAVE IT
873	002332	005037	177776			CLR	PSW	:ENABLE INTR'S
874	002336	000137	002616			JMP	CHANGE	:GO DISPLAY
875	002342	012737	177777	001352	ROUTM:	MOV	#-1,MSELECT	:SELECT 2ND BIT MAP
876	002350	004737	005112			JSR	PC,SELCTA	:GO SELECT VSVO1
877	002354	103407				BCS	2\$:BR IF BUS TIMEOUT ER FROM VSVO1
878	002356	013700	001222			MOV	VTVCSR,RO	:GET 2ND BIT MAP ADRS
879	002362	042760	000400	177760		BIC	#BITS,-20(RO)	:TURN OFF 1ST BIT MAP
880	002370	000137	001732		1\$:	JMP	LISN	:GO AWAIT NEXT COMMAND
881	002374	104400	013412		2\$:	TYPE	,MSG3	:GO TYPE 'BUS TIMEOUT ER - VSVO1 DISPLAY'
882	002400	000773				BR	1\$:GO BACK TO KEYBOARD LISEN
883	002402	004737	005506		ROUTN:	JSR	PC,NCSTP1	:GO STOP NC11 & CLR CORE MATRIX
884	002406	013777	001316	176562		MOV	COMSAV,3NCCSR	:RESTART WITH PREVIOUS CSR CONTENTS
885	002414	000137	001732			JMP	LISN	:COLLECT DATA & AWAIT NEXT KEYBRD COMMAND
886	002420	005037	001320		ROUTR:	CLR	GAIN	:WANT REGULAR GAIN
887	002424	042777	000040	176544		BIC	#40,3NCCSR	:INSURE REGULAR GAIN
888	002432	000137	001732			JMP	LISN	:LOOK FOR NEXT COMMAND
889	002436	042777	000003	176532	ROUTS:	BIC	#3,3NCCSR	:STOP NC
890	002444	005077	176526			CLR	3NCCSR	:ZERO NC CSR
891	002450	000137	001732			JMP	LISN	:LOOK FOR NEXT COMMAND
892	002454	004737	005464		ROUTT:	JSR	PC,NCSTP	:GO STOP THE NC11 IF RUNNING
893	002460	004737	005756			JSR	PC,LDIMGE	:GO SET UP TEST CORE IMAGE
894	002464	000137	002616			JMP	CHANGE	:GO DISPLAY IT
895	002470	012737	000340	177776	ROUTU:	MOV	#PR7,PSW	:DON'T WANT ANY INTR'S
896	002476	104400	013331			TYPE	,MSG2	:ASK FOR OCTAL DATA
897	002502	104410				RDOCT		:GO GET IT
898	002504	012637	001314			MOV	(SP)+,THHI	:SAVE IT
899	002510	005037	177776			CLR	PSW	:ENABLE INTR'S
900	002514	000137	002616			JMP	CHANGE	:GO DISPLAY
901	002520	004737	005262		ROUTV:	JSR	PC,SELCTB	:GO SELECT VTO1
902	002524	103402				BCS	2\$:BR IF TIMEOUT ER FROM VTO1
903	002526	000137	001732		1\$:	JMP	LISN	:GO AWAIT NEXT COMMAND
904	002532	104400	013453		2\$:	TYPE	,MSG4	:GO TYPE 'BUS TIMEOUT ER - VTO1 DISPLAY'
905	002536	000773				BR	1\$:GO BACK TO KEYBOARD LISEN
906	002540	005037	001352		ROUTW:	CLR	MSELECT	:SELECT 1ST BIT MAP
907	002544	004737	005112			JSR	PC,SELCTA	:GO SELECT VSVO1(DEFAULT COND)
908	002550	103407				BCS	2\$:BR IF TIMOUT ER FROM VSVO1
909	002552	013700	001222			MOV	VTVCSR,RO	:GET 1ST BIT MAP ADRS
910	002556	042760	000400	000020		BIC	#BITS,20(RO)	:TURN OFF 2ND BIT MAP
911	002564	000137	001732		1\$:	JMP	LISN	:GO AWAIT NEXT COMMAND
912	002570	104400	013412		2\$:	TYPE	,MSG3	:GO TYPE 'BUS TIMEOUT ER - VSVO1 DISPLAY'
913	002574	000773				BR	1\$:GO BACK TO KEYBOARD LISEN
914	002576	012737	000040	001320	ROUTZ:	MOV	#40,GAIN	:ENABLE GAIN 1
915	002604	053777	001320	176364		BIS	GAIN,3NCCSR	:SET IT AT NC CSR
916	002612	000137	001732			JMP	LISN	:GO AWAIT NEXT COMMAND

```

917          :GO CLEAR SCREEN OF SELECTED DISPLAY
918
919 002616 004737 005536  CHANGE: JSR    PC,ERASE          ;GO ERASE SCOPE
920
921          :STOP NC11 AND GET SET TO DISPLAY
922
923 002622 004737 005464  DISDAT: JSR    PC,NCSTP          ;GO STOP NC11
924
925          :NOW DRAW A BOX AROUND POTENTIAL MATRIX DISPLAY
926          :AREA ON THE SELECTED DISPLAY
927 002626 005737 001254  BOX:   TST     DTYPE          ;WHAT DISPLAY?
928 002632 001067          BNE     BOX1           ;BR IF VTO1
929 002634 012700 001750  MOV     #1750,R0        ;SET UP BIT MAP ADRS - TOP LINE
930 002640 012701 073567  MOV     #73567,R1       ;SET UP PIXEL DATA IN R1 - INT 7
931 002644 004737 005634  JSR    PC,DISPY        ;GO LOAD BIT MAP
932 002650 005200          INC     R0              ;ADVANCE ADRS
933 002652 022700 001770  CMP     #1770,R0        ;TOP LINE DONE?
934 002656 001403          BEQ    2$              ;BR IF SO
935 002660 004737 005646  JSR    PC,DCONT        ;LOAD NEXT PIXEL WD
936 002664 000771          BR     1$              ;NEXT MAP LOAD
937 002666 012700 006010  2$:   MOV     #6010,R0    ;SET UP BIT MAP ADRS - BOT LINE
938 002672 004737 005634  JSR    PC,DISPY        ;GO LOAD BIT MAP
939 002676 005200          INC     R0              ;ADVANCE ADRS
940 002700 022700 006030  3$:   CMP     #6030,R0    ;BOT LINE DONE?
941 002704 001403          BEQ    4$              ;BR IF SO
942 002706 004737 005646  JSR    PC,DCONT        ;LOAD NEXT PIXEL WD
943 002712 000771          BR     3$              ;NEXT MAP LOAD
944 002714 012700 001730  4$:   MOV     #1730,R0    ;SET UP BIT MAP ADRS SIDE LINES
945 002720 062700 000017  5$:   ADD     #17,R0        ;OFFSET NEXT ROW
946 002724 012701 070000  MOV     #70000,R1       ;SET UP PIXEL 3 DATA IN R1 - INT 7
947 002730 022700 006047  CMP     #6047,R0        ;AT BOTTOM OF SCREEN?
948 002734 001411          BEQ    6$              ;GO START VSVOI DISPLAY
949 002736 004737 005634  JSR    PC,DISPY        ;GO LOAD BIT MAP
950 002742 012701 000007  MOV     #7,R1           ;SET UP PIXEL 0 DATA IN R1 - INT 7
951 002746 062700 000021  ADD     #21,R0          ;OFFSET TO RIGHT LINE
952 002752 004737 005634  JSR    PC,DISPY        ;GO LOAD BIT MAP
953 002756 000760          BR     5$              ;DO NEXT ROW
954 002760 013777 001262  6$:   MOV     VTVSAV, JTVCSR ;START DISPLAY
955 002766 012737 002010  7$:   MOV     #2010,MAPADR ;OFFSET BIT MAP ADRS
956 002774 012737 000003  8$:   MOV     #3,PIXCNT    ;SET UP PIXEL BYTE COUNT
957 003002 005037 001272  CLR    PIXASM          ;CLR PIXEL ASSEMBLY WORD
958 003006 000137 003132  JMP    LARGES          ;GO DISPLAY CELL DATA
959
960 003012 012700 003777  BOX1: MOV     #3777,R0        ;SET UP X
961 003016 012701 003636  MOV     #3636,R1        ;SET UP Y
962 003022 004737 005676  1$:   JSR    PC,DISPY1     ;DISPLAY POINT
963 003026 005300          DEC     R0              ;MOVE X BY 1
964 003030 100374          BPL    1$              ;AGAIN TILL DONE
965 003032 005200          INC     R0              ;X=0,Y=3636
966 003034 004737 005676  2$:   JSR    PC,DISPY1     ;DISPLAY POINT
967 003040 005301          DEC     R1              ;MOVE Y BY 1
968 003042 022701 000635  CMP     #635,R1        ;AT BOTTOM LEFT?
969 003046 001372          BNE    2$              ;BR IF NOT
970 003050 005201          INC     R1              ;Y=636,X=0
971 003052 004737 005676  3$:   JSR    PC,DISPY1     ;DISPLAY POINT
972 003056 005200          INC     R0              ;MOV X BY 1

```

```

973 003060 022700 004000      CMP      #4000,R0      ;AT BOTTOM RIGHT?
974 003064 001372      BNE      3$          ;BR IF NOT
975 003066 005300      DEC      R0          ;X=3777,Y=636
976 003070 004737 005676      4$: JSR      PC,DISPY1 ;DISPLAY POINT
977 003074 005201      INC      R1          ;MOV Y BY 1
978 003076 022701 003636      CMP      #3636,R1   ;AT TOP RIGHT?
979 003102 001372      BNE      4$          ;BR IF NOT
980 003104 012737 000100 001274      MOV      #100,HORIZ ;# OF HORIZONTAL POINTS PER ROW
981 003112 012737 000100 001276      MOV      #100,VERT  ;# OF ROWS
982 003120 005037 001300      CLR      XREF        ;START AT LEFT MARGIN
983 003124 012737 003606 001302      MOV      #1926.,YREF ;AND TOP OF SCREEN
984
985                                     ;NOW LETS FIND THE LARGEST CELL
986 003132 013737 001322 001324      LARGES: MOV TOTSIZ,CHARCT ;NUMBER OF ELEMENTS TO CONSIDER
987 003140 017700 176174      MOV      @TABLEX,R0 ;THIS IS FIRST GUESS
988 003144 013701 001340      MOV      TABLEX,R1 ;R1 POINTS TO TABLE
989 003150 020021      LARGEL: CMP R0,(R1)+ ;COMPARE GUESS AGAINST NEW
990 003152 103005      BHS     GSG          ;GUESS STILL GOOD
991 003154 024137 001314      CMP     -(R1),THHI  ;GUESS SMALLER BUT CHECK NEW
992 003160 101001      BHI     OVTHHI      ;HI AGAINST UPPER THRESHOLD
993 003162 011100      MOV     (R1),R0     ;WITHIN BOUNDS. MAKE THIS NEW HI
994 003164 005721      OVTHHI: TST (R1)+  ;INCREASE REGISTER BY 2
995 003166 005337 001324      GSG:    DEC CHARCT  ;COUNT THIS LAST COMPARISON
996 003172 001366      BNE     LARGEL
997
998                                     ;THE LARGEST CELL WITHIN THE UPPER THRESHOLD IS NOW IN R0
999                                     ;NOW ACCOUNT FOR LOWER THRESHOLD - BOW OUT IF TOO SMALL
1000 003174 163700 001312      SUB     THLO,R0     ;SUBTRACT LOWER THRESHOLD
1001 003200 101002      BHI     1$          ;BR IF CELL VALUE GREATER THAN LO THRESHOLD
1002 003202 000137 003500      JMP     DISDN       ;DON'T DISPLAY-ALL VALUES BELOW LO THRESHOLD
1003 003206 010037 001334      1$:    MOV     R0,CPERL ;SAVE LARGEST CELL OF MATRIX
1004
1005                                     ;NOW PICK OUT EACH VALUE IN CORE MATRIX AND SCALE TO THE
1006                                     ;PROPER INTENSITY LEVEL. THEN DISPLAY IT ON THE SELECTED DISPLAY
1007
1008 003212 013737 001340 001264      DMATRIX: MOV TABLEX,MRXADR ;BEGIN AT TOP LEFT ROW
1009 003220 062737 017600 001264      ADD     #8064.,MRXADR ;OFFSET TO BOTTOM OF CORE MATRIX
1010 003226 012737 000100 001336      MOV     #64.,ROWCNT ;THERE ARE 64 CELLS PER ROW
1011 003234 017702 176024      DISLOP: MOV @MRXADR,R2 ;GET A CELL VALUE FROM MATRIX
1012 003240 062737 000002 001264      ADD     #2,MRXADR   ;BUMP MATRIX ADRS
1013 003246 005337 001336      DEC     ROWCNT      ;COUNT CELL THIS ROW
1014 003252 001006      BNE     CKVALU     ;BR IF ROW NOT FINISHED
1015 003254 012737 000100 001336      MOV     #64.,ROWCNT ;RESET NEXT ROW COUNT
1016 003262 162737 000400 001264      SUB     #256.,MRXADR ;SET UP FOR NEXT ROW IN MATRIX
1017 003270 020237 001314      CKVALU: CMP R2,THHI ;CELL WITHIN HI THRESHOLD?
1018 003274 101003      BHI     OUTLIM     ;BR IF NOT
1019 003276 163702 001312      SUB     THLO,R2    ;SUB LOW THRESHOLD
1020 003302 101006      BHI     SCLCEL     ;BR IF CELL ABOVE LOW THRESHOLD
1021 003304 005737 001254      OUTLIM: TST DTYPE  ;WHAT DISPLAY
1022 003310 001052      BNE     CHDN       ;BR IF VTO1
1023 003312 005004      CLR     R4         ;SET CELL TO LOWEST INTENSITY
1024 003314 000137 003504      JMP     MAPLD      ;LOAD BIT MAP
1025 003320 013701 001334      SCLCEL: MOV CPERL,R1 ;NOW ESTABLISH WHAT INTENSITY LEVEL
1026 003324 012703 000006      MOV     #6,R3      ;SCALE TO 32 LEVELS
1027 003330 005737 001254      TST     DTYPE      ;IS IT A VSVO1 DISPLAY?
1028 003334 001001      BNE     1$         ;BR IF NOT
    
```

```

1029 003336 005303          DEC      R3          ;SCALE TO 16 LEVELS ON VSVO1 DISPLAY
1030 003340 005004          1$: CLR      R4          ;LEVEL BUILDS IN R4
1031 003342 006304          DVLOOP: ASL   R4          ;MUL BY 2
1032 003344 020201          CMP     R2,R1         ;COMPARE THIS CELL TO LARGEST (DIVIDED BY 2)
1033 003346 103404          BLO    NODEF         ;BR IF SMALLER
1034 003350 005701          TST    R1            ;CHECK CASE WHERE 0 DIVISOR
1035 003352 001401          BEQ    1$            ;BR IF 0
1036 003354 005204          INC    R4            ;ACCOUNT FOR GOOD SUBTRACTION
1037 003356 160102          1$: SUB    R1,R2         ;NOW DO THE SUBTRACTION
1038 003360 000241          NODEF: CLC          ;
1039 003362 006001          ROR    R1            ;MAKE DIVISOR SMALLER
1040 003364 005303          DEC    R3            ;COUNT POSITION
1041 003366 001365          BNE    DVLOOP        ;AGAIN IF NOT SCALED TO 16 OR 32 LEVELS YET
1042 003370 160102          SUB    R1,R2         ;ACCOUNT FOR REMAINDER
1043 003372 101401          BLOS   GOON          ;BR IF TOO SMALL
1044 003374 005204          INC    R4            ;ROUND UP
1045 003376 005737 001254          GOON: TST   DTYPE       ;WHAT DISPLAY?
1046 003402 001440          BEQ    MAPLD         ;IF VSVO1 GO LOAD MAP
1047 003404 006304          ASL   R4            ;MAKE LEVEL A WORD POINTER
1048 003406 016405 011364          MOV    LEVTAB-2(R4),R5 ;POINTER TO CORRECT LEVEL
1049 003412 006204          ASR   R4            ;RESTORE ACTUAL LEVEL
1050 003414 001410          BEQ    CHDN          ;IF 0 - DON'T DISPLAY
1051
1052          ;THIS CODE DISPLAYS EACH SCALED CELL ON THE VTO1 (ONE OF 32 LEVELS)
1053
1054 003416 013700 001300          MLOOP: MOV   XREF,R0         ;GET READY TO DISPLAY
1055 003422 013701 001302          MOV   YREF,R1         ;THESE ARE THE CORNER COORD.
1056 003426 004737 006212          JSR   PC,GET          ;GET A POINT AND DISPLAY IT
1057 003432 005304          DEC   R4              ;DO AS MANY POINTS AS THE LEVEL
1058 003434 001370          BNE   MLOOP
1059 003436 062737 000040 001300          CHDN: ADD  #32.,XREF,^  ;SHIFT TO NEXT POSSIBLE DATUM
1060 003444 005337 001274          DEC   HORIZ          ;DONE WITH LINE?
1061 003450 001271          BNE   DISLOP         ;NO
1062 003452 012737 000100 001274          MOV   #100,HORIZ     ;RESET LINE COUNTER
1063 003460 005037 001300          CLR   XREF           ;DO CR
1064 003464 162737 000030 001302          SUB   #24.,YREF      ;DO LF
1065 003472 005337 001276          DEC   VERT           ;DONE PAGE?
1066 003476 001256          BNE   DISLOP         ;NO. GO BACK FOR MORE
1067 003500 000137 003662          DISDON: JMP  PATWRT   ;YES, GO DISPLAY PRAMETERS
1068
1069          ;THIS CODE DISPLAYS EACH SCALED CELL ON THE VSVO1 (ONE OF 16 LEVELS)
1070
1071 003504 013700 001266          MAPLD: MOV   MAPADR,R0  ;SET UP BIT MAP ADRS
1072 003510 005704          TST   R4              ;LOOK FOR NO INTENSITY
1073 003512 001401          BEQ   1$              ;BR IF NO INTENSITY
1074 003514 005304          DEC   R4              ;OFFSET TO 0-17
1075 003516 000304          1$: SWAB  R4           ;PREPARE FOR PIXEL LOC
1076 003520 006304          ASL   R4              ;MOVE TO TOP 4 BITS
1077 003522 006304          ASL   R4
1078 003524 006304          ASL   R4
1079 003526 006304          ASL   R4
1080 003530 000241          CLC          ;NOW ASSEMBLE THIS PIXEL INTO PIXEL WORD
1081 003532 006037 001272          ROR   PIXASM         ;NOW MAKE ROOM IN PIXEL WORD
1082 003536 006037 001272          ROR   PIXASM
1083 003542 006037 001272          ROR   PIXASM
1084 003546 006037 001272          ROR   PIXASM

```



```

1085 003552 060437 001272      ADD      R4,PIXASM      ;ADD THIS PIXEL TO OTHERS
1086 003556 005737 001270      TST      PIXCNT        ;ALL 4 PIXELS DONE FOR THIS WORD?
1087 003562 001004                BNE      2$            ;BR IF NOT
1088 003564 013701 001272      MOV      PIXASM,R1     ;LD PIXEL WORD INTO R1
1089 003570 004737 005634      JSR      PC,DISPY     ;LOAD BIT MAP
1090 003574 005337 001270      2$:     DEC      PIXCNT        ;COUNT PIXEL
1091 003600 100215                SPL      DISLOP       ;BR IF 4 PIXELS NOT ASSEMBLED YET
1092 003602 005037 001272      CLR      PIXASM       ;CLR PIXEL ASSEMBLY WORD
1093 003606 012737 000003 001270  MOV      #3,PIXCNT     ;RESET PIXEL COUNT
1094 003614 005287 001266      INC      MAPADR       ;ADVANCE MAP ADRS
1095 003620 013100 001266      MOV      MAPADR,RO    ;LOAD INTO RO
1096 003624 022700 005770      CMP      #5770,RO     ;HAVE ALL 64 ROWS BEEN DONE?
1097 003630 001002                BNE      3$            ;BR IF NOT
1098 003632 000137 003662      JMP      PATWRT       ;NOW GO DISPLAY PARAMETERS
1099 003636 042700 177740      3$:     BIC      #177740,RO   ;SAVE ROW POSITION BITS
1100 003642 022700 000030      CMP      #30,RO      ;NOW LOOK FOR END OF ROW
1101 003646 001003                BNE      4$            ;BR IF NOT AT END
1102 003650 062737 000020 001266  ADD      #20,MAPADR   ;ADVANCE MAP ADRS TO NEXT ROW
1103 003656 000137 003234      4$:     JMP      ,DISLOP     ;GO GET NEXT MATRIX DATUM
1104
1105      ;CODE TO WRITE PARAMETER DATA AT BOTTOM OF SCREEN
1106
1107 003662 005737 001254      PATWRT: TST      DTYPE?   ;WHAT DISPLAY?
1108 003666 100404                BMI      1$            ;BR IF VTO1
1109 003670 012777 012000 175322  MOV      #12000,AVTVPOS ;VSV01 - SET CHAR POS,LINE 20, LEFT MARGIN
1110 003676 000414                BR       2$            ;START 1ST MSG
1111 003700 012737 000006 001332  1$:     MOV      #6,INCXY    ;SET CHARACTER SIZE
1112 003706 012777 000014 175320  MOV      #14,AVTCSR   ;SET MODE
1113 003714 012737 000000 001326  MOV      #0,BASX     ;SET TO LEFT HAND MARGIN
1114 003722 012737 000600 001330  MOV      #384,BASY   ;AND Y LEVEL
1115 003730 004737 006246      2$:     JSR      PC,VTWRIT   ;"CR, LOWER THRESHOLD"
1116 003734 012506                MESS1
1117 003736 013737 001312 006724  MOV      THLO,DUMMY1
1118 003744 004737 006700      JSR      PC,AWRIT
1119 003750 004737 006246      JSR      PC,VTWRIT
1120 003754 012512                MESS2
1121 003756 013737 001314 006724  MOV      THHI,DUMMY1 ;"CR. UPPER THRESHOLD"
1122 003764 004737 006700      JSR      PC,AWRIT
1123 003770 004737 006246      JSR      PC,VTWRIT
1124 003774 012516                MESS3
1125 003776 017737 175204 006724  MOV      ANZLO,DUMMY1 ;"Z COUNT ="
1126 004004 017737 175174 006726  MOV      ANZHI,DUMMY2
1127 004012 004737 006704      JSR      PC,OTTY
1128 004016 004737 006246      JSR      PC,VTWRIT
1129 004022 012522                MESS4
1130 004024 013737 001322 001246  MOV      TOTSIZ,TEMPO ;"MATRIX COUNT="
1131 004032 005002                CLR      R2
1132 004034 005003                CLR      R3
1133 004036 013701 001340      MOV      TABLEX,R1
1134 004042 062103                MXSML:  ADD      (R1)+,R3   ;NOW ADD UP ALL VALUES IN MATRIX
1135 004044 005502                ADC      R2
1136 004046 005337 001246      DEC      TEMPO
1137 004052 001373                BNE      MXSML
1138 004054 010337 006724      MOV      R3,DUMMY1
1139 004060 010237 006726      MOV      R2,DUMMY2
1140 004064 004737 006704      JSR      PC,OTTY

```

```

1141 004070 005737 001320          TST GAIN
1142 004074 001403          BEQ 1$
1143 004076 004737 006246)        JSR PC,VTWRIT
1144 004102 012526          MESS5          ;"ZOOM"
1145 004104 023727 001340 040000 1$:  CMP TABLEX,#MATRIX+20000 ;ISOTOPE B?
1146 004112 001003          BNE 2$        ;NO
1147 004114 004737 006246          JSR PC,VTWRIT ;YES
1148 004120 012532          MESS6          ;"B-GAMMA"
1149 004122 005737 001342          2$:  TST     FREERN ;IN FREE RUN MODE?
1150 004126 001402          BEQ     3$    ;BR IF NOT
1151 004130 000137 002154          JMP     ROUTF ;YES, GO GET NEW DATA
1152 004134 013777 001316 175034 3$:  MOV COMSAV,#NCCSR ;RESUME THE NC11
1153 004142 000137 001732          JMP     LISN  ;DONE WITH MESSAGES
1154
1155
1156          ;THIS CODE DRAWS A BUG(VT01) OR CROSS HAIRS(VSVD1) WITH THE X & Y DATA
1157          ;FROM THE NC11(TYPE 'K' OR DEFAULT) OR THE AR11(TYPE 'J') - THE BUG,
1158          ;ETC. WILL FOLLOW THE JOYSTICK WHEN THE INTERRUPT BAR IS NOT DEPRESSED -
1159          ;ALL PARTS WITHIN THE DISPLAYED BOX ON THE SELECTED DISPLAY SHOULD
1160          ;BE ACCESSIBLE IN A UNIFORM MANNER
1161          ;A CNTRL 'C' WILL GET USER BACK TO KEYBOARD MONITOR
1162 004146 004737 005536          TJOY: JSR     PC,ERASE ;START FRESH
1163 004152 005737 001254          TST     DTYP   ;WHAT DISPLAY?
1164 004156 001053          BNE     JBOX1  ;BR IF VT01
1165
1166          ;DRAW A BOX UP ON VSVD1 SCREEN FOR NC11 OR AR11 JOYSTICK CALIBRATION
1167 004160 005000          JBOX: CLR     RO   ;SET UP BIT MAP ADRS - TOP LINE
1168 004162 012701 073567          MOV     #73567,R1 ;SET UP PIXEL DATA IN R1 - INT 7
1169 004166 004737 005634          JSR     PC,DISPY ;GO LOAD BIT MAP
1170 004172 005200          1$:  INC     RO   ;ADVANCE BIT MAP ADRS
1171 004174 022700 000040          CMP     #40,RO  ;TOP LINE DONE?
1172 004200 001403          BEQ     2$    ;BR IF SO
1173 004202 004737 005646          JSR     PC,DCONT ;LOAD NEXT PIXEL
1174 004206 000771          BR     1$    ;NEXT MAP LOAD
1175 004210 012700 007740          2$:  MOV     #7740,RO ;SET UP BIT MAP ADRS - BOT LINE
1176 004214 004737 005634          JSR     PC,DISPY ;GO LOAD BIT MAP
1177 004220 005200          3$:  INC     RO   ;ADVANCE ADRS
1178 004222 022700 010000          CMP     #10000,RO ;BOT LINE DONE?
1179 004226 001403          BEQ     4$    ;BR IF SO
1180 004230 004737 005646          JSR     PC,DCONT ;LOAD NEXT PIXEL WORD
1181 004234 000771          BR     3$    ;NEXT MAP LOAD
1182 004236 012700 000040          4$:  MOV     #40,RO  ;SET UP BIT MAP ADRS SIDE LINES
1183 004242 012701 000007          5$:  MOV     #7,R1   ;SET UP PIXEL 0 DATA IN R1 - INT 7
1184 004246 004737 005634          JSR     PC,DISPY ;GO LOAD BIT MAP
1185 004252 012701 070000          MOV     #70000,R1 ;SET UP PIXEL 3 DATA IN R1 - INT 7
1186 004256 062700 000037          ADD     #37,RO  ;OFFSET TO RIGHT SIDE LINE
1187 004262 004737 005634          JSR     PC,DISPY ;GO LOAD BIT MAP
1188 004266 005200          INC     RO   ;GO TO NEXT ROW ON LEFT
1189 004270 022700 007740          CMP     #7740,RO ;TO BOTTOM YET?
1190 004274 001362          BNE     5$    ;BR IF NOT
1191 004276 013777 001262 174716          MOV     VTVSAV,#VTVCSR ;ENABLE BIT MAP
1192 004304 000466          BR     DISBG  ;CONTINUE TO ANALOG DATA
1193
1194          ;DRAW BOX UP ON VT01 SCREEN - DIFF SIZE FOR AR11 JOYSTICK CALIBRATION
1195 004306 005737 001346          JBOX1: TST     JTYPE ;AR11 ANALOG SOURCE?
1196 004312 100432          BMI     JBOX2  ;BR IF SO
    
```



```

004544 005137 001354          COM      BELLEN          :YES, DO IT
004550 005737 001354      10$:    TST      BELLEN          :SOUND BELL ON NO CONVERT?
004554 001355          BNE      IS          :BR IF BELL NOT DESIRED
004556 012737 000207 001310    MOV      #207,TTYOUT :SET UP BELL CODE
004564 004737 005174          JSR      PC,TYPE    :GO ZING BELL
004570 000742          BR      IS          :GO TRY AGAIN
004572 017737 174404 001246    3$:    MOV      @INCAREG,TEMPO :STORE CONVERSION VALUES
004600 042737 100200 001246    BIC      #100200,TEMPO :MASK OFF BITS 7,15
004606 004737 007210          JSR      PC,XYAVE   :GO AVG LAST 32, X-Y JOYSTICK VALUES
004612 012777 000010 174372    MOV      #10,@JNCSFUN :CLEAR JOYSTICK DEPRESS FLAG
004620 005777 174352          TST      @JNCCSR   :LOOK FOR BUTTON DOWN
004624 100004          BPL      IS          :BR IF NOT
004626 005737 001254          TST      DTYPE     :DON'T MOVE BUG BUT REFRESH
004632 001025          BNE      IS          :REFRESH VTO1 ONLY
004634 000720          BR      IS          :NO NEED ON VSVO1
004636 006337 001246      4$:    ASL      TEMPO     :MUL BY 2
004642 005737 001254          TST      DTYPE     :ALL DONE SCALING IF VSVO1
004646 001425          BEQ      IS          :BR IF SO
004650 005137 001246          COM      TEMPO     :INVERT DATA FOR VTO1
004654 042737 000401 001246    BIC      #401,TEMPO :RID GARBAGE
004662 113700 001246          MOVB    TEMPO,RO   :GET X
004666 113701 001247          MOVB    TEMPO+1,R1 :GET Y
004672 006300          ASL      RO        :VTO1 REQUIRES X & Y TO BE INFLATED BY 16
004674 006300          ASL      RO
004676 006300          ASL      RO
004700 006301          ASL      R1
004702 006301          ASL      R1
004704 006301          ASL      R1
004706 012777 000020 174320    5$:    MOV      #20,@VTCR   :SELECT WRITE THRU MODE
004714 004737 005676          JSR      PC,DISPY1 :GO DISPLAY THE BUG - VTO1
004720 000666          BR      IS          :DO AGAIN
004722 004737 005726      6$:    JSR      PC,DISPY2 :GO DISPLAY CROSS HAIRS - VSVO1
004726 000663          BR      IS          :DO ANOTHER CONVERSION

:THIS CODE LOOKS AT THE AR11 FOR ANALOG DATA
004730          DISBG1:
004730 012737 020000 001350    1$:    MOV      #20000,ARCHAN :SELECT CHAN 0 & UNIPOLAR
004736 004737 006024          JSR      PC,ARCONV  :GO START A/D - CHAN 0
004742 010337 001246          MOV      R3,TEMPO  :SAVE Y
004746 105237 001351          INCB    ARCHAN+1   :SELECT CHAN 1
004752 004737 006024          JSR      PC,ARCONV  :GO START A/D - CHAN 1
004756 010337 001250          MOV      R3,TEMP1  :SAVE X
004762 105237 001351          INCB    ARCHAN+1   :SELECT CHAN 2
004766 004737 006024          JSR      PC,ARCONV  :GO START A/D - CHAN 2
004772 010337 001252          MOV      R3,TEMP2  :READ JOYSTICK BUTTON INDICATOR
004776 000337 001246          SWAB    TEMPO     :PUT Y IN HI BYTE
005002 113737 001250 001246    MOVB    TEMP1,TEMPO :AND X IN LO BYTE
005010 004737 007210          JSR      PC,XYAVE   :GO AVG LAST 32, X-Y JOYSTICK VALUES
005014 105737 001252          TSTB   TEMP2     :LOOK FOR BAR DOWN (ZERO COND)
005020 001004          BNE      IS          :BR IF NOT DEPRESSED
005022 005737 001254          TST      DTYPE     :DON'T MOVE POINT BUT REFRESH
005026 001023          BNE      IS          :REFRESH VTO1
005030 000737          BR      IS          :NO NEED ON VSVO1
005032 005737 001254      2$:    TST      DTYPE     :EACH DISPLAY SCALES DIFFERENTLY
005036 001422          BEQ      IS          :BR IF VSVO1
005040 005137 001246          COM      TEMPO     :INVERT FOR VTO1
    
```

DZNCB.911 INITIALIZE THE COMMON TAGS

11300	00504	113700	001246	MOV8	TEMPO.R0	:GET X
11310	00505	113701	001247	MOV8	TEMPO+1.R1	:GET Y
11320	00506	000600		MOV8		:SCALE X BY 8
11330	00507	000600		MOV8		
11340	00508	000600		MOV8		:SCALE Y BY 8
11350	00509	000600		MOV8		
11360	00510	000600		MOV8		
11370	00511	000600		MOV8		
11380	00512	000600		MOV8		
11390	00513	000600		MOV8		
11400	00514	000600		MOV8		
11410	00515	000600		MOV8		
11420	00516	000600		MOV8		
11430	00517	000600		MOV8		
11440	00518	000600		MOV8		
11450	00519	000600		MOV8		
11460	00520	000600		MOV8		
11470	00521	000600		MOV8		
11480	00522	000600		MOV8		
11490	00523	000600		MOV8		
11500	00524	000600		MOV8		
11510	00525	000600		MOV8		
11520	00526	000600		MOV8		
11530	00527	000600		MOV8		
11540	00528	000600		MOV8		
11550	00529	000600		MOV8		
11560	00530	000600		MOV8		
11570	00531	000600		MOV8		
11580	00532	000600		MOV8		
11590	00533	000600		MOV8		
11600	00534	000600		MOV8		
11610	00535	000600		MOV8		
11620	00536	000600		MOV8		
11630	00537	000600		MOV8		
11640	00538	000600		MOV8		
11650	00539	000600		MOV8		
11660	00540	000600		MOV8		
11670	00541	000600		MOV8		
11680	00542	000600		MOV8		
11690	00543	000600		MOV8		
11700	00544	000600		MOV8		
11710	00545	000600		MOV8		
11720	00546	000600		MOV8		
11730	00547	000600		MOV8		
11740	00548	000600		MOV8		
11750	00549	000600		MOV8		
11760	00550	000600		MOV8		
11770	00551	000600		MOV8		
11780	00552	000600		MOV8		
11790	00553	000600		MOV8		
11800	00554	000600		MOV8		
11810	00555	000600		MOV8		
11820	00556	000600		MOV8		
11830	00557	000600		MOV8		
11840	00558	000600		MOV8		
11850	00559	000600		MOV8		
11860	00560	000600		MOV8		
11870	00561	000600		MOV8		
11880	00562	000600		MOV8		
11890	00563	000600		MOV8		
11900	00564	000600		MOV8		
11910	00565	000600		MOV8		
11920	00566	000600		MOV8		
11930	00567	000600		MOV8		
11940	00568	000600		MOV8		
11950	00569	000600		MOV8		
11960	00570	000600		MOV8		
11970	00571	000600		MOV8		
11980	00572	000600		MOV8		
11990	00573	000600		MOV8		
12000	00574	000600		MOV8		
12010	00575	000600		MOV8		
12020	00576	000600		MOV8		
12030	00577	000600		MOV8		
12040	00578	000600		MOV8		
12050	00579	000600		MOV8		
12060	00580	000600		MOV8		
12070	00581	000600		MOV8		
12080	00582	000600		MOV8		
12090	00583	000600		MOV8		
12100	00584	000600		MOV8		
12110	00585	000600		MOV8		
12120	00586	000600		MOV8		
12130	00587	000600		MOV8		
12140	00588	000600		MOV8		
12150	00589	000600		MOV8		
12160	00590	000600		MOV8		
12170	00591	000600		MOV8		
12180	00592	000600		MOV8		
12190	00593	000600		MOV8		
12200	00594	000600		MOV8		
12210	00595	000600		MOV8		
12220	00596	000600		MOV8		
12230	00597	000600		MOV8		
12240	00598	000600		MOV8		
12250	00599	000600		MOV8		
12260	00600	000600		MOV8		
12270	00601	000600		MOV8		
12280	00602	000600		MOV8		
12290	00603	000600		MOV8		
12300	00604	000600		MOV8		
12310	00605	000600		MOV8		
12320	00606	000600		MOV8		
12330	00607	000600		MOV8		
12340	00608	000600		MOV8		
12350	00609	000600		MOV8		
12360	00610	000600		MOV8		
12370	00611	000600		MOV8		
12380	00612	000600		MOV8		
12390	00613	000600		MOV8		
12400	00614	000600		MOV8		
12410	00615	000600		MOV8		
12420	00616	000600		MOV8		
12430	00617	000600		MOV8		
12440	00618	000600		MOV8		
12450	00619	000600		MOV8		
12460	00620	000600		MOV8		
12470	00621	000600		MOV8		
12480	00622	000600		MOV8		
12490	00623	000600		MOV8		
12500	00624	000600		MOV8		
12510	00625	000600		MOV8		
12520	00626	000600		MOV8		
12530	00627	000600		MOV8		
12540	00628	000600		MOV8		
12550	00629	000600		MOV8		
12560	00630	000600		MOV8		
12570	00631	000600		MOV8		
12580	00632	000600		MOV8		
12590	00633	000600		MOV8		
12600	00634	000600		MOV8		
12610	00635	000600		MOV8		
12620	00636	000600		MOV8		
12630	00637	000600		MOV8		
12640	00638	000600		MOV8		
12650	00639	000600		MOV8		
12660	00640	000600		MOV8		
12670	00641	000600		MOV8		
12680	00642	000600		MOV8		
12690	00643	000600		MOV8		
12700	00644	000600		MOV8		
12710	00645	000600		MOV8		
12720	00646	000600		MOV8		
12730	00647	000600		MOV8		
12740	00648	000600		MOV8		
12750	00649	000600		MOV8		
12760	00650	000600		MOV8		
12770	00651	000600		MOV8		
12780	00652	000600		MOV8		
12790	00653	000600		MOV8		
12800	00654	000600		MOV8		
12810	00655	000600		MOV8		
12820	00656	000600		MOV8		
12830	00657	000600		MOV8		
12840	00658	000600		MOV8		
12850	00659	000600		MOV8		
12860	00660	000600		MOV8		
12870	00661	000600		MOV8		
12880	00662	000600		MOV8		
12890	00663	000600		MOV8		
12900	00664	000600		MOV8		
12910	00665	000600		MOV8		
12920	00666	000600		MOV8		
12930	00667	000600		MOV8		
12940	00668	000600		MOV8		
12950	00669	000600		MOV8		
12960	00670	000600		MOV8		
12970	00671	000600		MOV8		
12980	00672	000600		MOV8		
12990	00673	000600		MOV8		
13000	00674	000600		MOV8		
13010	00675	000600		MOV8		
13020	00676	000600		MOV8		
13030	00677	000600		MOV8		
13040	00678	000600		MOV8		
13050	00679	000600		MOV8		
13060	00680	000600		MOV8		
13070	00681	000600		MOV8		
13080	00682	000600		MOV8		
13090	00683	000600		MOV8		
13100	00684	000600		MOV8		
13110	00685	000600		MOV8		
13120	00686	000600		MOV8		
13130	00687	000600		MOV8		
13140	00688	000600		MOV8		
13150	00689	000600		MOV8		
13160	00690	000600		MOV8		
13170	00691	000600		MOV8		
13180	00692	000600		MOV8		
13190	00693	000600		MOV8		
13200	00694	000600		MOV8		
13210	00695	000600		MOV8		
13220	00696	000600		MOV8		
13230	00697	000600		MOV8		
13240	00698	000600		MOV8		
13250	00699	000600		MOV8		
13260	00700	000600		MOV8		
13270	00701	000600		MOV8		
13280	00702	000600		MOV8		
13290	00703	000600		MOV8		
13300	00704	000600		MOV8		
13310	00705	000600		MOV8		
13320	00706	000600		MOV8		
13330	00707	000600		MOV8		
13340	00708	000600		MOV8		
13350	00709	000600		MOV8		
13360	00710	000600		MOV8		
13370	00711	000600		MOV8		
13380	00712	000600		MOV8		
13390	00713	000600		MOV8		
13400	00714	000600		MOV8		
13410	00715	000600		MOV8		
13420	00716	000600		MOV8		
13430	00717	000600		MOV8		
13440	00718	000600		MOV8		
13450	00719	000600		MOV8		
13460	00720	000600		MOV8		
13470	00721	000600		MOV8		
13480	00722	000600		MOV8		
13490	00723	000600		MOV8		
13500	00724	000600		MOV8		
13510	00725	000600		MOV8		
13520	00726	000600		MOV8		
13530	00727	000600		MOV8		
13540	00728	000600		MOV8		</

.SBTTL PROGRAM SUBROUTINES

:ROUTINE SELECTS VSVO1 DISPLAY - SETS UP BUS ADRS AND INTENSITY LEVEL
:AND INTENSITY LOOK-UP TABLE - THE CARRY BIT IS SET ON EXIT IF THE
:VSVO1 IS NOT SEEN AT THE ASSIGNED BUS ADDRESS

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

SELCOA: MOV VTVADR,RO ;GET VSVO1 BASE ADRS
MOV #VTVCRG,R1 ;GET PTR ADRS
1S: MOV RO,(R1)+ ;SET UP REG ADRS PTRS
ADD #2,RO ;BUMP REG ADRS
CMP #VTVCSR,R1 ;CHAR GEN REGS ALL SET UP?
BNE 1S ;BR IF NOT
MOV VTMADR,RO ;GET BASE ADRS OF BIT MAP
TST MSELCT ;USING SECOND MAP?
BEQ 2S ;BR IF NOT
ADD #20,RO ;POINT TO 2ND BIT MAP ADRS'S
2S: MOV RO,(R1)+ ;CONTINUE TO BIT MAP ADRS'S
ADD #2,RO ;BUMP REG ADRS
CMP #VTCSR,R1 ;ALL SET UP?
BNE 2S ;BR IF NOT
MOV #5S,3#ERRVEC ;SET UP BUS TIMEOUT RETURN ADRS IF NO VSVO1
TST #VTVCRG ;IS CHARACTER GENERATOR THERE?
TST #VTVCSR ;IS BIT MAP THERE?
3S: MOV INTLUT,RO ;SET UP ADRS & DATA OF INTENSITY LOOK-UP TABLE
MOV RO,#VTVINT ;SET UP TABLE
ADD #401,RO ;ADVANCE ADRS & INTENSITY
BIT #10000,RO ;TABLE LOADED?
BEQ 3S ;BR IF NOT
CLR DTYPE ;DTYPE=0 SAYS VSVO1
MOV #16,,INTENS ;MAX 16 INTENSITIES
CLC ;ZERO CARRY SAYS VSVO1 THERE
4S: MOV #ERRVEC,3#ERRVEC ;RESTORE ER TRAP LOC TO PT TO 5
RTS PC ;EXIT
5S: CMP (SP)+,(SP)+ ;FIX STACK SINCE NO RTI
SEC ;CARRY ON EXIT SAYS NO VSVO1
BR 4S ;GO EXIT

:ROUTINE SELECTS VTO1 DISPLAY - SETS UP BUS ADRS AND INTENSITY LEVEL -
:THE CARRY BIT IS SET ON EXIT IF THE VTO1 IS NOT SEEN AT THE
:ASSIGNED BUS ADDRESS

1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377

SELCOB: MOV VTADR,RO ;GET VTO1 BASE ADRS
MOV #VTCSR,R1 ;GET PTR ADRS
1S: MOV RO,(R1)+ ;SET UP VTO1 REG ADRS PTRS
ADD #2,RO ;BUMP REG ADRS
CMP #ARCSR,R1 ;ALL SET UP?
BNE 1S ;BR IF NOT
MOV #3S,3#ERRVEC ;SET UP FOR VTO1 TIMEOUT
MOV #14,#VTCSR ;SELECT STORE MODE
MOV #32,,INTENS ;MAX 32 INTENSITIES
MOV #-1,DTYPE ;DTYPE=-1 SAYS VTO1
CLC ;ZERO CARRY SAYS VTO1 THERE
2S: MOV #ERRVEC+2,3#ERRVEC ;RESTORE LOC 4 PTR TO PT TO 5
RTS PC ;EXIT

E03

```

1378 005350 022626 3$: CMP (SP)+,(SP)+ ;FIX STACK SINCE NO RETURN
1379 005352 000261 SEC ;CARRY SET SAYS NO VTO1 DISPLAY
1380 005354 000771 BR 2$ ;GO EXIT

:*****
:ROUTINE SELECTS AND SETS UP BUS ADRS FOR AR11 JOYSTICK CALIBRATION
:THE CARRY SET ON EXIT IF THE AR11 IS NOT SEEN
:*****
1385 005356 013700 001174 SELCTC: MOV ARADR,RO ;GET AR11 BASE ADRS
1386 005358 012701 001242 MOV #ARCSR,R1 ;GET PTR ADRS
1387 005356 010021 1$: MOV RO,(R1)+ ;SET UP REG ADRS PTRS
1388 005370 062700 ADD #2,RO ;BUMP REG ADRS
1389 005374 022701 001246 CMP #TEMPO,R1 ;ALL SET UP?
1390 005400 001372 BNE 1$ ;BR IF NOT
1391 005402 012737 005434 000004 MOV #3$,J#ERRVEC ;SET UP TIMEOUT RETURN ADRS IF AR11 NOT THERE
1392 005410 005077 173626 CLR JARCSR ;SEE IF THERE
1393 005414 012737 177777 001346 MOV #-1,JTYPE ;JTYPE=-1 SAYS USE AR11 FOR JOYSTICK CAL
1394 005422 000241 CLC ;CARRY ZERO SAYS AR11 IS THERE
1395 005424 012737 000006 000004 2$: MOV #ERRVEC+2,J#ERRVEC ;RESTORE LOC 4 TO PT TO 6
1396 005432 000207 RTS PC ;EXIT
1397 005434 000261 3$: SEC ;CARRY SET SAYS THAT AR11 IS NOT ADRS ASSIGNED
1398 005436 000772 BR 2$ ;GO EXIT

:*****
:ROUTINE STARTS NC11 AT SELECTED GAIN
:*****
1404 005440 012777 001001 173530 NCSTRT: MOV #1001,JNCCSR ;SET UP 64*64 MATRIX AND ADC ENABLE
1405 005446 053777 001320 173522 BIS GAIN,JNCCSR ;SET UP GAIN
1406 005454 052777 000002 173514 BIS #BIT1,JNCCSR ;SET GO BIT
1407 005462 000207 RTS PC ;RETURN

:*****
:ROUTINE STOPS NC11 AND SAVES NC11 STATUS
:*****
1414 005464 017737 173506 001316 NCSTP: MOV JNCCSR,COMSAV ;SAVE THE INTERFACE ACTION
1415 005472 042777 000003 173476 BIC #3,JNCCSR ;DISABLE NPR'S
1416 005500 005077 173472 CLR JNCCSR ;ZERO ALL STATUS
1417 005504 000207 RTS PC ;RETURN

:*****
:ROUTINE STOPS NC11, SAVES STATUS AND CLEARS MATRIX CORE AREA
:*****
1424 005506 004737 005464 NCSTP1: JSR PC,NCSTP ;GO STOP NC11 AND SAVE STATUS
1425 005512 012777 000020 173472 MOV #CLZ,JNCSFUN ;ZERO Z REG COUNT
1426 005520 012700 020000 MOV #MATRIX,RO ;GET SET TO ZERO CORE MATRIX AREA
1427 005524 005020 1$: CLR (RO)+ ;ZERO LOC
1428 005526 020027 060000 CMP RO,#MATRIX+40000 ;ALL DONE?
1429 005532 001374 BNE 1$ ;BR IF MORE
1430 005534 000207 RTS PC ;RETURN

:*****
:ROUTINE WILL ERASE DISPLAY (VSV01 OR VTO1)
:*****
1433 005536 005737 001254 ERASE: TST DTYPE ;WHAT DISPLAY?
1434 005542 100017 BPL 2$ ;BR IF VSV01
1435 005544 012777 000016 173462 MOV #16,JVTCSR ;ERASE DISPLAY

```

```

1434 005552 005037 001246 CLR TEMPO ;SET UP VTO1 ERASE WAIT LOOPS
1435 005556 012737 000002 001250 MOV #2,TEMP1 ;DO LOOP 2 TIMES
1436 005564 005337 001246 1$: DEC TEMPO ;COUNT AWAY
1437 005570 001375 BNE 1$ ;TILL 0
1438 005572 005337 001250 DEC TEMP1 ;2ND LOOP DONE?
1439 005576 001372 BNE 1$ ;BR IF NOT
1440 005600 000414 BR 4$ ;ALL DONE - RETURN
1441 005602 052777 001000 173412 2$: BIS #1000,AVTVCSR ;ERASE DISPLAY (CLR BIT MAP)
1442 005610 105777 173406 3$: TSTB AVTVCSR ;LOOK FOR READY
1443 005614 100375 BPL 3$ ;WAIT FOR IT
1444 005616 042777 001000 173376 BIC #1000,AVTVCSR ;TURN OFF ERASE DISPLAY
1445 005624 012777 002035 173362 MOV #2035,AVTVCRG ;CLR CHAR BUFFER & DISABLE CURSOR
1446 005632 000207 4$: RTS PC ;EXIT

;*****
;ROUTINE WILL LOAD BIT MAP (VSVO1)
;RO CONTAINS BIT MAP ADRS AND R1 THE BIT MAP DATA
;*****
1447 005634 042777 000400 173360 DISPY: BIC #400,AVTVCSR ;STOP DISPLAY
1448 005642 010077 173356 MOV RO,AVTVMAP ;LOAD BIT MAP ADRS
1449 005646 042777 000400 173346 DCONT: BIC #400,AVTVCSR ;AGAIN IF ENTERING HERE
1450 005654 105777 173342 1$: TSTB AVTVCSR ;READY?
1451 005660 100375 BPL 1$ ;WAIT THEN
1452 005662 010177 173340 MOV R1,AVTVPX ;LOAD BIT MAP
1453 005666 052777 000400 173326 BIS #400,AVTVCSR ;RESUME DISPLAY
1454 005674 000207 RTS PC ;RETURN

;*****
;ROUTINE WILL DISPLAY A POINT (VTO1)
;*****
1455 005676 105777 173332 DISPY1: TSTB AVTVCSR ;DISPLAY READY?
1456 005702 100375 BPL DISPY1 ;WAIT FOR IT
1457 005704 042700 174000 BIC #174000,RO ;RID GARBAGE
1458 005710 042701 174000 BIC #174000,R1 ;RID GARBAGE
1459 005714 010077 173316 MOV RO,AVTXDAC ;SET UP X DAC
1460 005720 010177 173314 MOV R1,AVTYDAC ;SET UP Y DAC
1461 005724 000207 RTS PC ;EXIT

;*****
;ROUTINE WILL DISPLAY X & Y CROSS HAIRS ON VSVO1
;*****
1462 005726 005777 173262 DISPY2: TST AVTVCRG ;READY?
1463 005732 100375 BPL DISPY2 ;WAIT IF NOT
1464 005734 105137 001246 COMB TEMPO ;X NEEDS TO BE INVERTED
1465 005740 013777 001246 173250 MOV TEMPO,AVTVCHP ;LOAD X & Y CROSS HAIRS
1466 005746 052777 016000 173240 BIS #16000,AVTVCRG ;ENABLE THE CROSS HAIRS
1467 005754 000207 RTS PC ;RETURN

;*****
;ROUTINE WILL FILL CORE WITH AN IMAGE THAT WHEN
;DISPLAYED WILL CONTAIN ALL THE INTENSITY LEVELS -
;ROWS AT THE BOTTOM OF THE SCREEN WILL APPEAR BRIGHTEST -
;NOTE THAT THIS IS ONLY A DISPLAY TEST PATTERN FOR
;POSSIBLE DISPLAY ADJUSTMENTS BY THE USER
;*****
1468 005756 012700 000100 LDIMGE: MOV #100,RO ;COUNT 64 ROWS

```



```

1490 005762 013701 001340      MOV      TABLEX,R1      ;GET SELECTED ISOTOPE
1491 005766 012702 000100      MOV      #100,R2       ;COUNT 64 DATA POINTS PER ROW
1492 005772 012703 000100      MOV      #100,R3       ;100 WILL REPRESENT HIGHEST INTENSITY LEVEL(100-0)
1493 005776 010321 000000      15:     MOV      R3,(R1)+   ;LOAD CORE IMAGE
1494 006200 005302 000000      DEC      R2            ;DCNE ROW?
1495 006002 001375 000000      BNE      15           ;BR IF NOT
1496 006004 005300 000000      DEC      R0            ;DONE ROWS?
1497 006006 001405 000000      BEQ      25           ;BR IF SO
1498 006010 162703 000001      SUB      #1,R3         ;LOWER NEXT CELL VALUE
1499 006014 012702 000100      MOV      #100,R2       ;RESET ROW LENGTH COUNTER
1500 006020 000766 000000      BR       15           ;LOAD THIS ROW IMAGE
1501 006022 000207 000000      25:     RTS      PC        ;RETURN FOR DISPLAY

```

```

:*****
:ROUTINE WILL DO A CONVERSION AT CHAN # IN 'ARCHAN'
:EXIT WITH CONVERSION VALUE IN R3
:*****

```

```

1507 006024 005777 173214      ARCONV: TST      @ARBUF      ;CLR DONE FLAG
1508 006030 013777 001350      MOV      ARCHAN,@ARCSR ;LD CHAN & UNIPOLAR BITS
1509 006036 052777 000001      BIS      #1,@ARCSR     ;START A/D
1510 006044 105777 173172      15:     TSTB     @ARCSR      ;LOOK FOR DONE
1511 006050 100375 000000      BPL      15           ;WAIT FOR IT
1512 006052 017703 173166      MOV      @ARBUF,R3     ;GET VALUE
1513 006056 162703 000500      SUB      #500,R3       ;OFFSET VALUE
1514 006062 032703 177400      BIT      #177400,R3    ;SEE IF OUT OF RANGE
1515 006066 100402 000000      BMT      25           ;BR IF TOO SMALL
1516 006070 001003 000000      BNE      35           ;BR IF TOO LARGE
1517 006072 000404 000000      BR       45           ;NORMAL EXIT
1518 006074 005003 000000      25:     CLR      R2            ;LIMIT TO 0
1519 006076 000402 000000      BR       45           ;GO EXIT
1520 006100 012703 000377      35:     MOV      #377,R3     ;LIMIT TO 377
1521 006104 000207 000000      45:     RTS      PC        ;EXIT

```

```

:*****
:ROUTINE WILL DISPLAY A CHARACTER (VSVO!)
:*****

```

```

1526 006106 005777 173102      DCHAR:  TST      @VTVCRG  ;READY?
1527 006112 100375 000000      BPL      DCHAR        ;WAIT FOR IT
1528 006114 110377 173074      MOVB     R3,@VTVCRG   ;LOAD CHAR
1529 006120 000207 000000      RTS      PC           ;EXIT

```

```

:*****
:KEYBOARD INTERRUPT SERVICE ROUTINE
:*****

```

```

1534 006122 017737 173020 001306      KBINT:  MOV      @STKB,KBUFF  ;READ KEY BOARD
1535 006130 042737 000200 001306      BIC      #200,KBUFF    ;RID PARITY
1536 006136 022737 000003 001306      CMP      #3,KBUFF     ;CNTRL 'C'?
1537 006144 001002 000000      BNE      15           ;BR IF NOT
1538 006146 000137 001712      JMP      LISEN        ;ABORT WHATEVER & LOOK FOR NEXT COMMAND
1539 006152 000002 000000      15:     RTI

```

```

:*****
:ROUTINE TYPES 'CR' AND 'LF' OR CHAR IN TTYOUT
:*****

```

```

1544 006154 012737 000015 001310      TYPCR:  MOV      #15,TTYOUT ;SET UP FOR A 'CR'
1545 006162 004737 006174      JSR     PC,TYPO

```

```

1546 006166 012737 000012 001310      MOV #12,TTYOUT      ;SET UP FOR LF
1547 006174 105777 172750      TSTB %STPS        ;WAIT FOR LAST CHARACTER
1548 006200 100375      BPL TYP0
1549 006202 013777 001310 172742      MOV TTYOUT,%STPB  ;SEND IT OUT
1550 006210 000207      RTS PC

:*****
:GET IS A SUBROUTINE TO GET AN X,Y COORDINATE FROM THE
:CODED LEVEL MATRIX,DISPLAY IT AND EXIT.
:*****
1555 006212 111502      GET:  MOVB (R5),R2   ;MOVE A BYTE OF X AND Y
1556 006214 042702 177707      BIC #177707,R2    ;MASK ALL BUT X
1557 006220 006202      ASR R2            ;ABS VALUE IS 4*X
1558 006222 060200      ADD R2,R0        ;ADD TO CORNER VALUE
1559 006224 112502      MOVB (R5)+,R2    ;MOV SAME BYTE AND POKE POINTER
1560 006226 042702 177770      BIC #177770,R2    ;MASK ALL BUT Y
1561 006232 060201      ADD R2,R1        ;YABS Y INC IS 3
1562 006234 006302      ASL R2           ;SO DO Y=Y*3
1563 006236 060201      ADD R2,R1        ;ADD TO CORNER VALUE
1564 006240 004737 005676      JSR PC,DISPY1    ;GO DISPLAY POINT
1565 006244 000207      RTS PC           ;RETURN TO MAIN

:*****
:THIS SUBROUTINE IS CALLED WITH THE ADDRESS OF A
:MESSAGE TABLE FOLLOWING THE CALL. THE MESSAGE
:TABLE CONTAINS A LIST OF POINTERS TO PHRASES.
: EACH PHRASE IS TERMINATED WITH A 200 BYTE. THE TABLE
: IS TERMINATED WITH A 0.
:*****
1575 006246 017605 000000      VTWRIT: MOV %2(SP),R5 ;GET ADDRESS OF MESSAGE IN R5
1576 006252 062716 000002      ADD #2,(SP)      ;ADJUST RETURN
1577 006256 005715      VTPL:  TST (R5)    ;LOOK FOR ZERO AS TERMINATOR
1578 006260 001476      BEQ VTWDON      ;BR IF MSG DONE
1579 006262 012504      MOV (R5)+,R4    ;ADDRESS OF PHRASE TO R4
1580 006264 112403      VTWL:  MOVB (R4)+,R3 ;THIS IS LETTER TO BE DISPLAYED
1581 006266 100773      BMI VTPL        ;200 IS THE TERMINATOR
1582 006270 005737 001254      TST DTYPE      ;WHAT DISPLAY?
1583 006274 001003      BNE DVTO1      ;BR IF VTO1
1584 006276 004737 006106      JSR PC,DCHAR   ;GO DISPLAY CHAR VSVD1
1585 006302 000770      BR VTWL        ;GO LOOK AT NEXT CHAR
1586 006304 022703 000015      DVTO1: CMP #15,R3 ;CR?
1587 006310 001004      BNE %1         ;BR IF NOT
1588 006312 012737 000000 001326      MOV #0,BASX    ;RESET MARGIN
1589 006320 000761      BR VTWL        ;GO GET NEXT CHAR
1590 006322 022703 000012      %1:  CMP #12,R3  ;LF?
1591 006326 001004      BNE VTEXCP    ;BR IF NOT
1592 006330 162737 000060 001330      SUB #60,BASY   ;GO TO NEXT LINE
1593 006336 000752      BR VTWL        ;GO GET NEXT CHAR
1594 006340 110337 001304      VTEXCP: MOV R3,KDBBUF ;THERE ARE A FEW CHARACTERS
1595 006344 012737 000034 006460      MOV #34,TEMCHR ;THAT REQUIRE CONVERSION
1596 006352 004737 006462      JSR PC,BRAN    ;BEFORE DISPLAY.
1597 006356 006373      VTEXCT-1
1598 006360 006430      CH74          ;CHARACTER IS <

```

```

1600 006362 006422          CH75          ;=
1601 006364 006416          CH72          ;:
1602 006366 006410          CH76          ;>
1603 006370 006402          CH77          ;?
1604 006372 000420          BR VTNOSP      ;CHARACTER IS OK AS IS
1605 006374      074      075      072      VTEXCT: .BYTE 74,75,72,76,77,0
1606 006377      076      077      000
1607
1608          .EVEN
1609 006402 062737 000004 006460 CH77:  ADD #4,TEMCHR      ;CONVERT OCTAL TO 42
1610 006410 062737 000007 006460 CH76:  ADD #7,TEMCHR      ;CONVERT TO 46
1611 006416 005237 006460 CH72:  INC TEMCHR        ;CONVERT TO 37
1612 006422 062737 000002 006460 CH75:  ADD #2,TEMCHR      ;CONVERT TO 36
1613 006430 013703 006460 CH74:  MOV TEMCHR,R3     ;R3 NOW CONTAINS PROPER VALUE
1614 006434 042703 177700 VTNOSP: BIC #177700,R3   ;CLEAR GARBAGE
1615 006440 006303          ASL R3      ;MULTIPLY 5 BIT OCTAL BY 4
1616 006442 006303          ASL R3      ;TO FIND RELATIVE POSITION IN TABLE
1617 006444 062703 012652          ADD #TABLE-4,R3     ;THIS IS ABSOLUTE POSITION
1618 006450 004737 006532          JSR PC,TIXDIS      ;DRAW CHARACTER
1619 006454 000703          BR VTWL        ;AND CONSIDER NEXT
1620 006456 000207          VTWDON: RTS PC
1621 006460 000000          TEMCHR: 0          ;SPECIAL CHARS ARE BUILT HERE
1622
1623          ;*****
1624          ;THIS ROUTINE COMPARES THE CHARACTER IN KDBUF (BYTE)
1625          ;AGAINST A LIST POINTED TO BY CONTENTS+1 OF CALL+2.
1626          ;IF A MATCH IS FOUND, CONTROL IS TRANSFERED TO ADDRESS
1627          ;CONTAINED IN CALL +2+2N WHERE N IS THE NUMBER OF
1628          ;THE ENTRY IN THE MATCH TABLE THAT MATCHED.
1629          ;IF NO MATCH IS FOUND, CONTOL IS TRANSFERED TO THE LOCATION FOLLOWING
1630          ;THE LAST TRANSFER ADDRESS.
1631          ;A ZERO IS THE MATCH TABLE TERMINATOR.
1632          ;THE N AND C BITS ARE CLEARED ON EXIT
1633          ;*****
1633 006462 017637 000000 006530 BRAN:  MOV @ (SP),MACHER      ;GET ADDRESS OF MATCH LIST
1634 006470 062716 000002          BRANL:  ADD #2,(SP)        ;ADJUST RETURN POINTER
1635 006474 005237 006530          INC MACHER      ;MOVE MATCH POINTER
1636 006500 105777 000024          TSTB @MACHER     ;A ZERO INDICATES END OF LIST
1637 006504 001406          BEQ NOMACH
1638 006506 123777 001304 000014 CMPB KDBUF,@MACHER
1639 006514 001365          BNE BRANL        ;NO MATCH. TRY AGAIN
1640 006516 017616 000000          MOV @ (SP),(SP)   ;PUT TRANSFER ADDRESS ON STACK
1641 006522 000241          NOMACH: CLC        ;CLEAR CARRY BIT
1642 006524 000250          CLN          ;AND N BIT
1643 006526 000207          RTS PC         ;RETURN APPROPRIATELY
1644 006530 000000          MACHER: 0
1645
1646          ;*****
1647          ;THIS SUBROUTINE IS USED TO DISPLAY A MATRIX OF POINTS
1648          ;THE ROUTINE IS GENERALIZED AND MUST HAVE VARIOUS
1649          ;CONSTANTS SETUP BY THE CALLING PROGRAMS.
1650          ;R3 POINTS TO THE MATRIX CODE WORDS
1651          ;*****
1652
1653
1654 006532 013737 006660 006662 TIXDIS: MOV SHFTK,SHFTCT      ;SET SHIFT COUNT FOR DECODER
1655 006540 013737 006664 006666          MOV ROWK,ROW      ;SETUP THE NUMBER OF ROWS

```

```

1656 006546 013737 006670 006672
1657 006554 012302
1659 006556 013700 001326
1659 006562 013701 001330
1660 006566 006092
1661 006570 103002
1662 006572 004737 005676
1663 006576 005337 006662
1664 006602 001001
1665 006604 011302
1666 006606 063700 001332
1667 006612 005337 006666
1669 006616 001363
1669 006620 013737 006664 006666
1670 006626 063701 001332
1671 006632 005337 006672
1672 006636 001403
1673 006640 013700 001326
1674 006644 000750
1675 006646 063700 001332
1676 006652 010037 001326
1677 006656 000207
1678 006660 000017
1679 006662 000000
1680 006664 000005
1681 006666 000000
1682 006670 000006
1683 006672 000000
1684
1685
1686
1687
1688
1689 006674 007105
1690 006676 000000
1691 006700 005037 006726
1692 006704 004737 006732
1693 006710 006724
1694 006712 007105
1695 006714 004737 006246
1696 006720 006674
1697 006722 000207
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711

```

```

MOV COLK, COL ;AND THE NUMBER OF COLUMNS
MOV (R3)+, R2 ;GET THE WORD TO BE DECODED
MOV BASX, R0 ;R0 WILL CONTAIN THE X COORD.
MOV BASY, R1 ;R1 NOW CONTAINS THE Y COORDINATE
TIXL: ROR R2 ;CHECK CODE WORD FOR BIT
BCC TIXBNO ;IF CARRY BIT CLEAR NO POINT TO DISPLAY
JSR PC, DISPY1 ;GO DISPLAY POINT
TIXBNO: DEC SHFTCT ;COUNT INFO BITS IN WORD
SNE TIXWOK ;CODE WORD OK IF NON ZERO RESULT
MOV (R3), R2 ;GET NEW CODE WORD
TIXWOK: ADD INCXY, R0 ;REFERENCE NEXT POINT
DEC ROW ;DONT GO PAST END OF ROW
SNE TIXL ;NON ZERO OK
MOV ROWK, ROW ;RESET
ADD INCXY, R1 ;MOV Y UP A ROW
DEC COL ;WHEN COL=0 WE ARE DONE
BEQ TIXDON ;RESET X TO LEFT HAND EDGE
MOV BASX, R0
BR TIXL
TIXDON: ADD INCXY, R0 ;MAKE A SPACE BETWEEN LETTERS
MOV R0, BASX ;AND RESET BASX TO NEW VALUE
RTS PC

SHFTK: 15.
SHFTCT: 0
ROWK: 5
ROW: 0
COLK: 6
COL: 0

:*****
:THIS ROUTINE CALLS 'ASCIZP WHICH CONVERTS A SINGLE OR DOUBLE
:PRECISION OCTAL NUMBER TO ASCII DECIMAL AND THEN WRITES IT TO SCREEN
:*****
BLANK: ADUMMY
0
AWRIT: CLR DUMMY2
OTTY: JSR PC, ASCZSP
DUMMY1
ADUMMY
JSR PC, VTWRIT
BLANK
RTS PC

:*****
:THIS ROUTINE CONVERTS A DOUBLE PRECISION OCTAL NUMBER IN CORE
:TO AN ASCII DECIMAL NUMBER.
:THE CALL IS JSR PC, ASCIZ
:WITH THE ADDRESS OF THE LOW ORDER OCTAL WORD IN
:CALL+2
:AND THE ADDRESS OF THE PLACE THE ASCII DECIMAL IS TO BE STORED
:IN CALL+4
:THE HIGH ORDER OCTAL WORD MUST BE IN THE LOCATION FOLLOWING
:THE LOW ORDER WORD.
:THE SUBROUTINE SHOULD BE CALLED BY
:JSR PC, ASCZSP
:TO SUPPRESS LEADING ZEROES AND TO LEFT JUSTIFY THE

```

```

1712
1713
1714
1715 006724 000000
1716 006726 000000
1717 006730 000000
1718 006732 005237 006730
1719 006736 017601 000000
1720 006742 062716 000002
1721 006746 012103
1722 006750 011102
1723 006752 012737 000012 007070
1724 006760 012705 007072
1725 006764 112725 000200
1726 006770 012704 000012
1727 006774 010546
1728 006776 004737 007120
1729 007002 012605
1730 007004 062702 000060
1731 007010 110225
1732 007012 010103
1733 007014 010002
1734 007016 005337 007070
1735 007022 001362
1736
1737 007024 005737 006730
1738 007030 001410
1739 007032 005037 006730
1740 007036 124527 000060
1741 007042 001775
1742 007044 105725
1743 007046 100001
1744 007050 005205
1745 007052 017600 000000
1746 007056 062716 000002
1747 007062 114520
1748 007064 100376
1749 007066 000207
1750 007070 000000
1751 007072 007105
1752 007105 007120
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
    
```

```

:ANSWER. UNLESS THIS LAST CALL IS MADE THE OUTPUT
:WILL ALWAYS BE 11 BYTES.
:*****
DUMMY1: 0
DUMMY2: 0
ZESUP: 0 ;ZERO SUPPRESS FLAG
ASCZSP: INC ZESUP ;SET FLAG
ASCIZ: MOV @ (SP), R1 ;GET LOW ORDER ADDRESS
      ADD #2, (SP) ;ADJUST POINTER
      MOV (R1)+, R3 ;GET LOW ORDER
      MOV (R1), R2 ;GET HIGH ORDER
      MOV #10, DIVCNT ;10 POSSIBLE DIGITS
      MOV #ASCRES, R5 ;ADDRESS OF DIGITS
      MOV# #200, (R5)+ ;CHGEN TERMINATOR
ASCLP: MOV #10, R4 ;DIVISOR
      MOV R5, -(SP) ;SAVE R5 DURING DIVISION
      JSR PC, DIVIDE ;DIVIDE BY 10
      MOV (SP)+, R5 ;RESTORE R5
      ADD #60, R2 ;MAKE IT ASCII
      MOV# R2, (R5)+ ;PUT IT IN THE ASCII STRING
      MOV R1, R3
      MOV R0, R2 ;PUT ANSWER IN DIVIDEND SLOT
      DEC DIVCNT
      BNE ASCLP
:NOW TRANSFER THE ASCII TO PROPER PLACE IN CORRECT ORDER
      TST ZESUP ;SUPPRESS ZEROES?
      BEQ MOVALL ;NO MOVE ALL
      CLR ZESUP ;INIT THE SWITCH
SKZE: CMPB -(R5), #60 ;LOOK FOR ZEROES
      BEQ SKZE ;AND PASS BY THEM
      TSTB (R5)+ ;BACK-UP
      BPL MOVALL ;AT LEAST ONE ZERO
      INC R5 ;POINT TO FIRST NUMBER
MOVALL: MOV @ (SP), R0 ;ADDRESS OF RESULTS
      ADD #2, (SP) ;ADJUST RETURN
MOVAL: MOV# -(R5), (R0)+
      BPL MOVAL ;200 BYTE IS TERMINATOR
      RTS PC
DIVCNT: 0
ASCRES: .=.+11.
ADUMMY: .=.+11.
      .EVEN
:*****
:THIS ROUTINE DIVIDES A POSITIVE DOUBLE PRECISION NUMBER
:FOUND IN R2 (HI) AND R3 (LO) BY THE POSITIVE NUMBER IN
:R4. THE ANSWER IS PLACED IN R0 (HI) AND R1 (LO).
:THE ROUTINE LEFT JUSTIFIES DIVISOR KEEPING TRACK OF SHIFTS
:SO THAT FIRST BIT IS IN BIT 14 OF R4. THE SHIFT COUNT IS THEN
:ADDED TO 16. AND THE DIVISION IS STARTED BY COMPARING THE
:DIVISOR (SHIFTED) TO THE HI ORDER OF THE DIVIDEND.
:IF COMPARISON SHOWS DIVISOR SMALLER, A SUBTRACTION IS DONE
:BEETWEEN DIVSOR AND HI ORDER DIVIDEND AND THE ANSWER IS INCREMENTED.
:REGARDLESS OF COMPARISON RESULT, BOTH THE ANSWER AND
:THE DIVIDEND ARE MULTIPLIED BY 2.
:THIS SEQUENCE IS PERFORMED THE NUMBER OF TIMES INDICATED BY
    
```

```

1768
1769
1770
1771 007120 005005
1772 007122 005000
1773 007124 005001
1774 007126 005205
1775 007130 005304
1776 007132 100375
1777 007134 005004
1778 007136 010537 007206
1779 007142 062705 000020
1780 007146 005301
1781 007150 005100
1782 007152 020204
1783 007154 100402
1784 007156 160402
1785 007160 005201
1786 007162 005303
1787 007164 006102
1788 007166 005305
1789 007170 001366
1790 007172 000241
1791 007174 006002
1792 007176 005337 007206
1793 007202 001374
1794 007204 000207
1795 007206 000000
1796
1797
1798
1799
1800 007210 010046
1801 007212 013700 001246
1802 007216 013702 007436
1803 007222 010022
1804 007224 020227 007436
1805 007230 103402
1806 007232 012702 007336
1807 007236 010237 007436
1808 007242 012702 007336
1809 007246 004737 007300
1810 007252 110046
1811 007254 012702 007337
1812 007260 004737 007300
1813 007264 000300
1814 007266 152600
1815 007270 010037 001246
1816 007274 012600
1817 007276 000207
1818
1819 007300 005000
1820 007302 005005
1821 007304 152205
1822 007306 060500
1823 007310 005202

```

```

:SHIFTCOUNT.
:R2R3 / R4=ROR1
:*****
DIVIDE: CLR R5 ;INIT SHIFT COUNT
        CLR R0
        CLR R1 ;ZERO THE ANSWER
DIVJUS: INC R5 ;THIS IS THE SHIFT COUNT
        ASL R4 ;DIVSOR X 2
        BPL DIVJUS ;GO UNTIL BIT 15 SETS
        ROR R4 ;MOV IT BACK ONE
        MOV R5, RMJUST ;SAVE SIFT COUNT FOR REMAN. JUST
        ADD #16., R5 ;SIMULATE 16 BIT SHIFT
DIVL00: ASL R1 ;ANSWER X 2
        ROL R0
        CMP R2, R4 ;COMPARE DIVISOR AND DIVIDEND
        BMI NOSUB ;IF DIVSOR LARGER NO SUBTRCTION
        SUB R4, R2
        INC R1 ;NOTE SUBTRACTION IN ANSWER
NOSUB: ASL R3 ;MAKE DIVIDEND LARGER
        ROL R2 ;BY 2
        DEC R5 ;DO ALL THIS SHIFCNT TIMES
        BNE DIVL00
        CLC
RLJL: ROR R2 ;NOW RIGHT JUSTIFY THE REMAINDER
      DEC RMJUST
      BNE RLJL
      RTS PC
RMJUST: 0
:*****
:THIS ROUTINE AVERAGES THE LAST 32. X-Y JOYSTICK VALUES
:*****
XYAVE: MOV RO, -(SP) ;SAVE RO
        MOV TEMPO, RO ;GET X & Y
        MOV XYBUF, R2 ;GET CURRENT BUFFER POINTER
        MOV RO, (R2)+ ;SAVE NEW X-Y VALUE
        CMP R2, #XYBUFE ;END OF BUFFER?
        BLO 1$ ;NO
        MOV #XYBUF, R2 ;YES, GO BACK TO BEGINING OF BUFFER
1$: MOV R2, XYBUF ;SAVE NEW BUFFER POINTER
     MOV #XYBUF, R2 ;CALC AVE X
     JSR PC, 10$
     MOVB RO, -(SP) ;SAVE IT
     MOV #XYBUF+1, R2 ;CALC AVE Y
     JSR PC, 10$ ;GO DO IT
     SWAB RO ;GET X-Y IN RO
     BISB (SP)+, RO ;GET SAVED X
     MOV RO, TEMPO ;PUT IN TEMPO
     MOV (SP)+, RO ;RESTORE RO
     RTS PC ;EXIT WITH AVE X-Y IN TEMPO
10$: CLR RO ;ZERO SUM
11$: CLR R5 ;DO A MOVB TO A REG (UNSIGNED)
     BISB (R2)+, R5
     ADD R5, RO ;ADD IT IN
     INC R2 ;SKIP OTHER VALUE

```

1824	007312	020227	007436	CMP	R2,#XYBUFE	;END OF BUFFER?
1825	007316	103771		BLO	11\$;NO
1826	007320	006300		ASL	RO	;DIVIDE BY 32.
1827	007322	006300		ASL	RO	
1828	007324	006300		ASL	RO	
1829	007326	000300		SWAB	RO	
1830	007330	042700	177400	BIC	#177400,RO	;CLR HI BYTE
1831	007334	000207		RTS	PC	

1832						
1833	007336		XYBUF:			
1834	007336	000000		0		
1835	007340	000000		0		
1836	007342	000000		0		
1837	007344	000000		0		
1838	007346	000000		0		
1839	007350	000000		0		
1840	007352	000000		0		
1841	007354	000000		0		
1842	007356	000000		0		
1843	007360	000000		0		
1844	007362	000000		0		
1845	007364	000000		0		
1846	007366	000000		0		
1847	007370	000000		0		
1848	007372	000000		0		
1849	007374	000000		0		
1850	007376	000000		0		
1851	007400	000000		0		
1852	007402	000000		0		
1853	007404	000000		0		
1854	007406	000000		0		
1855	007410	000000		0		
1856	007412	000000		0		
1857	007414	000000		0		
1858	007416	000000		0		
1859	007420	000000		0		
1860	007422	000000		0		
1861	007424	000000		0		
1862	007426	000000		0		
1863	007430	000000		0		
1864	007432	000000		0		
1865	007434	000000		0		
1866		007436	XYBUFE=			
1867	007436	007336	XYBUFP: XYBUF			

```

;*****
;SBTTL TTY INPUT ROUTINE
;*****
;ENABL LSB
;*****
;SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
;ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
;SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
;WHEN OPERATING IN TTY FLAG MODE.

```

1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879

1880	007440	022737	000176	001140	\$CKSWR:	CMP	#SWREG, SWR	:: IS THE SOFT-SWR SELECTED?
1881	007446	001074				BNE	15\$:: BRANCH IF NO
1882	007450	105777	171470			TSTB	2\$TKS	:: CHAR THERE?
1883	007454	100071				BPL	15\$:: IF NO, DON'T WAIT AROUND
1884	007456	117746	171464			MOVB	2\$TKB, -(SP)	:: SAVE THE CHAR
1885	007462	042716	177600			BIC	#1C177, (SP)	:: STRIP-OFF THE ASCII
1886	007466	022726	000007			CMP	#7, (SP)+	:: IS IT A CONTROL G?
1887	007472	001062				BNE	15\$:: NO, RETURN TO USER
1888	007474	123727	001134	000001		CMPB	\$AUTOB, #1	:: ARE WE RUNNING IN AUTO-MODE?
1889	007502	001456				BEQ	15\$:: BRANCH IF YES
1890								
1891	007504	104400	010312			TYPE	,\$CNTLG	:: ECHO THE CONTROL-G (↑G)
1892	007510	104400	010317		\$GTSWR:	TYPE	,\$MSWR	:: TYPE CURRENT CONTENTS
1893	007514	013746	000176			MOV	\$WREG, -(SP)	:: SAVE SWREG FOR TYPEOUT
1894	007520	104401				TYPOC		:: GO TYPE--OCTAL ASCII(ALL DIGITS)
1895	007522	104400	010330			TYPE	,\$MNEW	:: PROMPT FOR NEW SWR
1896	007526	005046			19\$:	CLR	-(SP)	:: CLEAR COUNTER
1897	007530	005046				CLR	-(SP)	:: THE NEW SWR
1898	007532	105777	171406		7\$:	TSTB	2\$TKS	:: CHAR THERE?
1899	007536	100375				BPL	7\$:: IF NOT TRY AGAIN
1900								
1901	007540	117746	171402			MOVB	2\$TKB, -(SP)	:: PICK UP CHAR
1902	007544	042716	177600			BIC	#1C177, (SP)	:: MAKE IT 7-BIT ASCII
1903								
1904								
1905								
1906	007550	021627	000025		9\$:	CMP	(SP), #25	:: IS IT A CONTROL-U?
1907	007554	001005				BNE	10\$:: BRANCH IF NOT
1908	007556	104400	010305			TYPE	,\$CNTLU	:: YES, ECHO CONTROL-U (↑U)
1909	007562	062706	000006		20\$:	ADD	#6, SP	:: IGNORE PREVIOUS INPUT
1910	007566	000757				BR	19\$:: LET'S TRY IT AGAIN
1911								
1912								
1913	007570	021627	000015		10\$:	CMP	(SP), #15	:: IS IT A <CR>?
1914	007574	001022				BNE	16\$:: BRANCH IF NO
1915	007576	005766	000004			TST	4(SP)	:: YES, IS IT THE FIRST CHAR?
1916	007602	001403				BEQ	11\$:: BRANCH IF YES
1917	007604	016677	000002	171326		MOV	2(SP), 2SWR	:: SAVE NEW SWR
1918	007612	062706	000006		11\$:	ADD	#6, SP	:: CLEAR UP STACK
1919	007616	104400	001161		14\$:	TYPE	,\$CRLF	:: ECHO <CR> AND <LF>
1920	007622	123727	001135	000001		CMPB	\$INTAG, #1	:: RE-ENABLE TTY KBD INTERRUPTS?
1921	007630	001003				BNE	15\$:: BRANCH IF NOT
1922	007632	012777	000100	171304		MOV	#100, 2\$TKS	:: RE-ENABLE TTY KBD INTERRUPTS
1923	007640	000002			15\$:	RTI		:: RETURN
1924	007642	004737	010512		16\$:	JSR	PC, \$TYPEC	:: ECHO CHAR
1925	007646	021627	000060			CMP	(SP), #60	:: CHAR < 0?
1926	007652	002420				BLT	18\$:: BRANCH IF YES
1927	007654	021627	000067			CMP	(SP), #67	:: CHAR > 7?
1928	007660	003015				BGT	18\$:: BRANCH IF YES
1929	007662	042726	000060			BIC	#60, (SP)+	:: STRIP-OFF ASCII
1930	007666	005766	000002			TST	2(SP)	:: IS THIS THE FIRST CHAR
1931	007672	001403				BEQ	17\$:: BRANCH IF YES
1932	007674	006316				ASL	(SP)	:: NO, SHIFT PRESENT
1933	007676	006316				ASL	(SP)	:: CHAR OVER TO MAKE
1934	007700	006316				ASL	(SP)	:: ROOM FOR NEW ONE.
1935	007702	005266	000002		17\$:	INC	2(SP)	:: KEEP COUNT OF CHAR


```

007706 056616 177776      BIS      -2(SP), (SP)      :: SET IN NEW CHAR
007712 000707      BR       7$              :: GET THE NEXT ONE
007714 104400 001160 18$:     TYPE      $QUES      :: TYPE ?(CR)(LF)
007720 000720      BR       20$           :: SIMULATE CONTROL-U
.DSABL  LSB

```

*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
*CALL:

* RDCHR :: INPUT A SINGLE CHARACTER FROM THE TTY
* RETURN HERE :: CHARACTER IS ON THE STACK
* :: WITH PARITY BIT STRIPPED OFF

```

007722 011646 000004 000002 $RDCHR: MOV      (SP), -(SP)      :: PUSH DOWN THE PC
007724 016666 000004 000002      MOV      4(SP), 2(SP)    :: SAVE THE PS
007732 109777 171206 1$:     TSTB     2$TKS        :: WAIT FOR
007736 100375      BPL      1$              :: A CHARACTER
007740 117766 171202 000004      MOVB     2$TKB, 4(SP)    :: READ THE TTY
007746 042766 177600 000004      BIC      #177, 4(SP)     :: GET RID OF JUNK IF ANY
007754 026627 000004 000023      CMP      4(SP), #23     :: IS IT A CONTROL-S?
007762 001013      BNE      3$              :: BRANCH IF NO
007764 105777 171154 2$:     TSTB     2$TKS        :: WAIT FOR A CHARACTER
007770 100375      BPL      2$              :: LOOP UNTIL ITS THERE
007772 117746 171150      MOVB     2$TKB, -(SP)    :: GET CHARACTER
007776 042716 177600      BIC      #177, (SP)     :: MAKE IT 7-BIT ASCII
008002 022627 000021      CMP      (SP)+, #21     :: IS IT A CONTROL-Q?
008006 001366      BNE      2$              :: IF NOT DISCARD IT
008010 000750      BR       1$              :: YES, RESUME
008012 026627 000004 000140 3$:     CMP      4(SP), #140    :: IS IT UPPER CASE?
008020 002407      BLT      4$              :: BRANCH IF YES
008022 026627 000004 000175      CMP      4(SP), #175    :: IS IT A SPECIAL CHAR?
008030 003003      BGT      4$              :: BRANCH IF YES
008032 042766 000040 000004      BIC      #40, 4(SP)     :: MAKE IT UPPER CASE
008040 000002      RTI                    :: GO BACK TO USER

```

*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
*CALL:

* RDLIN :: INPUT A STRING FROM THE TTY
* RETURN HERE :: ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
* :: TERMINATOR WILL BE A BYTE OF ALL 0'S

```

010042 010246 $RDLIN: MOV      R3, -(SP)      :: SAVE R3
010044 005046      CLR      -(SP)         :: CLEAR THE RUBOUT KEY
010046 012703 010276 1$:     MOV      #STTYIN, R3    :: GET ADDRESS
010052 022703 010305 2$:     CMP      #STTYIN+7, R3  :: BUFFER FULL?
010056 101456      BLOS     4$              :: BR IF YES
010060 104406      RDCHR   :: GO READ ONE CHARACTER FROM THE TTY
010062 112613      MOVB     (SP)+, (R3)    :: GET CHARACTER
010064 122713 000177 10$:    CMPB     #177, (R3)     :: IS IT A RUBOUT
010070 001022      BNE      5$              :: BR IF NO
010072 005716      TST     (SP)           :: IS THIS THE FIRST RUBOUT?
010074 001007      BNE      6$              :: BR IF NO
010076 112737 000134 010274      MOVB     #'\.9$,      :: TYPE A BACK SLASH
010104 104400 010274      TYPE     .9$

```

```

1992 010110 012716 177777      MOV      # -1, (SP)      ;; SET THE RUBOUT KEY
1993 010114 005303      65:     DEC      R3          ;; BACKUP BY ONE
1994 010116 020327 010276     CMP      R3, #STTYIN   ;; STACK EMPTY?
1995 010122 103434      BLO      45           ;; BR IF YES
1996 010124 111337 010274     MOV      (R3), 95      ;; SETUP TO TYPEOUT THE DELETED CHAR.
1997 010130 104400 010274     TYPE    95           ;; GO TYPE
1998 010134 000746      BR      25           ;; GO READ ANOTHER CHAR.
1999 010136 005716      55:     TST      (SP)         ;; RUBOUT KEY SET?
2000 010140 001406      BEQ      75           ;; BR IF NO
2001 010142 112737 000134 010274     MOV      # '\, 95     ;; TYPE A BACK SLASH
2002 010150 104400 010274     TYPE    95           ;;
2003 010154 005016      CLR      (SP)         ;; CLEAR THE RUBOUT KEY
2004 010156 122713 000025      75:     CMP      #25, (R3)    ;; IS CHARACTER A CTRL U?
2005 010162 001003      BNE      85           ;; BR IF NO
2006 010164 104400 010305     TYPE    , SCNTLU     ;; TYPE A CONTROL "U"
2007 010170 000726      BR      15           ;; GO START OVER
2008 010172 122713 000022      85:     CMP      #22, (R3)    ;; IS CHARACTER A "R"?
2009 010176 001011      BNE      35           ;; BRANCH IF NO
2010 010200 105013      CLRB    (R3)         ;; CLEAR THE CHARACTER
2011 010202 104400 001161     TYPE    , SCRLF     ;; TYPE A "CR" & "LF"
2012 010206 104400 010276     TYPE    , STTYIN    ;; TYPE THE INPUT STRING
2013 010212 000717      BR      25           ;; GO PICKUP ANOTHER CHARACTER
2014 010214 104400 001160      45:     TYPE    , SQUES     ;; TYPE A '?'
2015 010220 000712      BR      15           ;; CLEAR THE BUFFER AND LOOP
2016 010222 111337 010274      35:     MOV      (R3), 95    ;; ECHO THE CHARACTER
2017 010226 104400 010274     TYPE    95           ;;
2018 010232 122723 000015     CMP      #15, (R3)+   ;; CHECK FOR RETURN
2019 010236 001305      BNE      25           ;; LOOP IF NOT RETURN
2020 010240 105063 177777     CLRB    -1(R3)       ;; CLEAR RETURN (THE 15)
2021 010244 104400 001162     TYPE    , SLF        ;; TYPE A LINE FEED
2022 010250 005726      TST      (SP)+       ;; CLEAN RUBOUT KEY FROM THE STACK
2023 010252 012603      MOV      (SP)+, R3    ;; RESTORE R3
2024 010254 011646      MOV      (SP), -(SP)  ;; ADJUST THE STACK AND PUT ADDRESS OF THE
2025 010256 016666 000004 000002      MOV      4(SP), 2(SP) ;; FIRST ASCII CHARACTER ON IT
2026 010264 012766 010276 000004      MOV      #STTYIN, 4(SP)
2027 010272 000002      RTI                ;; RETURN
2028 010274 000      95:     .BYTE   0           ;; STORAGE FOR ASCII CHAR. TO TYPE
2029 010276 000      .BYTE   0           ;; TERMINATOR
2030 010278 000007      .BLKB   7           ;; RESERVE 7 BYTES FOR TTY INPUT
2031 010305 136 006525 000012  STTYIN: .ASCIZ  /?U/<15><12>  ;; CONTROL "U"
2032 010312 043536 005015 000  SCNTLU: .ASCIZ  /?G/<15><12>  ;; CONTROL "G"
2033 010317 015 051412 051127  SMSWR: .ASCIZ  <15><12>/SWR = /
2034 010324 036440 000040  SMNEW: .ASCIZ  / NEW = /
2035 010330 020040 042516 020127
2036 010336 020075 000
2037 010342

```

```

.EVEN
*****
.SBTTL TYPE ROUTINE
*****
*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
*NOTE1: $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
*NOTE2: $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
*NOTE3: $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
*
```

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
000101
000102
000103

010342 105737 001157
010346 100002
010350 000000
010352 000407
010354 010046
010356 017600 000002
010362 112046
010364 001005
010366 005726
010370 012600
010372 062716 000002
010376 000002
010400 122716 000011
010404 001430
010406 122716 000200
010412 001006
010414 005726
010416 104400
010420 001161
010422 105037 010556
010426 000755
010430 004737 010512
010434 123726 001156
010440 001350
010442 013746 001154
010446 105366 000001
010452 002770
010454 004737 010512
010460 105337 010556
010464 000770

010466 112716 000040
010472 004737 010512 010556
010476 132737 000007
010504 001372
010506 005726
010510 000724
010512 105777 170432
010516 100375
010520 116677 000002 170424
010526 122766 000015 000002
010534 001003
010536 105037 010556
010542 000406
010544 122766 000012 000002

;;CALL:
;*1) USING A TRAP INSTRUCTION
;* TYPE .MESADR
;*OR
;* TYPE
;* MESADR
;*
\$TYPE: TSTB \$TFPLG
BPL 1\$
HALT
BR 3\$
1\$: MOV RO,-(SP)
MOV 2\$(SP),RO
2\$: MOVB (RO)+,-(SP)
BNE 4\$
TST (SP)+
60\$: MOV (SP)+,RO
3\$: ADD #2,(SP)
RTI
4\$: CMPB #HT,(SP)
BEQ 8\$
CMPB #CRLF,(SP)
BNE 5\$
TST (SP)+
TYPE
\$CRLF
CLRB \$CHARCNT
BR 2\$
5\$: JSR PC,\$TYPEC
6\$: CMPB \$FILLC,(SP)+
BNE 2\$
MOV \$NULL,-(SP)
7\$: DECB 1(SP)
BLT 6\$
JSR PC,\$TYPEC
DECB \$CHARCNT
BR 7\$

;HORIZONTAL TAB PROCESSOR
8\$: MOVB #' ,(SP)
9\$: JSR PC,\$TYPEC
BITB #7,\$CHARCNT
BNE 9\$
TST (SP)+
BR 2\$
\$TYPEC: TSTB 2\$TPS
BPL \$TYPEC
MOVB 2(SP),2\$TPB
CMPB #CR,2(SP)
BNE 1\$
CLRB \$CHARCNT
BR \$TYPEX
1\$: CMPB #LF,2(SP)

;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING

;; IS THERE A TERMINAL?
;; BR IF YES
;; HALT HERE IF NO TERMINAL
;; LEAVE
;; SAVE RO
;; GET ADDRESS OF ASCIZ STRING
;; PUSH CHARACTER TO BE TYPED ONTO STACK
;; BR IF IT ISN'T THE TERMINATOR
;; IF TERMINATOR POP IT OFF THE STACK
;; RESTORE RO
;; ADJUST RETURN PC
;; RETURN
;; BRANCH IF <HT>

;; BRANCH IF NOT <CRLF>

;; POP <CR><LF> EQUIV
;; TYPE A CR AND LF

;; CLEAR CHARACTER COUNT
;; GET NEXT CHARACTER
;; GO TYPE THIS CHARACTER
;; IS IT TIME FOR FILLER CHARS.?
;; IF NO GO GET NEXT CHAR.
;; GET # OF FILLER CHARS. NEEDED
;; AND THE NULL CHAR.
;; DOES A NULL NEED TO BE TYPED?
;; BR IF NO--GO POP THE NULL OFF OF STACK
;; GO TYPE A NULL
;; DO NOT COUNT AS A COUNT
;; LOOP

;; REPLACE TAB WITH SPACE
;; TYPE A SPACE
;; BRANCH IF NOT AT
;; TAB STOP
;; POP SPACE OFF STACK
;; GET NEXT CHARACTER
;; WAIT UNTIL PRINTER IS READY

;; LOAD CHAR TO BE TYPED INTO DATA REG.
;; IS CHARACTER A CARRIAGE RETURN?
;; BRANCH IF NO
;; YES--CLEAR CHARACTER COUNT
;; EXIT
;; IS CHARACTER A LINE FEED?

0104 010552 001402
010554 105327
010556 000000
010558 000000
010560 000000
010562 017646
010564 116637
010566 112637
010568 062716
010570 000406
010572 112737
010574 112737
010576 112737
010578 010346
010580 010446
010582 010546
010584 113704
010586 005404
010588 062704
010590 110437
010592 113704
010594 016605
010596 005003
010598 006105
010600 000404
010602 006105
010604 006105
010606 006105
010700 010503
010702 006102

```
BEG $TYPEX ;:BRANCH IF YES
INCB (PC)+ ;:COUNT THE CHARACTER
$CHARCNT: WORD 0 ;:CHARACTER COUNT STORAGE
$TYPEX: RTS PC
```

: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.
:*STYPOS---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
```

*STYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
:*STYPOS 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),%0FILL ;:LOAD ZERO FILL SWITCH
        MOV     (SP)+,%0MODE+1 ;:NUMBER OF DIGITS TO TYPE
        ADD     #2,(SP) ;:ADJUST RETURN ADDRESS
        BR     $TYPON
$TYPOC: MOV     #1,%0FILL ;:SET THE ZERO FILL SWITCH
        MOV     #6,%0MODE+1 ;:SET FOR SIX(6) DIGITS
$TYPON: MOV     #5,%0CNT ;:SET THE ITERATION COUNT
        MOV     R3,-(SP) ;:SAVE R3
        MOV     R4,-(SP) ;:SAVE R4
        MOV     R5,-(SP) ;:SAVE R5
        MOV     %0MODE+1,R4 ;:GET THE NUMBER OF DIGITS TO TYPE
        NEG     R4 ;:SUBTRACT IT FOR MAX. ALLOWED
        ADD     #6,R4 ;:SAVE IT FOR USE
        MOV     R4,%0MODE ;:GET THE ZERO FILL SWITCH
        MOV     %0FILL,R4 ;:PICKUP THE INPUT NUMBER
        MOV     12(SP),R5 ;:CLEAR THE OUTPUT WORD
        CLR     R3 ;:ROTATE MSB INTO "C"
        ROL     R5 ;:GO DO MSB
        BR     3$ ;:FORM THIS DIGIT
1$: ROL     R5
2$: ROL     R5
3$: ROL     R5
        MOV     R5,R3 ;:GET LSB OF THIS DIGIT
        ROL     R3
```

```

2160 010704 105337 011006      DECB      $OMODE      :: TYPE THIS DIGIT?
2161 010710 100016      BPL       7$      :: BR IF NO
2162 010712 042703 177770      BIC      #177770,R3  :: GET RID OF JUNK
2163 010716 001002      BNE      4$      :: TEST FOR 0
2164 010720 005704      TST      R4      :: SUPPRESS THIS 0?
2165 010722 001403      BEQ      5$      :: BR IF YES
2166 010724 005204      INC      R4      :: DON'T SUPPRESS ANYMORE 0'S
2167 010726 052703 000060      BIS      #'0,R3   :: MAKE THIS DIGIT ASCII
2168 010732 052703 000040      BIS      #' ,R3   :: MAKE ASCII IF NOT ALREADY
2169 010736 110337 011002      MOV      R3,8$    :: SAVE FOR TYPING
2170 010742 104400 011002      TYPE     8$      :: GO TYPE THIS DIGIT
2171 010746 105337 011004      DECB     $OCNT    :: COUNT BY 1
2172 010752 003347      BGT      2$      :: BR IF MORE TO DO
2173 010754 002402      BLT      6$      :: BR IF DONE
2174 010756 005204      INC      R4      :: INSURE LAST DIGIT ISN'T A BLANK
2175 010760 000744      BR       2$      :: GO DO THE LAST DIGIT
2176 010762 012605      MOV      (SP)+,R5  :: RESTORE R5
2177 010764 012604      MOV      (SP)+,R4  :: RESTORE R4
2178 010766 012603      MOV      (SP)+,R3  :: RESTORE R3
2179 010770 016666 000002 000004      MOV      2(SP),4(SP) :: SET THE STACK FOR RETURNING
2180 010776 012616      MOV      (SP)+,(SP)
2181 011000 000002      RTI      :: RETURN
2182 011002 000      .BYTE   0        :: STORAGE FOR ASCII DIGIT
2183 011003 000      .BYTE   0        :: TERMINATOR FOR TYPE ROUTINE
2184 011004 000      .BYTE   0        :: OCTAL DIGIT COUNTER
2185 011005 000      .BYTE   0        :: ZERO FILL SWITCH
2186 011006 000000      .WORD   0        :: NUMBER OF DIGITS TO TYPE
*****
.SBTTL READ AN OCTAL NUMBER FROM THE TTY
*****
*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
*CHANGE IT TO BINARY.
*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
*OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPED
*FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
*THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
*CALL:
*      RDOCT      :: READ AN OCTAL NUMBER
*      RETURN HERE  :: LOW ORDER BITS ARE ON TOP OF THE STACK
*                   :: HIGH ORDER BITS ARE IN $HI OCT
SRDOCT: MOV      (SP)-,(SP)  :: PROVIDE SPACE FOR THE
MOV      4(SP),2(SP)    :: INPUT NUMBER
MOV      R0,-(SP)      :: PUSH R0 ON STACK
MOV      R1,-(SP)      :: PUSH R1 ON STACK
MOV      R2,-(SP)      :: PUSH R2 ON STACK
1$: RDLIN      :: READ AN ASCII LINE
MOV      (SP)+,R0      :: GET ADDRESS OF 1ST CHARACTER
MOV      R0,5$        :: AND SAVE IT
CLR      R1           :: CLEAR DATA WORD
2$: MOV      (R0)+,-(SP) :: PICKUP THIS CHARACTER
BEQ      3$          :: IF ZERO GET OUT
CMP      #'0,(SP)    :: MAKE SURE THIS CHARACTER
BGT      4$          :: IS AN OCTAL DIGIT

```

```

2216 011054 122716 000067
2217 011060 002423
2218 011062 006301
2219 011064 006102
2220 011066 006301
2221 011070 006102
2222 011072 006301
2223 011074 006102
2224 011076 042716 177770
2225 011102 062601
2226 011104 000756
2227 011106 005726
2228 011110 010166 000012
2229 011114 010237 011146
2230 011120 012602
2231 011122 012601
2232 011124 012600
2233 011126 000002
2234 011130 005726
2235 011132 105010
2236 011134 104400
2237 011136 000000
2238 011140 104400 001160
2239 011144 000730
2240 011146 000000
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250 011150 010046
2251 011152 016600 000002
2252 011156 005740
2253 011160 111000
2254 011162 006300
2255 011164 016000 011172
2256 011170 000200
2257
2258
2259
2260
2261
2262
2263
2264
2265 011172
2266 011172 010342
2267 011174 010606
2268 011176 010562
2269 011200 010622
2270
2271 011202 007510

```

```

CMPB #7,(SP)
BLT 4$
ASL R1 ;;*2
ROL R2
ASL R1 ;;*4
ROL R2
ASL R1 ;;*8
ROL R2
BIC #107,(SP) ;;STRIP THE ASCII JUNK
ADD (SP)+,R1 ;;ADD IN THIS DIGIT
BR 2$ ;;LOOP
3$: TST (SP)+ ;;CLEAN TERMINATOR FROM STACK
MOV R1,12(SP) ;;SAVE THE RESULT
MOV R2,$SHIOCT
MOV (SP)+,R2 ;;POP STACK INTO R2
MOV (SP)+,R1 ;;POP STACK INTO R1
MOV (SP)+,R0 ;;POP STACK INTO R0
RTI ;;RETURN
4$: TST (SP)+ ;;CLEAN PARTIAL FROM STACK
CLRB (R0) ;;SET A TERMINATOR
TYPE ;;TYPE UP THRU THE BAD CHAR.
5$: .WORD 0
TYPE $QUES ;;"? "CR" & "LF"
BR 1$ ;;TRY AGAIN
$SHIOCT: .WORD 0 ;;HIGH ORDER BITS GO HERE
*****
.SBTTL TRAP DECODER

```

```

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

```

```

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

```

.SBTTL TRAP TABLE

```

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

```

	ROUTINE		
\$TRPAD:	\$TYPE	::CALL=TYPE	TRAP+0(104400) TTY TYPEOUT ROUTINE
	\$TYPOC	::CALL=TYPOC	TRAP+1(104401) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
	\$TYPOS	::CALL=TYPOS	TRAP+2(104402) TYPE OCTAL NUMBER (NO LEADING ZEROS)
	\$TYPON	::CALL=TYPON	TRAP+3(104403) TYPE OCTAL NUMBER (AS PER LAST CALL)
	\$GTSWR	::CALL=GTSWR	TRAP+4(104404) GET SOFT-SWR SETTING

0272									
0273	011204	007440			\$CKSWR	::CALL=CKSWR	TRAP+5(104405)	TEST FOR CHANGE IN SOFT-SWR	
0274	011206	007722			\$RDCHR	::CALL=RDCHR	TRAP+6(104406)	TTY TYPEIN CHARACTER ROUTINE	
0275	011210	010042			\$RDLIN	::CALL=RDLIN	TRAP+7(104407)	TTY TYPEIN STRING ROUTINE	
0276	011212	011010			\$RDOCT	::CALL=RDOCT	TRAP+10(104410)	READ AN OCTAL NUMBER FROM TTY	

::*****
:SBTTL POWER DOWN AND UP ROUTINES

::*****
:POWER DOWN ROUTINE

0291					\$PWRDN:	MOV	#\$ILLUP,2#PWRVEC	::SET FOR FAST UP	
0292	011214	012737	011360	000024		MOV	#340,2#PWRVEC+2	::PRIO:7	
0293	011222	012737	000340	000026		MOV	R0,-(SP)	::PUSH R0 ON STACK	
0294	011230	010046				MOV	R1,-(SP)	::PUSH R1 ON STACK	
0295	011232	010146				MOV	R2,-(SP)	::PUSH R2 ON STACK	
0296	011234	010246				MOV	R3,-(SP)	::PUSH R3 ON STACK	
0297	011236	010346				MOV	R4,-(SP)	::PUSH R4 ON STACK	
0298	011240	010446				MOV	R5,-(SP)	::PUSH R5 ON STACK	
0299	011242	010546				MOV	2JSWR,-(SP)	::PUSH 2JSWR ON STACK	
0290	011244	017746	167670			MOV	SP,\$SAVR6	::SAVE SP	
0291	011250	010637	011364			MOV	#\$PWRUP,2#PWRVEC	::SET UP VECTOR	
0292	011254	012737	011266	000024		MOV			
0293	011262	000000				HALT			
0294	011264	000776				BR	.-2	::HANG UP	

::*****
:POWER UP ROUTINE

0299	011266	012737	011360	000024	\$PWRUP:	MOV	#\$ILLUP,2#PWRVEC	::SET FOR FAST DOWN	
0299	011274	013706	011364			MOV	\$\$SAVR6,SP	::GET SP	
0300	011300	005037	011364			CLR	\$\$SAVR6	::WAIT LOOP FOR THE TTY	
0301	011304	005237	011364		1\$:	INC	\$\$SAVR6	::WAIT FOR THE INC	
0302	011310	001375				BNE	1\$::OF WORD	
0303	011312	012677	167622			MOV	(SP)+,2JSWR	::POP STACK INTO 2JSWR	
0304	011316	012605				MOV	(SP)+,R5	::POP STACK INTO R5	
0305	011320	012604				MOV	(SP)+,R4	::POP STACK INTO R4	
0306	011322	012603				MOV	(SP)+,R3	::POP STACK INTO R3	
0307	011324	012602				MOV	(SP)+,R2	::POP STACK INTO R2	
0308	011326	012601				MOV	(SP)+,R1	::POP STACK INTO R1	
0309	011330	012600				MOV	(SP)+,R0	::POP STACK INTO R0	
0310	011332	012737	011214	000024		MOV	#\$PWRDN,2#PWRVEC	::SET UP THE POWER DOWN VECTOR	
0311	011340	012737	000340	000026		MOV	#340,2#PWRVEC+2	::PRIO:7	
0312	011346	104400				TYPE		::REPORT THE POWER FAILURE	
0313	011350	013222			\$PWRMG:	.WORD	PWRMSG	::POWER FAIL MESSAGE POINTER	
0314	011352	012716				MOV	(PC)+,(SP)	::RESTART AT START1	
0315	011354	001616			\$PWRAD:	.WORD	START1	::RESTART ADDRESS	
0316	011356	000002				RTI			
0317	011360	000000			\$ILLUP:	HALT		::THE POWER UP SEQUENCE WAS STARTED	
0318	011362	000776				BR	.-2	::BEFORE THE POWER DOWN WAS COMPLETE	
0319	011364	000000			\$\$SAVR6:	0		::PUT THE SP HERE	

SBTTL VTO1 INTENSITY POINTERS & DATA
;TABLE OF POINTERS TO X AND Y INTENSITY PATTERNS FOR VTO1 DISPLAY

2320		
2321		
2322		
2323		
2324	011366	011466
2325	011370	011467
2326	011372	011471
2327	011374	011474
2328	011376	011500
2329	011400	011505
2330	011402	011513
2331	011404	011522
2332	011406	011532
2333	011410	011543
2334	011412	011555
2335	011414	011570
2336	011416	011604
2337	011420	011621
2338	011422	011637
2339	011424	011656
2340	011426	011676
2341	011430	011717
2342	011432	011741
2343	011434	011764
2344	011436	012010
2345	011440	012035
2346	011442	012063
2347	011444	012112
2348	011446	012142
2349	011450	012173
2350	011452	012225
2351	011454	012260
2352	011456	012314
2353	011460	012351
2354	011462	012407
2355	011464	012446
2356		
2357		
2358		
2359		
2360		
2361	011466	033
2362	011467	016
2363	011471	026
2364	011474	021
2365	011477	062
2366	011500	005
2367	011503	057
2368	011505	024
2369	011510	047
2370	011513	002
2371	011516	043
2372	011521	065
2373	011522	005
2374	011525	037
2375	011530	070

LEVTAB: L1
L2
L3
L4
L5
L6
L7
L8
L9
L10
L11
L12
L13
L14
L15
L16
L17
L18
L19
L20
L21
L22
L23
L24
L25
L26
L27
L28
L29
L30
L31
L32

;THIS IS THE CODED MATRIX TABLE FOR INTENSITY DISPLAY
;THE ENTRIES ARE PACKED 2X,Y COORDINATES TO A WORD
;THE EVEN BYTE CONTAINS THE FIRST PAIR,3 BITS PER
;COORDINATE.

2361	011466	033			L1: .BYTE 33
2362	011467	016	062		L2: .BYTE 16,62
2363	011471	026	032	064	L3: .BYTE 26,32,64
2364	011474	021	015	056	L4: .BYTE 21,15,56,62
2365	011477	062			
2366	011500	005	020	043	L5: .BYTE 05,20,43,57,71
2367	011503	057	071		
2368	011505	024	000	032	L6: .BYTE 24,00,32,47,65,51
2369	011510	047	065	051	
2370	011513	002	015	020	L7: .BYTE 02,15,20,43,46,61,65
2371	011516	043	046	061	
2372	011521	065			
2373	011522	005	012	024	L8: .BYTE 05,12,24,37,41,56,70,63
2374	011525	037	041	056	
2375	011530	070	063		

000	013	016	L9:	.BYTE 00,13,16,31,35,43,57,61,74
001	035	043		
002	061	074		
000	013	016	L10:	.BYTE 00,13,16,31,43,55,57,61,74,35
001	043	055		
002	061	074		
000	015	027	L11:	.BYTE 12,15,27,33,41,45,62,57,74,00,76
001	041	045		
002	057	074		
000	076	016	L12:	.BYTE 00,03,16,21,24,37,42,45,57,61,64,76
001	003	037		
002	024	057		
000	042	076	L13:	.BYTE 07,12,15,20,34,36,42,50,54,56,71,73,75
001	061	015		
002	007	036		
000	020	054		
001	042	073		
002	050	007	L14:	.BYTE 01,05,07,13,20,26,32,34,46,40,52,60,65,73
000	071	025		
001	005	046		
002	020	060		
000	034	027	L15:	.BYTE 03,06,27,11,25,30,33,45,47,50,54,52,71,74,66
001	040	030		
002	052	047		
000	073	052		
001	006	066	L16:	.BYTE 06,10,23,25,27,30,32,44,46,50,53,64,71,77,02,04
002	011	030		
000	025	046		
001	045	064		
002	054	002		
000	071	006	L17:	.BYTE 02,04,06,10,22,25,27,30,33
001	006	025		
002	027	033		
000	044	050		.BYTE 45,47,50,52,65,71,73,77
001	050	071		
002	077	006	L18:	.BYTE 02,14,06,10,22,25,27,30,33
000	004	025		
001	002	033		
002	010	050		.BYTE 45,47,50,52,64,66,71,73,77
000	027	066		
001	045	077		
002	064	006	L19:	.BYTE 02,04,06,10,22,25,27,30,34,41,43,46,50,57
000	071	025		
001	002	034		
002	004	046		
000	027	071		.BYTE 63,65,71,74,77
001	041	071		
002	050	005	L20:	.BYTE 01,03,05,07,12,30,23,64,25,42,27,34,36,51
000	063	071		
001	074	003		
002	001			

2432	0111767	007	012	030	
2433	0111772	023	064	025	
2434	0111775	042	027	034	
2435	0120000	036	051		
2436	0120002	055	057	070	.BYTE 55,57,70,53,66,72
2437	0120005	053	066	072	
2438	012010	001	003	005	L21: .BYTE 01,03,05,07,10,40,16,21,23,25,42,27,34,45
2439	012013	007	010	040	
2440	012016	016	021	023	
2441	012021	025	042	027	
2442	012024	034	045		
2443	012026	051	055	057	.BYTE 51,55,57,60,64,66,72
2444	012031	060	064	066	
2445	012034	072			
2446	012035	001	003	005	L22: .BYTE 01,03,05,07,10,40,16,21,23,25,42,27,34,46
2447	012040	007	010	040	
2448	012043	016	021	023	
2449	012046	025	042	027	
2450	012051	034	046		
2451	012052	051	053	055	.BYTE 51,53,55,57,60,64,66,72
2452	012056	057	060	064	
2453	012061	066	072		
2454	012063	000	002	005	L23: .BYTE 00,02,05,11,13,16,22,34,27,31,25,40,42,47
2455	012066	011	013	016	
2456	012071	022	034	027	
2457	012074	031	025	040	
2458	012077	042	047		
2459	012101	051	054	056	.BYTE 51,54,56,60,63,65,71,74,76
2460	012104	060	063	065	
2461	012107	071	074	076	
2462	012112	000	003	007	L24: .BYTE 00,03,07,12,14,16,21,23,25,27,32,34,30,43
2463	012115	012	014	016	
2464	012120	021	023	025	
2465	012123	027	032	034	
2466	012126	030	043		
2467	012130	046	050	052	.BYTE 46,50,52,55,63,65,67,71,74,76
2468	012133	055	063	065	
2469	012136	067	071	074	
2470	012141	076			
2471	012142	003	005	007	L25: .BYTE 03,05,07,10,12,14,16,21,23,27,32,34,36,41
2472	012145	010	012	014	
2473	012150	016	021	023	
2474	012152	027	032	034	
2475	012156	036	041		
2476	012160	045	047	050	.BYTE 45,47,50,52,54,63,65,67,70,72,76
2477	012163	052	054	063	
2478	012166	065	067	070	
2479	012171	072	076		
2480	012173	003	005	007	L26: .BYTE 03,05,07,10,12,16,21,23,25,30,32,34,36,41
2481	012176	010	012	016	
2482	012201	021	023	025	
2483	012204	030	032	034	
2484	012207	036	041		
2485	012211	045	047	052	.BYTE 45,47,52,54,56,61,63,67,70,72,74,76
2486	012214	054	056	061	
2487	012217	063	067	070	

2488	012222	072	074	076		
2489	012225	001	003	007	L27:	.BYTE 01,03,07,12,14,16,21,23,25,27,30,32,34,36
2490	012230	012	014	016		
2491	012233	021	023	025		
2492	012236	027	030	032		
2493	012241	034	036			
2494	012243	041	045	047		.BYTE 41,45,47,50,52,54,61,63,65,67,70,74,76
2495	012246	050	052	054		
2496	012251	061	063	065		
2497	012254	067	070	074		
2498	012257	076				
2499	012260	003	005	007	L28:	.BYTE 03,05,07,10,12,14,16,21,23,27,30,32,34,36
2500	012263	010	012	014		
2501	012266	016	021	023		
2502	012271	027	030	032		
2503	012274	034	036			
2504	012276	043	045	047		.BYTE 43,45,47,50,52,54,56,61,63,65,67,70,72,76
2505	012301	050	052	054		
2506	012304	056	061	063		
2507	012307	065	067	070		
2508	012312	072	076			
2509	012314	001	003	005	L29:	.BYTE 01,03,05,07,10,12,16,14,21,23,27,30,32,34,36,43
2510	012317	007	010	012		
2511	012322	016	014	021		
2512	012325	023	027	030		
2513	012330	032	034	036		
2514	012333	043				
2515	012334	045	047	050		.BYTE 45,47,50,52,54,56,61,63,67,70,72,74,76
2516	012337	052	054	056		
2517	012342	061	063	067		
2518	012345	070	072	074		
2519	012350	076				
2520	012351	001	003	005	L30:	.BYTE 01,03,05,07,10,12,14,16,21,23,25,27,30,32,36,41
2521	012354	007	010	012		
2522	012357	014	016	021		
2523	012362	023	025	027		
2524	012365	030	032	036		
2525	012370	041				
2526	012371	043	045	047		.BYTE 43,45,47,50,54,56,61,63,65,67,70,72,74,76
2527	012374	050	054	056		
2528	012377	061	063	065		
2529	012402	067	070	072		
2530	012405	074	076			
2531	012407	001	003	005	L31:	.BYTE 01,03,05,07,10,12,14,16,21,23,25,27,30,32,36,41
2532	012412	007	010	012		
2533	012415	014	016	021		
2534	012420	023	025	027		
2535	012423	030	032	036		
2536	012426	041				
2537	012427	043	045	047		.BYTE 43,45,47,50,52,54,56,61,63,65,67,70,72,74,76
2538	012432	050	052	054		
2539	012435	056	061	063		
2540	012440	065	067	070		
2541	012443	072	074	076		
2542	012446	001	003	005	L32:	.BYTE 01,03,05,07,10,12,14,16,21,23,25,27,30,32,34,36
2543	012451	007	010	012		

2544	012454	014	016	021
2545	012457	023	025	027
2546	012462	030	032	034
2547	012465	036		
2548	012466	041	043	045
2549	012471	047	050	052
2550	012474	054	056	061
2551	012477	063	065	067
2552	012502	070	072	074
2553	012505	076		
2554				

.BYTE 41,43,45,47,50,52,54,56,61,63,65,67,70,72,74,76

.EVEN

```

2555
2556
2557 000000
2558 012506 012536
2559 012510 000000
2560 012512 012561
2561 012514 000000
2562 012516 012604
2563 012520 000000
2564 012522 012617
2565 012524 000000
2566 012526 012636
2567 012530 000000
2568 012532 012645
2569 012534 000000
2570 012536 005015 047514 042527
2571 012544 020122 044124 042522
2572 012552 044123 046117 020104
2573 012560 200
2574 012561 015 052412 050120
2575 012566 051105 052040 051110
2576 012574 051505 047510 042114
2577 012602 100040
2578 012604 005015 020132 047503
2579 012612 047125 036524 200
2580 012617 040 040515 051124
2581 012624 054111 041440 052517
2582 012632 052116 100075
2583 012636 020040 047532 046517
2584 012644 200
2585 012645 040 026502 040507
2586 012652 046515 100101
2587
2588 000200

```

```

.SBTTL DISPLAY MESSAGES
;THIS IS THE MESSAGE FOR THE BOTTOM OF THE SCOPE
TERM=0
MESS1: W0001
TERM
MESS2: W0002
0
MESS3: W0003
0
MESS4: W0004
0
MESS5: W0005
0
MESS6: W0006
0
W0001: .ASCII <15><12>/LOWER THRESHOLD /<END>
W0002: .ASCII <15><12>/UPPER THRESHOLD /<END>
W0003: .ASCII <15><12>/Z COUNT=/<END>
W0004: .ASCII / MATRIX COUNT=/<END>
W0005: .ASCII / ZOOM/<END>
W0006: .ASCII / B-GAMMA/<END>
.EVEN
END=200 ;MSG TERMINATOR FOR DISPLAY CHARS

```

J

013306	010108
013307	010109
013308	010110
013309	010111
013310	010112
013311	010113
013312	010114
013313	010115
013314	010116
013315	010117
013316	010118
013317	010119
013318	010120
013319	010121
013320	010122
013321	010123
013322	010124
013323	010125
013324	010126
013325	010127
013326	010128
013327	010129
013328	010130
013329	010131
013330	010132
013331	010133
013332	010134
013333	010135
013334	010136
013335	010137
013336	010138
013337	010139
013338	010140
013339	010141
013340	010142
013341	010143
013342	010144
013343	010145
013344	010146
013345	010147
013346	010148
013347	010149
013348	010150
013349	010151
013350	010152
013351	010153
013352	010154
013353	010155
013354	010156
013355	010157
013356	010158
013357	010159
013358	010160
013359	010161
013360	010162

10102 :7
 77010 :8
 43056 :8
 35056 :8
 41016 :9
 35077 :9

.SBTTL ASCII MESSAGES

PWRMSG: .ASCIZ <15><12><12>/RESTARTED AFTER POWER FAILURE/

MSG1: .ASCIZ <15><12><12>/MD-11-DZNCB-C GAMMA 11 EXERCISER/

MSG2: .ASCIZ <15><12>/ENTER THRESHOLD VALUE IN OCTAL - THEN RETURN =/

MSG3: .ASCIZ <15><12>/BUS TIMEOUT ER - VSV01 DISPLAY/

MSG4: .ASCIZ <15><12>/BUS TIMEOUT ER - VTO1 DISPLAY/

MSG5: .ASCIZ <15><12>/BUS TIMEOUT ER - NC11/

MSG6: .ASCIZ <15><12>/ENTER KEYBOARD COMMAND(S)/

MSG7: .ASCIZ <15><12>/BUS TIMEOUT ER - AR11/

E05

MACY11-21-0200-C GAMMA 11 EXERCISE MACY11 27(732) 21-SEP-76 15:32 PAGE 56
MACY11-21-0200-C GAMMA 11 EXERCISE MACY11 27(732) 21-SEP-76 15:32 PAGE 56

000000
000000
000000

000000
000000
000000

000000
000000
000000

051101 000461

.END

SW8 = 000400	465#														
SW9 = 001000	464#														
TABLE 012656	1616	2593#													
TABLEX 001340	685#	748#	835*	837*	987	988	1008	1133	1145	1490					
TBITVE = 000014	506#														
TEMCHR 005460	1596*	1608*	1609*	1610*	1611*	1612	1620#								
TEMPO 001246	656#	1130*	1136*	1259*	1260*	1268*	1271*	1272*	1273	1274	1291*	1298*	1299*		
	1308*	1309	1310	1390	1434*	1436*	1477*	1478	1801	1915*					
TEMP1 001250	657#	1294*	1299	1435*	1438*										
TEMP2 001252	658#	1297*	1301												
TERM = 000000	2557#	2559													
THHI 001314	675#	750*	898*	991	1017	1121									
THLO 001312	674#	749*	972*	1000	1019	1117									
TIME 001344	687#	848													
TIXBNO 006576	1661	1663#													
TIXDIS 006532	1617	1654#													
TIXDON 006646	1672	1675#													
TIXL 006566	1660#	1668	1674												
TIXWOK 006606	1664	1666#													
TJOY 004146	864	868	1162#												
TKVEC = 000060	513#	751*	752*												
TOTSIZ 001322	678#	986	1130												
TPVEC = 000064	514#														
TRAPVE = 000034	512#	703*	704*												
TRTEC = 000014	507#														
TTYOUT 001310	673#	830*	1256*	1544*	1546*	1549									
TYPCR 006154	832	1544#													
TYPE = 104400	723	729	729	741	754	865	870	881	896	904	912	1891	1892		
	1895	1908	1919	1938	1991	1997	2002	2006	2011	2012	2014	2017	2021		
	2073	2170	2236	2238	2266#	2312									
	831	1257	1545	1547#	1548										
TYPO 006174	1894	2267#													
TYPOC = 104401	2269#														
TYPON = 104403	2268#														
TYPOS = 104402	668#	981*	1065*												
VERT 001276	613#	1365													
VTADR 001172	645#	1112*	1281*	1317*	1341	1366	1372*	1433*	1464						
VTCSR 001234	1592	1595#													
VTEXCP 006340	1598	1605#													
VTEXCT 006374	609#	1335													
VTMADR 001170	1604	1613#													
VTNOSP 006434	1578#	1582													
VTPL 006256	605#	1329													
VTVADR 001166	632#	1478*													
VTVCHP 001216	631#	735	1330	1344	1445*	1475	1479*	1526	1528*						
VTVCRG 001214	637#	878	909	954*	1191*	1333	1345	1441*	1442	1444*	1452*	1454*	1455		
VTVCSR 001222	1458#														
VTVINT 001232	641#	1347*													
VTVMAP 001224	638#	1453*													
VTVPOS 001220	633#	1109*													
VTVPX 001226	639#	1457*													
VTVPX1 001230	640#														
VTVSAV 001262	662#	954	1191												
VTWDON 006456	1579	1619#													
VTWL 006264	1581#	1586	1590	1594	1618										
VTWRIT 006246	1115	1119	1123	1128	1143	1147	1576#	1695							

VTXDAC	001236	646#	1468*						
VTYDAC	001240	647#	1469*						
W0001	012536	2558	2570#						
W0002	012561	2560	2574#						
W0003	012604	2562	2578#						
W0004	012617	2564	2580#						
W0005	012636	2566	2583#						
W0006	012645	2568	2585#						
XREF	001300	669#	982*	1054	1059*	1063*			
XYAVE	007210	1261	1300	1800#					
XYBUF	007336	1806	1808	1811	1833#	1867			
XYBUFE=	007436	1804	1824	1866#					
XYBUP	007436	1802	1807*	1867#					
YREF	001302	670#	993*	1055	1064*				
ZESUP	006730	1717#	1718*	1737	1739*				
\$AUTOB	001134	559#	1889	2037					
\$BDADR	001122	554#							
\$BDDAT	001126	556#							
\$CHARC	010556	2075*	2085*	2092	2101*	2106#			
\$CKSWR	007440	1890#	2273						
\$CMTAG	001100	542#	696	697					
\$CM3 =	000000	572#							
\$CNTLG	010312	1891	2032#						
\$CNTLU	010305	1908	2006	2031#					
\$CRLF	001161	573#	1919	2011	2031	2074	2109	2241	
\$ERFLG	001103	545#							
\$ERMAX	001115	551#							
\$ERRPC	001116	552#							
\$ERRTB	001164	590#							
\$ERTTL	001112	549#							
\$FILLC	001156	570#	2078	2109					
\$FILLS	001155	569#	2109						
\$GDADR	001120	553#							
\$GDDAT	001124	555#							
\$GTSWR	007510	1892#	2271						
\$HD =	000003	404	405						
\$HIOCT	011146	2229*	2240#						
\$ICNT	001104	546#							
\$ILLUP	011360	2282	2298	2317#					
\$INTAG	001135	560#	1920	2037					
\$ITEMB	001114	550#							
\$LF	001162	574#	2021	2031	2109	2241			
\$LPADR	001106	547#							
\$LPERR	001110	548#							
\$MAIL =	***** U	723	2062						
\$MNEW	010330	1895	2035#						
\$MSWR	010317	1892	2033#						
\$NULL	001154	568#	2080	2109					
\$OCNT	011004	2142*	2171*	2184#					
\$OMODE	011006	2137*	2141*	2146	2149*	2160*	2186#		
\$PASS	001100	543#							
\$PWRAD	011354	2315#							
\$PWDRN	011214	705	2282#	2310					
\$PWRMG	011350	2313#							
\$PWRUP	011266	2292	2298#						
\$QUES	001160	572#	1938	2014	2031	2109	2238	2241	

CROSS REFERENCE TABLE -- MACRO NAMES

MACRO NAME	2268	2269	2271	2273	2274	2275	2276							
MACRO NAME	1323	1323	1328	1350	1364	1382	1385	1401	1403	1409	1411	1417	1419	1419
MACRO NAME	1449	1453	1461	1463	1472	1474	1482	1488	1503	1506	1523	1527	1528	1533
MACRO NAME	1797	1799	1809	1872	1875	1893	1972	2038	2041	2066	2112	2197	2198	2198
MACRO NAME	2268	2269	2271	2273	2274	2275	2276							

CROSS REFERENCE TABLE -- PERMANENT SYMBOLS

11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
11000	11001	11002	11003	11004	11005	11006	11007	11008	11009	11010	11011	11012	11013	11014
11015	11016	11017	11018	11019	11020	11021	11022	11023	11024	11025	11026	11027	11028	11029
11030	11031	11032	11033	11034	11035	11036	11037	11038	11039	11040	11041	11042	11043	11044
11045	11046	11047	11048	11049	11050	11051	11052	11053	11054	11055	11056	11057	11058	11059
11060	11061	11062	11063	11064	11065	11066	11067	11068	11069	11070	11071	11072	11073	11074
11075	11076	11077	11078	11079	11080	11081	11082	11083	11084	11085	11086	11087	11088	11089
11090	11091	11092	11093	11094	11095	11096	11097	11098	11099	11100	11101	11102	11103	11104
11105	11106	11107	11108	11109	11110	11111	11112	11113	11114	11115	11116	11117	11118	11119
11120	11121	11122	11123	11124	11125	11126	11127	11128	11129	11130	11131	11132	11133	11134
11135	11136	11137	11138	11139	11140	11141	11142	11143	11144	11145	11146	11147	11148	11149
11150	11151	11152	11153	11154	11155	11156	11157	11158	11159	11160	11161	11162	11163	11164
11165	11166	11167	11168	11169	11170	11171	11172	11173	11174	11175	11176	11177	11178	11179
11180	11181	11182	11183	11184	11185	11186	11187	11188	11189	11190	11191	11192	11193	11194
11195	11196	11197	11198	11199	11200	11201	11202	11203	11204	11205	11206	11207	11208	11209
11210	11211	11212	11213	11214	11215	11216	11217	11218	11219	11220	11221	11222	11223	11224
11225	11226	11227	11228	11229	11230	11231	11232	11233	11234	11235	11236	11237	11238	11239
11240	11241	11242	11243	11244	11245	11246	11247	11248	11249	11250	11251	11252	11253	11254
11255	11256	11257	11258	11259	11260	11261	11262	11263	11264	11265	11266	11267	11268	11269
11270	11271	11272	11273	11274	11275	11276	11277	11278	11279	11280	11281	11282	11283	11284
11285	11286	11287	11288	11289	11290	11291	11292	11293	11294	11295	11296	11297	11298	11299
11300	11301	11302	11303	11304	11305	11306	11307	11308	11309	11310	11311	11312	11313	11314
11315	11316	11317	11318	11319	11320	11321	11322	11323	11324	11325	11326	11327	11328	11329
11330	11331	11332	11333	11334	11335	11336	11337	11338	11339	11340	11341	11342	11343	11344
11345	11346	11347	11348	11349	11350	11351	11352	11353	11354	11355	11356	11357	11358	11359
11360	11361	11362	11363	11364	11365	11366	11367	11368	11369	11370	11371	11372	11373	11374
11375	11376	11377	11378	11379	11380	11381	11382	11383	11384	11385	11386	11387	11388	11389
11390	11391	11392	11393	11394	11395	11396	11397	11398	11399	11400	11401	11402	11403	11404
11405	11406	11407	11408	11409	11410	11411	11412	11413	11414	11415	11416	11417	11418	11419
11420	11421	11422	11423	11424	11425	11426	11427	11428	11429	11430	11431	11432	11433	11434
11435	11436	11437	11438	11439	11440	11441	11442	11443	11444	11445	11446	11447	11448	11449
11450	11451	11452	11453	11454	11455	11456	11457	11458	11459	11460	11461	11462	11463	11464
11465	11466	11467	11468	11469	11470	11471	11472	11473	11474	11475	11476	11477	11478	11479
11480	11481	11482	11483	11484	11485	11486	11487	11488	11489	11490	11491	11492	11493	11494
11495	11496	11497	11498	11499	11500	11501	11502	11503	11504	11505	11506	11507	11508	11509
11510	11511	11512	11513	11514	11515	11516	11517	11518	11519	11520	11521	11522	11523	11524
11525	11526	11527	11528	11529	11530	11531	11532	11533	11534	11535	11536	11537	11538	11539
11540	11541	11542	11543	11544	11545	11546	11547	11548	11549	11550	11551	11552	11553	11554
11555	11556	11557	11558	11559	11560	11561	11562	11563	11564	11565	11566	11567	11568	11569
11570	11571	11572	11573	11574	11575	11576	11577	11578	11579	11580	11581	11582	11583	11584
11585	11586	11587	11588	11589	11590	11591	11592	11593	11594	11595	11596	11597	11598	11599
11600	11601	11602	11603	11604	11605	11606	11607	11608	11609	11610	11611	11612	11613	11614
11615	11616	11617	11618	11619	11620	11621	11622	11623	11624	11625	11626	11627	11628	11629
11630	11631	11632	11633	11634	11635	11636	11637	11638	11639	11640	11641	11642	11643	11644
11645	11646	11647	11648	11649	11650	11651	11652	11653	11654	11655	11656	11657	11658	11659
11660	11661	11662	11663	11664	11665	11666	11667	11668	11669	11670	11671	11672	11673	11674
11675	11676	11677	11678	11679	11680	11681	11682	11683	11684	11685	11686	11687	11688	11689
11690	11691	11692	11693	11694	11695	11696	11697	11698	11699	11700	11701	11702	11703	11704
11705	11706	11707	11708	11709	11710	11711	11712	11713	11714	11715	11716	11717	11718	11719
11720	11721	11722	11723	11724	11725	11726	11727	11728	11729	11730	11731	11732	11733	11734
11735	11736	11737	11738	11739	11740	11741	11742	11743	11744	11745	11746	11747	11748	11749
11750	11751	11752	11753	11754	11755	11756	11757	11758	11759	11760	11761	11762	11763	11764
11765	11766	11767	11768	11769	11770	11771	11772	11773	11774	11775	11776	11777	11778	11779
11780	11781	11782	11783	11784	11785	11786	11787	11788	11789	11790	11791	11792	11793	11794
11795	11796	11797	11798	11799	11800	11801	11802	11803	11804	11805	11806	11807	11808	11809
11810	11811	11812	11813	11814	11815	11816	11817	11818	11819	11820	11821	11822	11823	11824
11825	11826	11827	11828	11829	11830	11831	11832	11833	11834	11835	11836	11837	11838	11839
11840	11841	11842	11843	11844	11845	11846	11847	11848	11849	11850	11851	11852	11853	11854
11855	11856	11857	11858	11859	11860	11861	11862	11863	11864	11865	11866	11867	11868	11869
11870	11871	11872	11873	11874	11875	11876	11877	11878	11879	11880	11881	11882	11883	11884
11885	11886	11887	11888	11889	11890	11891	11892	11893	11894	11895	11896	11897	11898	11899
11900	11901	11902	11903	11904	11905	11906	11907	11908	11909	11910	11911	11912	11913	11914
11915	11916	11917	11918	11919	11920	11921	11922	11923	11924	11925	11926	11927	11928	11929
11930	11931	11932	11933	11934	11935	11936	11937	11938	11939	11940	11941	11942	11943	11944
11945	11946	11947	11948	11949	11950	11951	11952	11953	11954	11955	11956	11957	11958	11959
11960	11961	11962	11963	11964	11965	11966	11967	11968	11969	11970	11971	11972	11973	11974
11975	11976	11977	11978	11979	11980	11981	11982	11983	11984	11985	11986	11987	11988	11989
11990	11991	11992	11993	11994	11995	11996	11997	11998	11999	12000	12001	12002	12003	12004

* DZNCB.SFO.SOL / CAP-DZNCB.P11
COUNT: 11137 200 501 1 00000000
COUNT: 11137 200 501 1 00000000
COUNT: 11137 200 501 1 00000000

2

