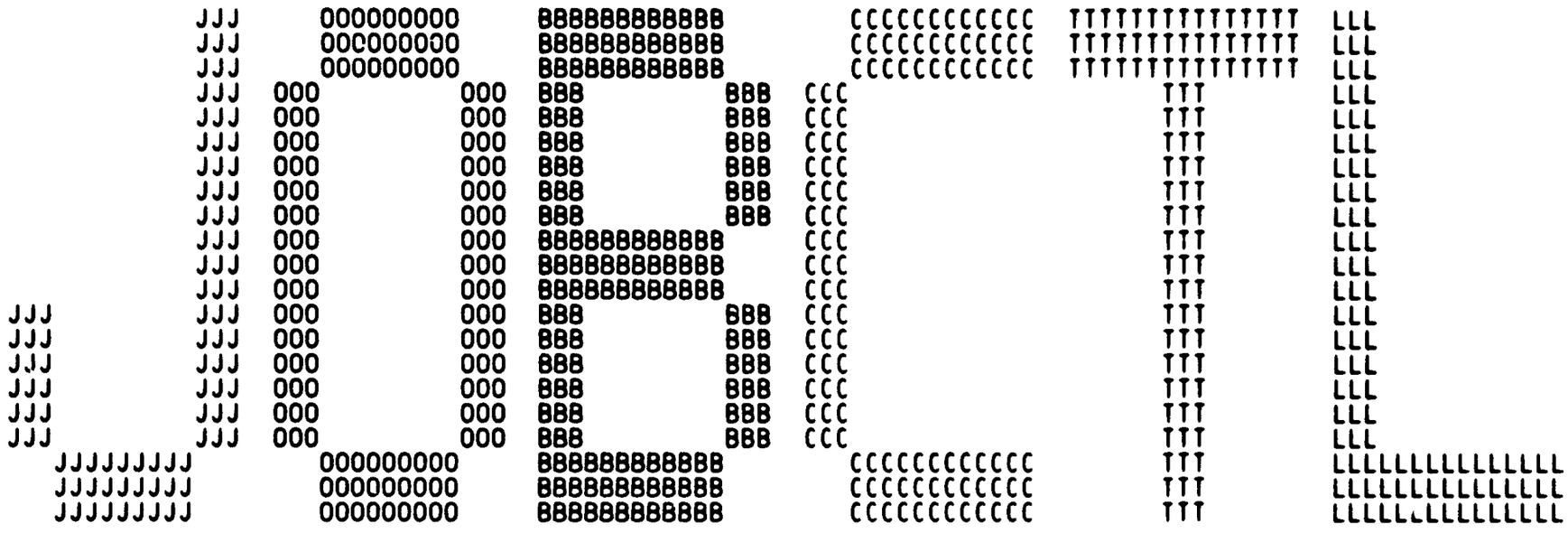
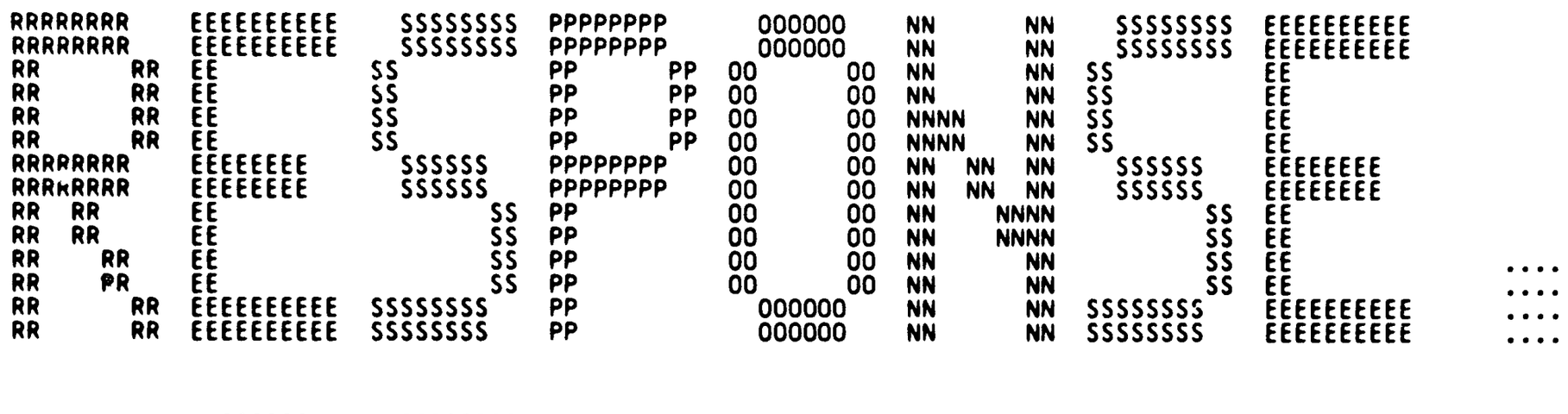


_$2

Val

0000
0000
0000
0000
0000
0000
0000
0000
0000
0000
0000
0000
0000
0000
0000
314
485
7FFF
7FFF
7FFF
7FFF
7FFF
7FFF
7FFF
7FFF
7FFF
7FFF
7FFF
7FFF
7FFF
7FFF
7FFF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF


RRRRRRRR EEEEEEEEEE SSSSSSSS PPPPPPPP 000000 NN NN SSSSSSSS EEEEEEEEEE
RRRRRRRR EEEEEEEEEE SSSSSSSS PPPPPPPP 000000 NN NN SSSSSSSS EEEEEEEEEE
RR RR EE SS PP PP 00 00 NN NN SS EE
RR RR EE SS PP PP 00 00 NN NN SS EE
RR RR EE SS PP PP 00 00 NNNN NN SS EE
RR RR EE SS PP PP 00 00 NNNN NN SS EE
RRRRRRRR EEEEEEEE SSSSSS PPPPPPPP 00 00 NN NN NN SSSSSS EEEEEEEE
RRRRRRRR EEEEEEEE SSSSSS PPPPPPPP 00 00 NN NN NN SSSSSS EEEEEEEE
RR RR EE SS PP 00 00 NN NNNN SS EE
RR RR EE SS PP 00 00 NN NNNN SS EE
RR RR EE SS PP 00 00 NN NN SS EE
RR RR EE SS PP 00 00 NN NN SS EE
RR RR EEEEEEEEEE SSSSSSSS PP 000000 NN NN SSSSSSSS EEEEEEEEEE
RR RR EEEEEEEEEE SSSSSSSS PP 000000 NN NN SSSSSSSS EEEEEEEEEE
LL II SSSSSSSS
LL II SSSSSSSS
LL II SS
LL II SS
LL II SS
LL II SS
LL II SSSSSS
LL II SSSSSS
LL II SS
LL II SS
LL II SS
LL II SS
LLLLLLLLLL IIIIII SSSSSSSS
LLLLLLLLLL IIIIII SSSSSSSS

```
;     1        0001  0 MODULE RESPONSF (%TITLE 'Service response message'
;     2        0002  0                   IDENT = 'V04-000'
;     3        0003  0                   ) =
;     4        0004  1 BEGIN
;     5        0005  1
;     6        0006  1 !
;     7        0007  1 !**********************************************************************
;     8        0008  1 !*                                                                    *
;     9        0009  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                          *
;    10        0010  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.           *
;    11        0011  1 !*  ALL RIGHTS RESERVED.                                            *
;    12        0012  1 !*                                                                    *
;    13        0013  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED  *
;    14        0014  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
;    15        0015  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER  *
;    16        0016  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY  *
;    17        0017  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY  *
;    18        0018  1 !*  TRANSFERRED.                                                       *
;    19        0019  1 !*                                                                    *
;    20        0020  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE  *
;    21        0021  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT  *
;    22        0022  1 !*  CORPORATION.                                                       *
;    23        0023  1 !*                                                                    *
;    24        0024  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS  *
;    25        0025  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.            *
;    26        0026  1 !*                                                                    *
;    27        0027  1 !*                                                                    *
;    28        0028  1 !**********************************************************************
;    29        0029  1 !
;    30        0030  1
;    31        0031  1 !++
;    32        0032  1 ! FACILITY:
;    33        0033  1 !       Job controller.
;    34        0034  1 !
;    35        0035  1 ! ABSTRACT:
;    36        0036  1 !       This module contains the routines that send a response to a caller of
;    37        0037  1 !       $SNDACC, $SNDSMB, or $SNDJBC.
;    38        0038  1 !
;    39        0039  1 ! ENVIRONMENT:
;    40        0040  1 !       VAX/VMS user and kernel mode.
;    41        0041  1 !--
;    42        0042  1 !
;    43        0043  1 ! AUTHOR: M. Jack, CREATION DATE: 16-Feb-1982
;    44        0044  1 !
;    45        0045  1 ! MODIFIED BY:
;    46        0046  1 !
;    47        0047  1 !       V03-006 JAK0200         J A Krycka      15-Mar-1984
;    48        0048  1 !               Add IO$M_NORSWAIT function modifier to mailbox write.
;    49        0049  1 !
;    50        0050  1 !       V03-005 MLJ0115         Martin L. Jack, 30-Jul-1983  15:01
;    51        0051  1 !               Changes for job controller baselevel.
;    52        0052  1 !
;    53        0053  1 !       V03-004 MLJ0114         Martin L. Jack, 23-Jun-1983  5:02
;    54        0054  1 !               Changes for job controller baselevel.
;    55        0055  1 !
;    56        0056  1 !       V03-003 MLJ0112         Martin L. Jack, 29-Apr-1983  3:07
;    57        0057  1 !               Changes for job controller baselevel.
```

```
;    58          0058  1 !
;    59          0059  1 !       V03-002 MLJ0109         Martin L. Jack, 14-Apr-1983  12:48
;    60          0060  1 !               Changes for job controller baselevel.
;    61          0061  1 !
;    62          0062  1 !       V03-001 CWH1002         CW Hobbs                     1-Mar-1983
;    63          0063  1 !               Convert the extended pid to an internal pid when we make
;    64          0064  1 !               the ast control block.
;    65          0065  1 !**
```

```
;    67          0066  1 REQUIRE 'SRC$:JOBCTLDEF';
;    68          1107  1
;    69          1108  1
;    70          1109  1 FORWARD ROUTINE
;    71          1110  1         CREATE_SRB:                         NOVALUE,
;    72          1111  1         LOCATE_SRB_OUTPUT_ITEM,
;    73          1112  1         COMPLETE_SRB_OUTPUT_ITEM:           NOVALUE,
;    74          1113  1         SEND_SERVICE_RESPONSE_KERNEL,
;    75          1114  1         SEND_SERVICE_RESPONSE_MESSAGE:      NOVALUE,
;    76          1115  1         SEND_SERVICE_RESPONSE:              NOVALUE;
;    77          1116  1
;    78          1117  1
;    79          1118  1 LINKAGE
;    80          1119  1         EXE$ALLOCBUF_LINKAGE =
;    81          1120  1             JSB(REGISTER=1; REGISTER=2):
;    82          1121  1             NOPRESERVE(3, 4) PRESERVE(5) NOTUSED(6, 7, 8, 9, 10, 11),
;    83          1122  1
;    84          1123  1         EXE$EPID_LINKAGE =
;    85          1124  1             JSB(REGISTER=0):
;    86          1125  1             PRESERVE(1, 2, 3, 4, 5) NOTUSED(6, 7, 8, 9, 10, 11),
;    87          1126  1
;    88          1127  1         SCH$QAST_LINKAGE =
;    89          1128  1             JSB(REGISTER=0, REGISTER=5):
;    90          1129  1             NOPRESERVE(1, 2, 3, 4, 5) NOTUSED(6, 7, 8, 9, 10, 11);
;    91          1130  1
;    92          1131  1
;    93          1132  1 EXTERNAL ROUTINE
;    94          1133  1         EXE$ALLOCBUF:   EXE$ALLOCBUF_LINKAGE    ADDRESSING_MODE(GENERAL),
;    95          1134  1         EXE$EPID_TO_IPID: EXE$EPID_LINKAGE      ADDRESSING_MODE(GENERAL),
;    96          1135  1         EXE$JBCRSP:                             ADDRESSING_MODE(GENERAL),
;    97          1136  1         SCH$QAST:       SCH$QAST_LINKAGE        ADDRESSING_MODE(GENERAL);
;    98          1137  1
;    99          1138  1
;   100          1139  1 ! Extension to AST control block.  These are also known to SYSSNDJBC.
;   101          1140  1 !
;   102          1141  1 MACRO
;   103          1142  1         ACB_L_IMGCNT=           28,0,32,0 %,    ! Image counter
;   104          1143  1         ACB_L_EFN=              32,0,32,0 %,    ! Event flag number
;   105          1144  1         ACB_L_IOSB=             36,0,32,0 %,    ! IOSB address
;   106          1145  1         ACB_L_STATUS=           40,0,32,0 %,    ! Status for IOSB
;   107          1146  1         ACB_W_ITEMCOUNT=        44,0,16,0 %,    ! Count of data items
;   108          1147  1         ACB_B_ITEMS=            46,0,0,0 %,     ! Base of item descriptors
;   109          1148  1
;   110          1149  1         ACI_W_SIZE=             0,0,16,0 %,     ! Size of return buffer
;   111          1150  1         ACI_W_DATASIZE=         2,0,16,0 %,     ! Size of data to return
;   112          1151  1         ACI_L_ADDRESS=          4,0,32,0 %,     ! Address of return buffer
;   113          1152  1         ACI_L_LENADDR=          8,0,32,0 %,     ! Address of length buffer
;   114          1153  1         ACI_T_DATA=             12,0,0,0 %;     ! Data to return
;   115          1154  1
;   116          1155  1
;   117          1156  1 LITERAL
;   118          1157  1         ACB_S_BUFFER=           46 + 12;        ! Size of minimal ACB buffer
;   119          1158  1
;   120          1159  1
;   121          1160  1
;   122          1161  1 ! Service response block.
;   123          1162  1 !
```

```
;  124     1163  1 MACRO
;  125     1164  1         SRB_W_TYPE=             0,0,16,0 %,     ! ACM$W_TYPE
;  126     1165  1         SRB_W_MAILBOX=          2,0,16,0 %,     ! ACM$W_MAILBOX
;  127     1166  1         SRB_Q_RSP=              4,0,0,0 %,      ! Quadword response message
;  128     1167  1         SRB_T_ACB=              12,0,0,0 %;     ! Extended AST control block
;  129     1168  1
;  130     1169  1
;  131     1170  1 LITERAL
;  132     1171  1         SRB_S_RSP=              8,
;  133     1172  1         SRB_S_BUFFER=           12 + ACB_S_BUFFER;
;  134     1173  1
;  135     1174  1
;  136     1175  1 BUILTIN
;  137     1176  1         MTPR;
```

```
;  139          1177  1 GLOBAL ROUTINE CREATE_SRB(SRB): NOVALUE=
;  140          1178  1
;  141          1179  1 !++
;  142          1180  1 !
;  143          1181  1 ! FUNCTIONAL DESCRIPTION:
;  144          1182  1 !       This routine creates a service response block from information in the
;  145          1183  1 !       mailbox message buffer, value buffers, and the response buffer.
;  146          1184  1 !
;  147          1185  1 ! INPUT PARAMETERS:
;  148          1186  1 !       SRB             - Pointer to SRB buffer.
;  149          1187  1 !
;  150          1188  1 ! IMPLICIT INPUTS:
;  151          1189  1 !       MBX             - Pointer to buffered mailbox message.
;  152          1190  1 !       RSP             - Response buffer.
;  153          1191  1 !
;  154          1192  1 ! OUTPUT PARAMETERS:
;  155          1193  1 !       NONE
;  156          1194  1 !
;  157          1195  1 ! IMPLICIT OUTPUTS:
;  158          1196  1 !       Service response block in SRB buffer.
;  159          1197  1 !
;  160          1198  1 ! ROUTINE VALUE:
;  161          1199  1 !       NONE
;  162          1200  1 !
;  163          1201  1 ! SIDE EFFECTS:
;  164          1202  1 !       NONE
;  165          1203  1 !
;  166          1204  1 !--
;  167          1205  1
;  168          1206  2 BEGIN
;  169          1207  2 MAP
;  170          1208  2         SRB:            REF BBLOCK;     ! Pointer to SRB buffer
;  171          1209  2
;  172          1210  2
;  173          1211  2 SRB[SRB_W_TYPE] = .MBX[ACM$W_TYPE];
;  174          1212  2 SRB[SRB_W_MAILBOX] = .MBX[ACM$W_MAILBOX];
;  175          1213  2 (SRB[SRB_Q_RSP])+0 = .RSP;
;  176          1214  2 (SRB[SRB_Q_RSP])+4 = 0;
;  177          1215  2 BBLOCK[SRB[SRB_T_ACB], ACB$L_ASTQFL] = 0;
;  178          1216  2 BBLOCK[SRB[SRB_T_ACB], ACB$L_ASTQBL] = 0;
;  179          1217  2 BBLOCK[SRB[SRB_T_ACB], ACB$W_SIZE] = $BYTEOFFSET(ACB_B_ITEMS);
;  180          1218  2 BBLOCK[SRB[SRB_T_ACB], ACB$B_TYPE] = DYN$C_ACB;
;  181          1219  2 BBLOCK[SRB[SRB_T_ACB], ACB$B_RMOD] = .MBX[ACM$B_RMOD] OR ACB$M_KAST;
;  182          1220  2 BBLOCK[SRB[SRB_T_ACB], ACB$L_PID] = EXE$EPID_TO_IPID(.MBX[ACM$L_PROCID]);
;  183          1221  2 BBLOCK[SRB[SRB_T_ACB], ACB$L_AST] = .MBX[ACM$L_ASTADR];
;  184          1222  2 IF .MBX[ACM$L_ASTADR] NEQ 0 THEN BBLOCK[SRB[SRB_T_ACB], ACB$V_QUOTA] = TRUE;
;  185          1223  2 BBLOCK[SRB[SRB_T_ACB], ACB$L_ASTPRM] = .MBX[ACM$L_ASTPRM];
;  186          1224  2 BBLOCK[SRB[SRB_T_ACB], ACB$L_KAST] = EXE$JBCRSP;
;  187          1225  2 BBLOCK[SRB[SRB_T_ACB], ACB_L_IMGCNT] = .MBX[ACM$L_IMAGECNT];
;  188          1226  2 BBLOCK[SRB[SRB_T_ACB], ACB_L_EFN] = .MBX[ACM$L_EFN];
;  189          1227  2 BBLOCK[SRB[SRB_T_ACB], ACB_L_IOSB] = .MBX[ACM$L_IOSB];
;  190          1228  2 BBLOCK[SRB[SRB_T_ACB], ACB_L_STATUS] = 0;
;  191          1229  2 BBLOCK[SRB[SRB_T_ACB], ACB_W_ITEMCOUNT] = 0;
;  192          1230  1 END;

                                                .TITLE  RESPONSE Service response message
```

```
                .IDENT  \V04-000\

                .PSECT  COMMON,NOEXE,  OVR,2

00000 DIAG_STORAGE_BASE:
                .BLKB   0
00000 DIAG_TRACE:
                .BLKB   96
00060 DIAG_COUNT:
                .BLKB   96
000C0 DIAG_FLAGS:
                .BLKB   4
000C4 WORK_AREA:
                .BLKB   44
000F0 SNDJBC_COUNT:
                .BLKB   132
00174 GETQUI_COUNT:
                .BLKB   40
0019C SNDACC_COUNT:
                .BLKB   28
001B8 SNDSMB_COUNT:
                .BLKB   72
00200 DIAG_STORAGE_END:
                .BLKB   0
00200 FLAGS:    .BLKB   4
00204 IMAGE_DUMP_STSFLG:
                .BLKB   4
00208 THIS_SYSID:
                .BLKB   6
0020E           .BLKB   2
00210 CUR_TIME:
                .BLKB   8
00218 HOURLY_TIME:
                .BLKB   8
00220 HOURLY_PARAMS:
                .BLKB   20
00234 SYMBIONT_COUNT:
                .BLKB   4
00238 QUEUE_REFERENCE_COUNT:
                .BLKB   4
0023C MBX_MESSAGE_COUNT:
                .BLKB   4
00240 MBX:      .BLKB   4
00244 MBX_END:  .BLKB   4
00248 MEMORY_FREE_QUEUES:
                .BLKB   40
00270 NONAST_WORK_QUEUE:
                .BLKB   8
00278 BCB_FREE_LIST:
                .BLKB   4
0027C BCB_ACTIVE_LIST:
                .BLKB   4
00280 GQL_FREE_LIST:
                .BLKB   4
00284 GQL_ACTIVE_LIST:
                .BLKB   4
00288 OPEN_GETQUI_LIST:
```

```
                        .BLKB   4
0028C PROCESS_DATA_LIST:
                        .BLKB   4
00290 SYMBIONT_CONTROL:
                        .BLKB   4
00294 SPARE_AREA:
                        .BLKB   12
002A0 REMOTE_REQUEST_LKSB:
                        .BLKB   8
002A8 QUEUE_FILE_LKSB:
                        .BLKB   8
002B0 QUEUE_LOCK_LKSB:
                        .BLKB   8
002B8 RSP:              .BLKB   8
002C0 JBC_PRIORITY:
                        .BLKB   4
002C4 JBC_PRIVILEGES:
                        .BLKB   8
002CC JBC_QUOTAS:
                        .BLKB   66
0030E                   .BLKB   2
00310 JBC_UIC:          .BLKB   4
00314 QUEUE_FAB:
                        .BLKB   80
00364 QUEUE_RAB:
                        .BLKB   68
003A8 QUEUE_NAM:
                        .BLKB   96
00408 QUEUE_XAB:
                        .BLKB   88
00460 QUEUE_RSA:
                        .BLKB   255
0055F                   .BLKB   1
00560 QUEUE_ALQ:
                        .BLKB   4
00564 QUEUE_MBF:
                        .BLKB   1
00565                   .BLKB   3
00568 ACCOUNTING_FABS:
                        .BLKB   8
00570 ACCOUNTING_RABS:
                        .BLKB   8
00578 ACCOUNT_FAB_A:
                        .BLRB   80
005C8 ACCOUNT_RAB_A:
                        .BLRB   68
0060C ACCOUNT_NAM_A:
                        .BLRB   96
0066C ACCOUNT_RSA_A:
                        .BLRB   255
0076B                   .BLKB   1
0076C ACCOUNT_FAB_B:
                        .BLKB   80
007BC ACCOUNT_RAB_B:
                        .BLRB   68
00800 ACCOUNT_NAM_B:
                        .BLRB   96
```

```
00860 ACCOUNT_RSA_B:
                .BLKB   255
0095F           .BLKB   1
00960 DIAG_FAB:
                .BLKB   80
009B0 DIAG_RAB:
                .BLKB   68
009F4 MBX_CHAN:
                .BLKB   4
009F8 MBX_IOSB:
                .BLKB   8
00A00 MBX_BUFFER:
                .BLKB   1024
00E00 VALUE_STORAGE_BASE:
                .BLKB   0
00E00 ITEM_PRESENT:
                .BLKB   32
00E20 VALUE_GETQUI_BASE:
                .BLKB   0
00E20 VALUE_ACCOUNTING_MESSAGE:
                .BLKB   6
00E26 VALUE_ACCOUNTING_TYPES:
                .BLKB   4
00E2A VALUE_AFTER_TIME:
                .BLKB   8
00E32 VALUE_ALIGNMENT_PAGES:
                .BLKB   1
00E33 VALUE_BASE_PRIORITY:
                .BLKB   1
00E34 VALUE_BATCH_INPUT:
                .BLKB   6
00E3A VALUE_BATCH_OUTPUT:
                .BLKB   10
00E44 VALUE_BUFFER_COUNT:
                .BLKB   1
00E45 VALUE_CHARACTERISTIC_NAME:
                .BLKB   6
00E4B VALUE_CHARACTERISTIC_NUMBER:
                .BLKB   1
00E4C VALUE_CHARACTERISTICS:
                .BLKB   16
00E5C VALUE_CHECKPOINT_DATA:
                .BLKB   6
00E62 VALUE_CLI:
                .BLKB   6
00E68 VALUE_CPU_DEFAULT:
                .BLKB   4
00E6C VALUE_CPU_LIMIT:
                .BLKB   4
00E70 VALUE_DESTINATION_QUEUE:
                .BLKB   8
00E78 VALUE_DEVICE_NAME:
                .BLKB   6
00E7E VALUE_ENTRY_NUMBER:
                .BLKB   4
00E82 VALUE_ENTRY_NUMBER_OUTPUT:
                .BLKB   10
```

```
00E8C VALUE_EXTEND_QUANTITY:
                .BLKB   2
00E8E VALUE_FILE_COPIES:
                .BLKB   1
00E8F VALUE_FILE_IDENTIFICATION:
                .BLKB   36
00EB3 VALUE_FILE_SETUP_MODULES:
                .BLKB   6
00EB9 VALUE_FILE_SPECIFICATION:
                .BLKB   6
00EBF VALUE_FIRST_PAGE:
                .BLKB   4
00EC3 VALUE_FORM_DESCRIPTION:
                .BLKB   6
00EC9 VALUE_FORM_LENGTH:
                .BLKB   1
00ECA VALUE_FORM_MARGIN_BOTTOM:
                .BLKB   1
00ECB VALUE_FORM_MARGIN_LEFT:
                .BLKB   2
00ECD VALUE_FORM_MARGIN_RIGHT:
                .BLKB   2
00ECF VALUE_FORM_MARGIN_TOP:
                .BLKB   1
00ED0 VALUE_FORM_NAME:
                .BLKB   6
00ED6 VALUE_FORM_NUMBER:
                .BLKB   4
00EDA VALUE_FORM:
                .BLKB   8
00EE2 VALUE_FORM_SETUP_MODULES:
                .BLKB   6
00EE8 VALUE_FORM_STOCK:
                .BLKB   6
00EEE VALUE_FORM_WIDTH:
                .BLKB   2
00EF0 VALUE_GENERIC_TARGET:
                .BLKB   996
012D4 VALUE_JOB_COPIES:
                .BLKB   1
012D5 VALUE_JOB_LIMIT:
                .BLKB   1
012D6 VALUE_JOB_NAME:
                .BLKB   6
012DC VALUE_JOB_RESET_MODULES:
                .BLKB   6
012E2 VALUE_JOB_SIZE_MAXIMUM:
                .BLKB   4
012E6 VALUE_JOB_SIZE_MINIMUM:
                .BLKB   4
012EA VALUE_JOB_STATUS_OUTPUT:
                .BLKB   10
012F4 VALUE_LAST_PAGE:
                .BLKB   4
012F8 VALUE_LIBRARY_SPECIFICATION:
                .BLKB   6
012FE VALUE_LOG_QUEUE:
```

```
                .BLKB   8
01306 VALUE_LOG_SPECIFICATION:
                .BLKB   6
0130C VALUE_NOTE:
                .BLKB   6
01312 VALUE_OPERATOR_REQUEST:
                .BLKB   6
01318 VALUE_OWNER_UIC:
                .BLKB   4
0131C VALUE_PAGE_SETUP_MODULES:
                .BLKB   6
01322 VALUE_PARAMETER_1:
                .BLKB   6
01328 VALUE_PARAMETER_2:
                .BLKB   6
0132E VALUE_PARAMETER_3:
                .BLKB   6
01334 VALUE_PARAMETER_4:
                .BLKB   6
0133A VALUE_PARAMETER_5:
                .BLKB   6
01340 VALUE_PARAMETER_6:
                .BLKB   6
01346 VALUE_PARAMETER_7:
                .BLKB   6
0134C VALUE_PARAMETER_8:
                .BLKB   6
01352 VALUE_PRIORITY:
                .BLKB   1
01353 VALUE_PROCESSOR:
                .BLKB   6
01359 VALUE_PROTECTION:
                .BLKB   4
0135D VALUE_QUEUE:
                .BLKB   6
01363 VALUE_QUEUE_FILE_SPECIFICATION:
                .BLKB   6
01369 VALUE_RELATIVE_PAGE:
                .BLKB   4
0136D VALUE_RESERVED_INPUT_1:
                .BLKB   1
0136E VALUE_RESERVED_INPUT_2:
                .BLKB   2
01370 VALUE_RESERVED_INPUT_3:
                .BLKB   4
01374 VALUE_RESERVED_INPUT_4:
                .BLKB   6
0137A VALUE_RESERVED_OUTPUT_1:
                .BLKB   10
01384 VALUE_RESERVED_OUTPUT_2:
                .BLKB   10
0138E VALUE_SEARCH_STRING:
                .BLKB   6
01394 VALUE_SCSNODE_NAME:
                .BLKB   6
0139A VALUE_WSDEFAULT:
                .BLKB   2
```

```
                                        0139C VALUE_WSEXTENT:
                                                .BLKB   2
                                        0139E VALUE_WSQUOTA:
                                                .BLKB   2
                                        013A0 VALUE_STORAGE_END:
                                                .BLKB   0

                                              JBC$_CLOSEOUT=      266328
                                              JBC$_NOCMKRNL=      272388
                                              JBC$_NOOPER=        272532
                                              JBC$_NOSYSNAM=      272404
                                              JBC$_OPENIN=        266392
                                              JBC$_OPENOUT=       266400
                                              JBC$_READERR=       266416
                                              JBC$_WRITEERR=      266448
                                                      .EXTRN  EXE$ALLOCBUF, EXE$EPID_TO_IPID
                                                      .EXTRN  EXE$JBCRSP, SCH$QAST

                                                      .PSECT  CODE,NOWRT,2

                                        003C 00000          .ENTRY  CREATE_SRB, Save R2,R3,R4,R5    ; 1177
                53 00000000'    EF   9E 00002           MOVAB   MBX, R3                         ;
                50        04    AC   D0 00009           MOVL    SRB, R0                         ; 1211
                52              63   D0 0000D           MOVL    MBX, R2                         ;
                60              62   D0 00010           MOVL    (R2), (R0)                      ;
          04    A0        78    A3   D0 00013           MOVL    RSP, 4(R0)                      ; 1213
                            08    A0   D4 00018           CLRL    8(R0)                           ; 1214
                51          0C  A0   9E 0001B           MOVAB   12(R0), R1                      ; 1215
                                  61   7C 0001F           CLRQ    (R1)                            ;
          08    A1                2E   B0 00021           MOVW    #46, 8(R1)                      ; 1217
          0A    A1                02   90 00025           MOVB    #2, 10(R1)                      ; 1218
    0B    A1        25    A2        80 8F 89 00029           BISB3   #128, 37(R2), 11(R1)            ; 1219
                      50        FC  A2   D0 00030           MOVL    -4(R2), R0                      ; 1220
                          00000000G   00   16 00034           JSB     EXE$EPID_TO_IPID                ;
          0C    A1                50   D0 0003A           MOVL    R0, 12(R1)                      ;
                50                63   D0 0003E           MOVL    MBX, R0                         ; 1221
          10    A1        50      A0   D0 00041           MOVL    80(R0), 16(R1)                  ;
                                  05   13 00046           BEQL    1$                              ; 1222
          0B    A1        40      8F   88 00048           BISB2   #64, 11(R1)                     ;
          14    A1        54      A0   D0 0004D 1$:       MOVL    84(R0), 20(R1)                  ; 1223
          18    A1  00000000G     00   9E 00052           MOVAB   EXE$JBCRSP, 24(R1)              ; 1224
          1C    A1        44      A0   7D 0005A           MOVQ    68(R0), 28(R1)                  ; 1225
          24    A1        4C      A0   D0 0005F           MOVL    76(R0), 36(R1)                  ; 1227
                            28    A1   D4 00064           CLRL    40(R1)                          ; 1228
                            2C    A1   B4 00067           CLRW    44(R1)                          ; 1229
                                       04 0006A           RET                                     ; 1230

; Routine Size:  107 bytes,    Routine Base:  CODE + 0000
```

```
;  194          1231  1 GLOBAL ROUTINE LOCATE_SRB_OUTPUT_ITEM(SRB,CODE,USER_DESC)=
;  195          1232  1
;  196          1233  1 !++
;  197          1234  1 !
;  198          1235  1 ! FUNCTIONAL DESCRIPTION:
;  199          1236  1 !       This routine locates storage for an output value item in a
;  200          1237  1 !       service response block.
;  201          1238  1 !
;  202          1239  1 ! INPUT PARAMETERS:
;  203          1240  1 !       SRB             - Pointer to SRB buffer.
;  204          1241  1 !       CODE            - $GETQUI or $SNDJBC output value item code.
;  205          1242  1 !       USER_DESC       - Structure describing user's return buffer.
;  206          1243  1 !
;  207          1244  1 ! IMPLICIT INPUTS:
;  208          1245  1 !       NONE
;  209          1246  1 !
;  210          1247  1 ! OUTPUT PARAMETERS:
;  211          1248  1 !       NONE
;  212          1249  1 !
;  213          1250  1 ! IMPLICIT OUTPUTS:
;  214          1251  1 !       SRB buffer updated.
;  215          1252  1 !
;  216          1253  1 ! ROUTINE VALUE:
;  217          1254  1 !       Pointer to data storage, or 0 if item should not be stored.
;  218          1255  1 !
;  219          1256  1 ! SIDE EFFECTS:
;  220          1257  1 !       NONE
;  221          1258  1 !
;  222          1259  1 !--
;  223          1260  1
;  224          1261  2 BEGIN
;  225          1262  2 MAP
;  226          1263  2         SRB:            REF BBLOCK,     ! Pointer to SRB buffer
;  227          1264  2         USER_DESC:      REF BBLOCK;     ! Pointer to output item descriptor
;  228          1265  2 LOCAL
;  229          1266  2         ACB:            REF BBLOCK,     ! Pointer to ACB
;  230          1267  2         ACI:            REF BBLOCK;     ! Pointer to extended ACB output item
;  231          1268  2
;  232          1269  2
;  233          1270  3 IF (.SRB[SRB_W_TYPE] EQL MSG$_SNDJBC OR .SRB[SRB_W_TYPE] EQL MSG$_GETQUI)
;  234          1271  2 AND .ITEM_PRESENT[.CODE]
;  235          1272  2 THEN
;  236          1273  3     BEGIN
;  237          1274  3     ACB = SRB[SRB_T_ACB];
;  238          1275  3     ACI = .ACB + .ACB[ACB$W_SIZE];
;  239          1276  3     ACB[ACB_W_ITEMCOUNT] = .ACB[ACB_W_ITEMCOUNT] + 1;
;  240          1277  3     ACI[ACI_W_SIZE] = .USER_DESC[ODSC_W_LENGTH];
;  241          1278  3     ACI[ACI_W_DATASIZE] = 0;
;  242          1279  3     ACI[ACI_L_ADDRESS] = .USER_DESC[ODSC_A_POINTER];
;  243          1280  3     ACI[ACI_L_LENADDR] = .USER_DESC[ODSC_A_LENPOINTER];
;  244          1281  3     ACI[ACI_T_DATA]
;  245          1282  3     END
;  246          1283  2 ELSE
;  247          1284  2     0
;  248          1285  1 END;
```

```
                                        0000 00000          .ENTRY  LOCATE_SRB_OUTPUT_ITEM, Save nothing    ; 1231
                                  0F    04   BC  B1 00002    CMPW    @SRB, #15                              ; 1270
                                        06   13  00006    BEQL    1$
                                  10    04   BC  B1 00008    CMPW    @SRB, #16
                                        24   12  0000C    BNEQ    2$
                1B 00000000'      EF    08   AC  E1 0000E 1$:  BBC     CODE, ITEM_PRESENT, 2$                 ; 1271
                51        04      AC          0C  C1 00017    ADDL3   #12, SRB, ACB                          ; 1274
                                  50    08    A1  3C 0001C    MOVZWL  8(ACB), ACI                            ; 1275
                                  50          51  C0 00020    ADDL2   ACB, ACI
                                              2C  A1 B6 00023    INCW    44(ACB)                            ; 1276
                                  51    0C    AC  D0 00026    MOVL    USER_DESC, R1                          ; 1277
                                  80          61  3C 0002A    MOVZWL  (R1)+, (ACI)+
                                  80    02    A1  7D 0002D    MOVQ    2(R1), (ACI)+                          ; 1279
                                              04  00031    RET                                            ; 1281
                                              50  D4 00032 2$.  CLRL    R0                                 ; 1270
                                                  04 00034    RET                                            ; 1285

; Routine Size:  53 bytes,    Routine Base:  CODE + 006B
```

```
;  250          1286  1 GLOBAL ROUTINE COMPLETE_SRB_OUTPUT_ITEM(SRB,VALUE_LENGTH): NOVALUE=
;  251          1287  1
;  252          1288  1 !++
;  253          1289  1 !
;  254          1290  1 ! FUNCTIONAL DESCRIPTION:
;  255          1291  1 !       This routine completes the modifications required to append an output
;  256          1292  1 !       value item to a service response block.
;  257          1293  1 !
;  258          1294  1 ! INPUT PARAMETERS:
;  259          1295  1 !       SRB             - Pointer to SRB buffer.
;  260          1296  1 !       VALUE_LENGTH    - Length of item value.
;  261          1297  1 !
;  262          1298  1 ! IMPLICIT INPUTS:
;  263          1299  1 !       NONE
;  264          1300  1 !
;  265          1301  1 ! OUTPUT PARAMETERS:
;  266          1302  1 !       NONE
;  267          1303  1 !
;  268          1304  1 ! IMPLICIT OUTPUTS:
;  269          1305  1 !       SRB buffer updated.
;  270          1306  1 !
;  271          1307  1 ! ROUTINE VALUE:
;  272          1308  1 !       NONE
;  273          1309  1 !
;  274          1310  1 ! SIDE EFFECTS:
;  275          1311  1 !       NONE
;  276          1312  1 !
;  277          1313  1 !--
;  278          1314  1
;  279          1315  2 BEGIN
;  280          1316  2 MAP
;  281          1317  2         SRB:            REF BBLOCK;     ! Pointer to SRB buffer
;  282          1318  2 LOCAL
;  283          1319  2         ACB:            REF BBLOCK,     ! Pointer to ACB
;  284          1320  2         ACI:            REF BBLOCK;     ! Pointer to extended ACB output item
;  285          1321  2
;  286          1322  2
;  287          1323  2 ! Locate the ACB and the output item, and update the length information.
;  288          1324  2 !
;  289          1325  2 ACB = SRB[SRB_T_ACB];
;  290          1326  2 ACI = .ACB + .ACB[ACB$W_SIZE];
;  291          1327  2 ACB[ACB$W_SIZE] = .ACB[ACB$W_SIZE] + $BYTEOFFSET(ACI_T_DATA) + .VALUE_LENGTH;
;  292          1328  2 ACI[ACI_W_DATASIZE] = .VALUE_LENGTH;
;  293          1329  1 END;


                                                0004 00000          .ENTRY  COMPLETE_SRB_OUTPUT_ITEM, Save R2       ; 1286
                50      04    AC                  0C   C1 00002    ADDL3   #12, SRB, ACB                           ; 1325
                              52          08      A0   3C 00007    MOVZWL  8(ACB), ACI                             ; 1326
                              52                  50   C0 0000B    ADDL2   ACB, ACI                                ;
                              51          08      A0   3C 0000E    MOVZWL  8(ACB), R1                              ; 1327
                              51          08      AC   C0 00012    ADDL2   VALUE_LENGTH, R1                        ;
          08    A0            51                  0C   A1 00016    ADDW3   #12, R1, 8(ACB)                         ;
                        02    A2          08      AC   B0 0001B    MOVW    VALUE_LENGTH, 2(ACI)                    ; 1328
```

```
                                04 00020        RET                             ; 1329

; Routine Size:  33 bytes,    Routine Base:  CODE + 00A0
```

```
;  295     1330  1 ROUTINE SEND_SERVICE_RESPONSE_KERNEL=
;  296     1331  1
;  297     1332  1 !++
;  298     1333  1 !
;  299     1334  1 ! FUNCTIONAL DESCRIPTION:
;  300     1335  1 !       This routine is used by SEND_SERVICE_RESPONSE_MESSAGE to allocate an
;  301     1336  1 !       extended AST control block from nonpaged pool and queue a special
;  302     1337  1 !       kernel AST to a caller of $GETQUI or $SNDJBC.  It executes in kernel
;  303     1338  1 !       mode and partially at IPL$_ASTDEL.
;  304     1339  1 !
;  305     1340  1 ! INPUT PARAMETERS:
;  306     1341  1 !       AP              - Pointer to prototype ACB.
;  307     1342  1 !
;  308     1343  1 ! IMPLICIT INPUTS:
;  309     1344  1 !       NONE
;  310     1345  1 !
;  311     1346  1 ! OUTPUT PARAMETERS:
;  312     1347  1 !       NONE
;  313     1348  1 !
;  314     1349  1 ! IMPLICIT OUTPUTS:
;  315     1350  1 !       Special kernel AST queued to requesting process.
;  316     1351  1 !
;  317     1352  1 ! ROUTINE VALUE:
;  318     1353  1 !       Completion status.
;  319     1354  1 !
;  320     1355  1 ! SIDE EFFECTS:
;  321     1356  1 !       NONE
;  322     1357  1 !
;  323     1358  1 !--
;  324     1359  1
;  325     1360  2 BEGIN
;  326     1361  2 BUILTIN
;  327     1362  2         AP;
;  328     1363  2 MAP
;  329     1364  2         AP:             REF BBLOCK;     ! Pointer to prototype ACB
;  330     1365  2 LOCAL
;  331     1366  2         ACB:            REF BBLOCK,     ! Pointer to AST control block
;  332     1367  2         STATUS;                         ! Status return
;  333     1368  2
;  334     1369  2
;  335     1370  2 ! Allocate the ACB from nonpaged pool.  If successful, the routine returns at
;  336     1371  2 ! IPL$_ASTDEL.
;  337     1372  2 !
;  338     1373  2 STATUS = EXE$ALLOCBUF(.AP[ACB$W_SIZE]; ACB);
;  339     1374  2 IF .STATUS
;  340     1375  2 THEN
;  341     1376  3     BEGIN
;  342     1377  3
;  343     1378  3     ! Initialize the ACB.
;  344     1379  3     !
;  345     1380  3     CH$MOVE(.AP[ACB$W_SIZE], .AP, .ACB);
;  346     1381  3
;  347     1382  3
;  348     1383  3     ! Queue the special kernel AST to the requesting process.  If the process
;  349     1384  3     ! no longer exists, the ACB is deallocated by SCH$QAST.
;  350     1385  3     !
;  351     1386  3     SCH$QAST(0, .ACB);
```

```
;  352          1387  3
;  353          1388  3
;  354          1389  3      ! Lower IPL.
;  355          1390  3      !
;  356          1391  3      MTPR(%REF(0), PR$_IPL);
;  357          1392  2      END;
;  358          1393  2
;  359          1394  2
;  360          1395  2  ! Return status.
;  361          1396  2  !
;  362          1397  2  .STATUS
;  363          1398  1  END;


                                        00FC 00000 SEND_SERVICE_RESPONSE_KERNEL:
                                                   .WORD   Save R2,R3,R4,R5,R6,R7                  ; 1330
                              51      08    AC  3C 00002         MOVZWL  8(AP), R1                 ; 1373
                                 00000000G  00  16 00006         JSB     EXE$ALLOCBUF              ;
                              57            50  D0 0000C         MOVL    R0, STATUS                ;
                              56            52  D0 0000F         MOVL    R2, R6                    ;
                              13            57  E9 00012         BLBC    STATUS, 1$                ; 1374
                        66    6C      08    AC  28 00015         MOVC3   8(AP), (AP), (ACB)        ; 1380
                              55            56  D0 0001A         MOVL    ACB, R5                   ; 1386
                                            50  D4 0001D         CLRL    R0                        ;
                                 00000000G  00  16 0001F         JSB     SCH$QAST                  ;
                              12            00  DA 00025         MTPR    #0, #18                   ; 1391
                              50            57  D0 00028 1$:     MOVL    STATUS, R0                ; 1398
                                                04 0002B         RET                               ;

; Routine Size:  44 bytes,    Routine Base:  CODE + 00C1
```

```
;  365     1399  1 GLOBAL ROUTINE SEND_SERVICE_RESPONSE_MESSAGE(SRB,STS): NOVALUE=
;  366     1400  1
;  367     1401  1 !++
;  368     1402  1 !
;  369     1403  1 ! FUNCTIONAL DESCRIPTION:
;  370     1404  1 !       This routine sends a response message to a caller of $GETQUI, $SNDSMB,
;  371     1405  1 !       $SNDACC, or $SNDJBC.
;  372     1406  1 !
;  373     1407  1 ! INPUT PARAMETERS:
;  374     1408  1 !       SRB             - Service response block.
;  375     1409  1 !       STS             - Request completion status.
;  376     1410  1 !
;  377     1411  1 ! IMPLICIT INPUTS:
;  378     1412  1 !       NONE
;  379     1413  1 !
;  380     1414  1 ! OUTPUT PARAMETERS:
;  381     1415  1 !       NONE
;  382     1416  1 !
;  383     1417  1 ! IMPLICIT OUTPUTS:
;  384     1418  1 !       NONE
;  385     1419  1 !
;  386     1420  1 ! ROUTINE VALUE:
;  387     1421  1 !       NONE
;  388     1422  1 !
;  389     1423  1 ! SIDE EFFECTS:
;  390     1424  1 !       Message written to mailbox, or special kernel AST queued to process.
;  391     1425  1 !
;  392     1426  1 !--
;  393     1427  1
;  394     1428  2 BEGIN
;  395     1429  2 MAP
;  396     1430  2         SRB:            REF BBLOCK;     ! Service response block
;  397     1431  2
;  398     1432  2
;  399     1433  2 IF .SRB[SRB_W_TYPE] EQL MSG$_SNDACC
;  400     1434  2 OR .SRB[SRB_W_TYPE] EQL MSG$_SNDSMB
;  401     1435  2 THEN
;  402     1436  3     BEGIN
;  403     1437  3     LOCAL
;  404     1438  3         DEVICE_DESC:    VECTOR[2],      ! Descriptor for device name
;  405     1439  3         DEVICE_BUFFER:  VECTOR[10,BYTE],! Buffer for device name
;  406     1440  3         CHANNEL:        WORD,           ! Channel assigned to mailbox
;  407     1441  3         STATUS;                         ! Status return
;  408     1442  3
;  409     1443  3
;  410     1444  3     ! If the unit number is zero, exit quietly.
;  411     1445  3     !
;  412     1446  3     IF .SRB[SRB_W_MAILBOX] NEQ 0
;  413     1447  3     THEN
;  414     1448  4         BEGIN
;  415     1449  4
;  416     1450  4         ! Set up the device name as "_MBAu:".
;  417     1451  4         !
;  418     1452  4         DEVICE_DESC[0] = %ALLOCATION(DEVICE_BUFFER);
;  419     1453  4         DEVICE_DESC[1] = DEVICE_BUFFER;
;  420   P 1454  4         $FAO(
;  421   P 1455  4             $DESCRIPTOR('_MBA!UW:'),
```

```
;  422        P 1456  4              DEVICE_DESC,
;  423        P 1457  4              DEVICE_DESC,
;  424          1458  4              .SRB[SRB_W_MAILBOX]);
;  425          1459  4
;  426          1460  4
;  427          1461  4          ! Assign a channel to the mailbox.
;  428          1462  4          !
;  429          1463  5          IF $ASSIGN(DEVNAM=DEVICE_DESC, CHAN=CHANNEL)
;  430          1464  4          THEN
;  431          1465  5              BEGIN
;  432          1466  5
;  433          1467  5              ! Set the completion status into the message buffer.
;  434          1468  5              !
;  435          1469  5              BBLOCK[SRB[SRB_Q_RSP], RSP_L_STATUS] = .STS;
;  436          1470  5
;  437          1471  5
;  438          1472  5              ! Write the message without waiting.
;  439          1473  5              !
;  440        P 1474  5              STATUS = $QIO(
;  441        P 1475  5                  FUNC=IO$_WRITEVBLK OR IO$M_NOW OR IO$M_NORSWAIT,
;  442        P 1476  5                  CHAN=.CHANNEL,
;  443        P 1477  5                  P1=SRB[SRB_Q_RSP],
;  444          1478  5                  P2=SRB_S_RSP);
;  445          1479  5              IF NOT .STATUS
;  446          1480  5              THEN SIGNAL(JBC$_WRIRSPMSG OR STS$K_ERROR, 0, .STATUS);
;  447          1481  5
;  448          1482  5
;  449          1483  5              ! Deassign the channel.
;  450          1484  5              !
;  451          1485  5              $DASSGN(CHAN=.CHANNEL);
;  452          1486  4              END;
;  453          1487  3          END;
;  454          1488  3      END
;  455          1489  2 ELSE
;  456          1490  3      BEGIN
;  457          1491  3      LOCAL
;  458          1492  3          STATUS;
;  459          1493  3
;  460          1494  3
;  461          1495  3      ! Set the completion status into the ACB.
;  462          1496  3      !
;  463          1497  3      BBLOCK[SRB[SRB_T_ACB], ACB_L_STATUS] = .STS;
;  464          1498  3
;  465          1499  3
;  466          1500  3      ! Send the completion.
;  467          1501  3      !
;  468        P 1502  3      STATUS = $CMKRNL(
;  469        P 1503  3          ROUTIN=SEND_SERVICE_RESPONSE_KERNEL,
;  470          1504  3          ARGLST=SRB[SRB_T_ACB]);
;  471          1505  3      IF NOT .STATUS
;  472          1506  3      THEN
;  473          1507  3          SIGNAL(JBC$_WRIRSPMSG OR STS$K_ERROR, 0, .STATUS);
;  474          1508  2      END;
;  475          1509  1 END;
```

```
                3A 57 55 21 41 42 4D 5F  000ED P.AAB:  .ASCII  \_MBA!UW:\                                    ;
                                         000F5         .BLKB   3                                            ;
                         00000008        000F8 P.AAA:  .LONG   8                                            ;
                         00000000'       000FC         .ADDRESS P.AAB                                       ;

                                                       .EXTRN  SYS$FAO, SYS$ASSIGN
                                                       .EXTRN  SYS$QIO, SYS$DASSGN
                                                       .EXTRN  SYS$CMKRNL

                                          000C 00000   .ENTRY  SEND_SERVICE_RESPONSE_MESSAGE, Save R2,R3      ; 1399
                    53 00000000G    00    9E 00002     MOVAB   LIB$SIGNAL, R3                               ;
                    5E              18    C2 00009     SUBL2   #24, SP                                      ;
                    52        04    AC    D0 0000C     MOVL    SRB, R2                                      ; 1433
                    0A              62    B1 00010     CMPW    (R2), #10                                    ;
                                    05    13 00013     BEQL    1$                                           ;
                    04              62    B1 00015     CMPW    (R2), #4                                     ; 1434
                                    74    12 00018     BNEQ    4$                                           ;
                              02    A2    B5 0001A 1$: TSTW    2(R2)                                        ; 1446
                                    01    12 0001D     BNEQ    2$                                           ;
                                          04 0001F     RET                                                  ;
                 10    AE               0A    D0 00020 2$: MOVL    #10, DEVICE_DESC                         ; 1452
                 14    AE         04    AE    9E 00024     MOVAB   DEVICE_BUFFER, DEVICE_DESC+4             ; 1453
                    7E         02    A2    3C 00029     MOVZWL  2(R2), -(SP)                                 ; 1458
                               14    AF    9F 0002D     PUSHAB  DEVICE_DESC                                  ;
                               18    AE    9F 00030     PUSHAB  DEVICE_DESC                                  ;
                               C2    AF    9F 00033     PUSHAB  P.AAA                                        ;
          00000000G  00              04    FB 00036     CALLS   #4, SYS$FAO                                  ;
                                     7E    7C 0003D     CLRQ    -(SP)                                        ; 1463
                               08    AE    9F 0003F     PUSHAB  CHANNEL                                      ;
                               1C    AE    9F 00042     PUSHAB  DEVICE_DESC                                  ;
          00000000G  00              04    FB 00045     CALLS   #4, SYS$ASSIGN                               ;
                     62              50    E9 0004C     BLBC    R0, 5$                                       ;
                08   A2         08    AC    D0 0004F     MOVL    STS, 8(R2)                                   ; 1469
                                     7E    7C 00054     CLRQ    -(SP)                                        ; 1478
                                     7E    7C 00056     CLRQ    -(SP)                                        ;
                                     08    DD 00058     PUSHL   #8                                           ;
                               04    A2    9F 0005A     PUSHAB  4(R2)                                        ;
                                     7E    7C 0005D     CLRQ    -(SP)                                        ;
                                     7E    D4 0005F     CLRL    -(SP)                                        ;
                    7E        0470     8F    3C 00061     MOVZWL  #1136, -(SP)                                 ;
                    7E          28     AE    3C 00066     MOVZWL  CHANNEL, -(SP)                               ;
                                     7E    D4 0006A     CLRL    -(SP)                                        ;
          00000000G  00              0C    FB 0006C     CALLS   #12, SYS$QIO                                 ;
                     0D              50    E8 00073     BLBS    STATUS, 3$                                   ; 1479
                                     50    DD 00076     PUSHL   STATUS                                       ; 1480
                                     7E    D4 00078     CLRL    -(SP)                                        ;
                         00048472    8F    DD 0007A     PUSHL   #296050                                      ;
                     63              03    FB 00080     CALLS   #3, LIB$SIGNAL                               ;
                     7E              6E    3C 00083 3$: MOVZWL  CHANNEL, -(SP)                               ; 1485
          00000000G  00              01    FB 00086     CALLS   #1, SYS$DASSGN                               ;
                                           04 0008D     RET                                                  ; 1433
                34   A2         08    AC    D0 0008E 4$: MOVL    STS, 52(R2)                                  ; 1497
                               0C    A2    9F 00093     PUSHAB  12(R2)                                       ; 1504
                         FF27    CF    9F 00096     PUSHAB  SEND_SERVICE_RESPONSE_KERNEL                 ;
          00000000G  00              02    FB 0009A     CALLS   #2, SYS$CMKRNL                               ;
                     0D              50    E8 000A1     BLBS    STATUS, 5$                                   ; 1505
                                     50    DD 000A4     PUSHL   STATUS                                       ; 1507
```

```
                        7E  D4 000A6          CLRL    -(SP)                                   ;
                00048472    8F  DD 000A8          PUSHL   #296050                         ;
                        63  03  FB 000AE          CALLS   #3, LIB$SIGNAL                  ;
                            04 000B1 5$:        RET                                     ; 1509

; Routine Size:  178 bytes,    Routine Base:  CODE + 0100
```

```
;  477          1510  1 GLOBAL ROUTINE SEND_SERVICE_RESPONSE: NOVALUE=
;  478          1511  1
;  479          1512  1 !++
;  480          1513  1 !
;  481          1514  1 ! FUNCTIONAL DESCRIPTION:
;  482          1515  1 !       This routine finishes processing for the $GETQUI, $SNDSMB, $SNDACC,
;  483          1516  1 !       and $SNDJBC services by sending a response message if appropriate.
;  484          1517  1 !
;  485          1518  1 ! INPUT PARAMETERS:
;  486          1519  1 !       NONE
;  487          1520  1 !
;  488          1521  1 ! IMPLICIT INPUTS:
;  489          1522  1 !       See CREATE_SRB.
;  490          1523  1 !
;  491          1524  1 ! OUTPUT PARAMETERS:
;  492          1525  1 !       NONE
;  493          1526  1 !
;  494          1527  1 ! IMPLICIT OUTPUTS:
;  495          1528  1 !       NONE
;  496          1529  1 !
;  497          1530  1 ! ROUTINE VALUE:
;  498          1531  1 !       NONE
;  499          1532  1 !
;  500          1533  1 ! SIDE EFFECTS:
;  501          1534  1 !       Response returned to caller.
;  502          1535  1 !
;  503          1536  1 !--
;  504          1537  1
;  505          1538  2 BEGIN
;  506          1539  2 LOCAL
;  507          1540  2         SRB:            BBLOCK[SRB_S_BUFFER];   ! Service response block
;  508          1541  2
;  509          1542  2
;  510          1543  2 IF .RSP[RSP_L_STATUS] NEQ 0
;  511          1544  2 THEN
;  512          1545  3     BEGIN
;  513          1546  3     IF NOT .RSP[RSP_L_STATUS]
;  514          1547  3         THEN BBLOCK[RSP[RSP_L_STATUS], STS$V_SEVERITY] = STS$K_ERROR;
;  515          1548  3     IF .BBLOCK[RSP[RSP_L_STATUS], STS$V_FAC_NO] EQL 0
;  516          1549  3         THEN BBLOCK[RSP[RSP_L_STATUS], STS$V_FAC_NO] = JBC$_FACILITY;
;  517          1550  3
;  518          1551  3
;  519          1552  3     CREATE_SRB(SRB);
;  520          1553  3     SEND_SERVICE_RESPONSE_MESSAGE(SRB, .RSP[RSP_L_STATUS]);
;  521          1554  2     END;
;  522          1555  1 END;


                                0004 00000          .ENTRY  SEND_SERVICE_RESPONSE, Save R2          ; 1510
52 00000000'    EF   9E 00002       MOVAB   RSP+4, R2                                               ;
5E         B8   AE   9E 00009       MOVAB   -72(SP), SP                                             ;
50              62   D0 0000D       MOVL    RSP+4, R0                                               ; 1543
                27   13 00010       BEQL    3$                                                      ;
05              50   E8 00012       BLBS    R0, 1$                                                  ; 1546
```

```
                62              03                      00          02  F0 00015            INSV    #2, #0, #3, RSP+4                       ; 1547
                                                OFFF    8F      02  A2  B3 0C01A 1$:        BITW    RSP+6, #4095                            ; 1548
                                                                    06  12 00020            BNEQ    2$                                      ;
              02    A2              0C                      00      04  F0 00022            INSV    #4, #0, #12, RSP+6                      ; 1549
                                                                    5E  DD 00028 2$:        PUSHL   SP                                      ; 1552
                                                FE1F    CF          01  FB 0002A            CALLS   #1, CREATE_SRB                          ;
                                                                    62  DD 0002F            PUSHL   RSP+4                                   ; 1553
                                                            04      AE  9F 00031            PUSHAB  SRB                                     ;
                                                FF15    CF          02  FB 00034            CALLS   #2, SEND_SERVICE_RESPONSE_MESSAGE       ;
                                                                        04 00039 3$:        RET                                             ; 1555

; Routine Size:  58 bytes,    Routine Base:  CODE + 01B2
```

```
;  524          1556  1 END
;  525          1557  0 ELUDOM


                                                        .EXTRN  LIB$SIGNAL

;                               PSECT SUMMARY

;       Name                    Bytes                   Attributes

;  COMMON                       5024  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  OVR,NOPIC,ALIGN(2)
;  CODE                          492  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)


;                           Library Statistics

;                                    -------- Symbols --------      Pages       Processing
;       File                         Total    Loaded   Percent      Mapped      Time

;  _$255$DUA28:[SYSLIB]LIB.L32;1     18619        67         0      1000          00:01.4



;                                   COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:RESPONSE/OBJ=OBJ$:RESPONSE MSRC$:RESPONSE/UPDATE=(ENH$:RESPONSE)

; Size:          473 code + 5043 data bytes
; Run Time:          00:17.2
; Elapsed Time:      00:58.7
; Lines/CPU Min:      5437
; Lexemes/CPU-Min:  50427
; Memory Used:  247 pages
; Compilation Complete
```

RESTRICT
LIS

RESPONSE
LIS

SND:BC
LIS

SCHEDULER
LIS