

```

JJJ      000000000      BBBB BBBB BBBB      CCCCCCCCCC      TTTTTTTTTTTTTTTT      LLL
JJJ      00C000000      BBBB BBBB BBBB      CCCCCCCCCC      TTTTTTTTTTTTTTTT      LLL
JJJ      000000000      BBBB BBBB BBBB      CCCCCCCCCC      TTTTTTTTTTTTTTTT      LLL
JJJ      000      000      BBB      BBB      CCC      TTT      LLL
JJJ      000      000      BBB      BBB      CCC      TTT      LLL
JJJ      000      000      BBB      BBB      CCC      TTT      LLL
JJJ      000      000      BBB      BBB      CCC      TTT      LLL
JJJ      000      000      BBB      BBB      CCC      TTT      LLL
JJJ      000      000      BBB      BBB      CCC      TTT      LLL
JJJ      000      000      BBB      BBB      CCC      TTT      LLL
JJJ      000      000      BBBB BBBB BBBB      CCC      TTT      LLL
JJJ      000      000      BBBB BBBB BBBB      CCC      TTT      LLL
JJJ      000      000      BBBB BBBB BBBB      CCC      TTT      LLL
JJJ      000      000      BBB      BBB      CCC      TTT      LLL
JJJ      000      000      BBB      BBB      CCC      TTT      LLL
JJJ      000      000      BBB      BBB      CCC      TTT      LLL
JJJ      000      000      BBB      BBB      CCC      TTT      LLL
JJJ      000      000      BBB      BBB      CCC      TTT      LLL
JJJ      000      000      BBB      BBB      CCC      TTT      LLL
JJJ      000      000      BBB      BBB      CCC      TTT      LLL
JJJJJJJJ  000000000      BBBB BBBB BBBB      CCCCCCCCCC      TTT      LLLLLLLLLLLLLLLLL
JJJJJJJJ  000000000      BBBB BBBB BBBB      CCCCCCCCCC      TTT      LLLLLLLLLLLLLLLLL
JJJJJJJJ  000000000      BBBB BBBB BBBB      CCCCCCCCCC      TTT      LLLLLLLLLLLLLLLLL

```

[illegible][illegible]

```
EEEEEEEEEE XX XX EEEEEEEEE CCCCCCCC UU UU TTTTTTTTT 000000 RRRRRRRR
EEEEEEEEEE XX XX EEEEEEEEE CCCCCCCC UU UU TTTTTTTTT 000000 RRRRRRRR
EE XX XX EE EE CC CC UU UU TT TT 00 00 RR RR
EE XX XX EE EE CC CC UU UU TT TT 00 00 RR RR
EE XX XX EE EE CC CC UU UU TT TT 00 00 RR RR
EEEEEEEE XX XX EEEEEEEEE CCCCCCCC UU UU TT TT 00 00 RRRRRRRR
EEEEEEEE XX XX EEEEEEEEE CCCCCCCC UU UU TT TT 00 00 RRRRRRRR
EE XX XX EE EE CC CC UU UU TT TT 00 00 RR RR
EE XX XX EE EE CC CC UU UU TT TT 00 00 RR RR
EE XX XX EE EE CC CC UU UU TT TT 00 00 RR RR
EEEEEEEEEE XX XX EEEEEEEEE CCCCCCCC UUUUUUUUUU TT TT 000000 RR RR
EEEEEEEEEE XX XX EEEEEEEEE CCCCCCCC UUUUUUUUUU TT TT 000000 RR RR
.....
.....
.....
.....
```

```
LL I I I I I SSSSSSSS
LL I I I I I SSSSSSSS
LL I I SS
LL I I SS
LL I I SS
LL I I SSSSSS
LL I I SSSSSS
LL I I SS
LL I I SS
LL I I SS
LLLLLLLLLL I I I I I SSSSSSSS
LLLLLLLLLL I I I I I SSSSSSSS
```



```
1 0001 0 MODULE EXECUTOR (%TITLE 'Executor control'
2 0002 0 IDENT = 'V04-002'
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 control executors.
37 0037 1
38 0038 1 ENVIRONMENT:
39 0039 1 VAX/VMS user and kernel mode.
40 0040 1 --
41 0041 1
42 0042 1 AUTHOR: M. Jack, CREATION DATE: 16-Feb-1982
43 0043 1
44 0044 1 MODIFIED BY:
45 0045 1
46 0046 1 V04-002 JAK0234 J A Krycka 11-Sep-1984
47 0047 1 Increase process rundown timer to 2 minutes.
48 0048 1
49 0049 1 V04-001 JAK0233 J A Krycka 10-Sep-1984
50 0050 1 Set flag in START_EXECUTOR_JOB if the job could not be
51 0051 1 placed in execution and the job header record is released.
52 0052 1
53 0053 1 V03-005 SHZ0001 Stephen H. Zalewski, 07-Dec-1983
54 0054 1 Allow a process 5 seconds to rundown before reissuing
55 0055 1 the force exit in the next higher access mode.
56 0056 1
57 0057 1 V03-004 MLJ0115 Martin L. Jack, 30-Jul-1983 14:41
```


EXECUTOR
V04-002

Executor control

J 3
16-Sep-1984 00:03:38
14-Sep-1984 12:37:01

VAX-11 Bliss-32 V4.0-742
[JOBCTL.SRC]EXECUTOR.B32;2

Page 2
(1)

:	58	0058	1	:	Changes for job controller baselevel.	
:	59	0059	1	:		
:	60	0060	1	:	V03-003 MLJ0114	Martin L. Jack, 23-Jun-1983 4:58
:	61	0061	1	:	Changes for job controller baselevel.	
:	62	0062	1	:		
:	63	0063	1	:	V03-002 MLJ0113	Martin L. Jack, 26-May-1983 21:06
:	64	0064	1	:	Changes for job controller baselevel.	
:	65	0065	1	:		
:	66	0066	1	:	V03-001 MLJ0112	Martin L. Jack, 29-Apr-1983 2:57
:	67	0067	1	:	Changes for job controller baselevel.	
:	68	0068	1	:		
:	69	0069	1	:	**	


```

71 0070 1 REQUIRE 'SRC$:JOBCTLDEF';
72 1111 1
73 1112 1
74 1113 1 FORWARD ROUTINE
75 1114 1   START_EXECUTION,
76 1115 1   START_EXECUTOR_JOB:      NOVALUE,
77 1116 1   ABORT_EXECUTION,
78 1117 1   ABORT_EXECUTOR_JOB,
79 1118 1   PAUSE_EXECUTION,
80 1119 1   RESUME_EXECUTION,
81 1120 1   RESET_EXECUTOR_QUEUE:    NOVALUE;
82 1121 1
83 1122 1
84 1123 1 EXTERNAL ROUTINE
85 1124 1   ALLOCATE_MEMORY,
86 1125 1   COMPLETE_JOB:             NOVALUE,
87 1126 1   CREATE_SRQ_RECORD,
88 1127 1   DEALLOCATE_MEMORY:        NOVALUE,
89 1128 1   ENTER_PROCESS_DATA:        NOVALUE,
90 1129 1   FIND_PROCESS_DATA:        L_OUTPUT_3,
91 1130 1   PAUSE_SYMBIONT_TASK:      NOVALUE,
92 1131 1   READ_RECORD,
93 1132 1   RELEASE_RECORD:           NOVALUE,
94 1133 1   RESET_SYMBIONT_STREAM:    NOVALUE,
95 1134 1   RESUME_SYMBIONT_TASK:     NOVALUE,
96 1135 1   REWRITE_RECORD:           NOVALUE,
97 1136 1   START_SYMBIONT_TASK:      NOVALUE,
98 1137 1   STOP_SYMBIONT_TASK:       NOVALUE,
99 1138 1   UPDATE_GETQUI_DATA:       NOVALUE;
100 1139 1
101 1140 1
102 1141 1 EXTERNAL
103 1142 1   ALL_PRIVILEGES,
104 1143 1   NLA0_DESC;
105 1144 1
106 1145 1
107 1146 1 LINKAGE
108 1147 1   EXE$EPID TO PCB LINKAGE =
109 1148 1   JSB(REGISTER=0):
110 1149 1   PRESERVE(1, 2, 3, 4, 5) NOTUSED(6, 7, 8, 9, 10, 11),
111 1150 1
112 1151 1
113 1152 1   SCH$FORCEDEXIT LINKAGE =
114 1153 1   JSB(REGISTER=0, REGISTER=3, REGISTER=4):
115 1154 1   NOPRESERVE(1, 2) PRESERVE(5) NOTUSED(6, 7, 8, 9, 10, 11);
116 1155 1
117 1156 1
118 1157 1 EXTERNAL ROUTINE
119 1158 1   EXE$EPID TO PCB: EXE$EPID TO PCB LINKAGE ADDRESSING_MODE(GENERAL),
120 1159 1   SCH$FORCEDEXIT: SCH$FORCEDEXIT_LINKAGE ADDRESSING_MODE(GENERAL);
121 1160 1
122 1161 1
123 1162 1 FORWARD
124 1163 1   LOCK_IPL;
125 1164 1
126 1165 1
127 1166 1 BUILTIN
```

EXECUTOR
V04-002

Executor control

; 128

1167 1

MTPR;

L 3
16-Sep-1984 00:03:38
14-Sep-1984 12:37:01

VAX-11 Bliss-32 V4.0-742
[JOBCTL.SRC]EXECUTOR.B32;2

Page 4
(2)


```
130 1168 1 GLOBAL ROUTINE START_EXECUTION(SMQ_N,SMQ,SJH_N,SJH)=
131 1169 1
132 1170 1 ++
133 1171 1
134 1172 1 FUNCTIONAL DESCRIPTION:
135 1173 1 This routine performs the actual communication that starts a job on an
136 1174 1 executor.
137 1175 1
138 1176 1 INPUT PARAMETERS:
139 1177 1 SMQ_N - Record number of SMQ.
140 1178 1 SMQ - Pointer to SMQ.
141 1179 1 SJH_N - Record number of SJH.
142 1180 1 SJH - Pointer to SJH.
143 1181 1
144 1182 1 IMPLICIT INPUTS:
145 1183 1 NONE
146 1184 1
147 1185 1 OUTPUT PARAMETERS:
148 1186 1 NONE
149 1187 1
150 1188 1 IMPLICIT OUTPUTS:
151 1189 1 NONE
152 1190 1
153 1191 1 ROUTINE VALUE:
154 1192 1 Completion status.
155 1193 1
156 1194 1 SIDE EFFECTS:
157 1195 1 NONE
158 1196 1
159 1197 1 --
160 1198 1
161 1199 2 BEGIN
162 1200 2 MAP
163 1201 2 SMQ: REF BBLOCK, ! Pointer to SMQ
164 1202 2 SJH: REF BBLOCK; ! Pointer to SJH
165 1203 2
166 1204 2
167 1205 2 IF .SMQ[SMQ$V_BATCH]
168 1206 2 THEN
169 1207 3 BEGIN
170 1208 3 LOCAL
171 1209 3 IMAGE_BUFFER: VECTOR[63,BYTE], ! Buffer for image name
172 1210 3 IMAGE_DESC: VECTOR[2], ! Descriptor for image name
173 1211 3 PRCNAM_BUFFER: VECTOR[15,BYTE], ! Buffer for process name
174 1212 3 PRCNAM_DESC: VECTOR[2], ! Descriptor for process name
175 1213 3 PRCNAM, ! Process name parameter
176 1214 3 STSFLG: BBLOCK[4], ! Process status
177 1215 3 STATUS; ! Status return
178 1216 3
179 1217 3
180 1218 3 ! Set up the process name as 'BATCH_n'.
181 1219 3
182 1220 3 PRCNAM_DESC[0] = %ALLOCATION(PRCNAM_BUFFER);
183 1221 3 PRCNAM_DESC[1] = PRCNAM_BUFFER;
184 1222 3 $FAO(
185 1223 3 $DESCRIPTOR('BATCH_!UL'),
186 1224 3 PRCNAM_DESC,
```

```
187 P 1225 3 PRCNAM_DESC,  
188 1226 .SJH[SJH$ENTRY_NUMBER]);  
189 1227  
190 1228  
191 1229 ! Set up the image name as 'SYS$SYSTEM:name.EXE'.  
192 1230  
193 1231 IMAGE_DESC[0] = %ALLOCATION(IMAGE_BUFFER);  
194 1232 IMAGE_DESC[1] = IMAGE_BUFFER;  
195 P 1233 $FAO(  
196 P 1234 $DESCRIPTOR('SYS$SYSTEM:!AC.EXE'),  
197 P 1235 IMAGE_DESC,  
198 P 1236 IMAGE_DESC,  
199 P 1237 (IF CH$RCHAR(SJH[SJH$T_CLI]) EQL 0  
200 1238 THEN UPLIT BYTE (%ASCIC 'LOGINOUT')  
201 1239 ELSE SJH[SJH$T_CLI]));  
202 1240  
203 1241  
204 1242 ! Set up the process status.  
205 1243  
206 1244 STSFLG = PRC$M_BATCH;  
207 1245 IF NOT .SMQ[SMQ$V_SWAP] THEN STSFLG[PRC$V_PSWAPM] = TRUE;  
208 1246  
209 1247  
210 1248 ! The following loop is executed at most twice.  
211 1249  
212 1250 PRCNAM = PRCNAM_DESC;  
213 1251 WHILE TRUE DO  
214 1252 BEGIN  
215 1253  
216 1254 ! Try to create the process.  
217 1255  
218 P 1256 STATUS = $CREPRC(  
219 P 1257 PIDADR=SJH[SJH$EXECUTOR_PID],  
220 P 1258 IMAGE=IMAGE_DESC,  
221 P 1259 INPUT=NLAO_DESC,  
222 P 1260 OUTPUT=PRCNAM_DESC,  
223 P 1261 ERROR=PRCNAM_DESC,  
224 P 1262 PRVADR=ALL_PRIVILEGES,  
225 P 1263 PRCNAM=.PRCNAM,  
226 P 1264 BASPRI=.SMQ[SMQ$B_BASE_PRIORITY],  
227 P 1265 UIC=.JBC_UIC,  
228 1266 STSFLG=.STSFLG);  
229 1267  
230 1268  
231 1269 IF NOT .STATUS  
232 1270 THEN  
233 1271 BEGIN  
234 1272  
235 1273 ! Create failed. If the status is not 'duplicate process name', or  
236 1274 ! if a create has already been tried with no name, give up.  
237 1275 ! Otherwise, loop to try creation with no name.  
238 1276  
239 1277 IF .STATUS<0,16> NEQ SS$_DUPLNAM OR .PRCNAM EQL 0  
240 1278 THEN RETURN .STATUS;  
241 1279 PRCNAM = 0;  
242 1280 END  
243 1281 ELSE
```



```

: 244      1282 4      EXITLOOP;
: 245      1283      END;
: 246      1284
: 247      1285
: 248      1286      ! Create succeeded. Record the batch process in the process database
: 249      1287      ! and mark the job executing.
: 250      1288
: 251      1289      ENTER_PROCESS_DATA(PDE_K_BATCH, .SJH[SJH$L_EXECUTOR_PID], .SMQ_N, .SJH_N);
: 252      1290      QUEUE_REFERENCE_COUNT = .QUEUE_REFERENCE_COUNT + 1;
: 253      1291      SJH[SJH$V_EXECUTING] = TRUE;
: 254      1292      END
: 255      1293      ELSE
: 256      1294      BEGIN
: 257      1295      LOCAL
: 258      1296      SQR_N,
: 259      1297      SQR;
: 260      1298      REF BBLOCK;
: 261      1299      ! Record number of SQR
: 262      1300      ! Pointer to SQR
: 263      1301
: 264      1302      ! Start the first file in the job.
: 265      1303      !
: 266      1304      COPY TIME(CUR TIME, SMQ[SMQ$Q_ACM_BEGTIM]);
: 267      1305      SQR_N = .SJH[SJH$L_CURRENT_FILE_CHKPT];
: 268      1306      IF .SQR_N EQL 0 THEN SQR_N = .SJH[SJH$L_FILE_LIST];
: 269      1307      SQR = READ_RECORD(.SQR_N);
: 270      1308      START_SYMBIONT_TASK(.SMQ_N, .SMQ, .SJH_N, .SJH, .SQR_N, .SQR);
: 271      1309      RELEASE_RECORD(.SQR_N);
: 272      1310      END;
: 273      1311      ! Update SJH.
: 274      1312      !
: 275      1313      SJH[SJH$L_CONDITION_1] = 0;
: 276      1314      SJH[SJH$L_CONDITION_2] = 0;
: 277      1315      SJH[SJH$L_CONDITION_3] = 0;
: 278      1316      SSS_NORMAL
: 279      1317      1 END;
```

.TITLE EXECUTOR Executor control
.IDENT \V04-002\

.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

C 4
16-Sep-1984 00:03:38
14-Sep-1984 12:37:01VAX-11 Bliss-32 V4.0-742
[JOBCTL.SRC]EXECUTOR.B32;2

0019C SNDACC_COUNT:
 .BKLB 28
001B8 SNDSMB_COUNT:
 .BKLB 72
00200 DIAG_STORAGE_END:
 .BKLB 0
00200 FLAGS: .BKLB 4
00204 IMAGE_DUMP_STSFLG:
 .BKLB 4
00208 THIS_SYSID:
 .BKLB 6
0020E .BKLB 2
00210 CUR_TIME:
 .BKLB 8
00218 HOURLY_TIME:
 .BKLB 8
00220 HOURLY_PARAMS:
 .BKLB 20
00234 SYMBIONT_COUNT:
 .BKLB 4
00238 QUEUE_REFERENCE_COUNT:
 .BKLB 4
0023C MBX_MESSAGE_COUNT:
 .BKLB 4
00240 MBX: .BKLB 4
00244 MBX_END: .BKLB 4
00248 MEMORY_FREE_QUEUES:
 .BKLB 40
00270 NONAST_WORK_QUEUE:
 .BKLB 8
00278 BCB_FREE_LIST:
 .BKLB 4
0027C BCB_ACTIVE_LIST:
 .BKLB 4
00280 GQL_FREE_LIST:
 .BKLB 4
00284 GQL_ACTIVE_LIST:
 .BKLB 4
00288 OPEN_GETQUI_LIST:
 .BKLB 4
0028C PROCESS_DATA_LIST:
 .BKLB 4
00290 SYMBIONT_CONTROL:
 .BKLB 4
00294 SPARE_AREA:
 .BKLB 12
002A0 REMOTE_REQUEST_LKSB:
 .BKLB 8
002A8 QUEUE_FILE_LKSB:
 .BKLB 8
002B0 QUEUE_LOCK_LKSB:
 .BKLB 8
002B8 RSP: .BKLB 8
002C0 JBC_PRIORITY:
 .BKLB 4
002C4 JBC_PRIVILEGES:
 .BKLB 8

Executor control

D 4
16-Sep-1984 00:03:38
14-Sep-1984 12:37:01

VAX-11 Bliss-32 V4.0-742
[JOBCTL.SRC]EXECUTOR.B32;2

Page 9
(3)

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:	.BLRB	80
0078C	ACCOUNT_RAB B:	.BLRB	68
00800	ACCOUNT_NAM B:	.BLRB	96
00860	ACCOUNT_RSA B:	.BLRB	255
0095F		.BLKB	1
00960	DIAG_FAB:	.BLKB	80
009B0	DIAG_RAB:	.BLKB	68
009F4	MBX_CHAN:	.BLKB	4
009F8	MBX_IOSB:	.BLKB	8
00A0C	MBX_BUFFER:	.BLKB	1024
00E00	VALUE_STORAGE_BASE:	.BLKB	0
00E00	ITEM_PRESENT:	.BLKB	32
00E20	VALUE_GETQUI_BASE:		

00E20 VALUE_ACCOUNTING_MESSAGE:
.BLKB 0
00E26 VALUE_ACCOUNTING_TYPES:
.BLKB 8
00E2A VALUE_AFTER_TIME:
.BLKB 4
00E32 VALUE_ALIGNMENT_PAGES:
.BLKB 8
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 8
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 8
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:
 .B[KB] 2
00ECD VALUE_FORM MARGIN_RIGHT:
 .B[KB] 2
00ECF VALUE_FORM MARGIN_TOP:
 .B[KB] 1
00ED0 VALUE_FORM NAME:
 .B[KB] 6
00ED6 VALUE_FORM NUMBER:
 .B[KB] 4
00EDA VALUE_FORM:
 .B[KB] 8
00EE2 VALUE_FORM SETUP_MODULES:
 .B[KB] 6
00EE8 VALUE_FORM STOCK:
 .B[KB] 6
00EEE VALUE_FORM WIDTH:
 .B[KB] 2
00EF0 VALUE_GENERIC_TARGET:
 .B[KB] 996
012D4 VALUE_JOB COPIES:
 .B[KB] 1
012D5 VALUE_JOB LIMIT:
 .B[KB] 1
012D6 VALUE_JOB NAME:
 .B[KB] 6
012DC VALUE_JOB RESET_MODULES:
 .B[KB] 6
012E2 VALUE_JOB SIZE_MAXIMUM:
 .B[KB] 4
012E6 VALUE_JOB SIZE_MINIMUM:
 .B[KB] 4
012EA VALUE_JOB STATUS_OUTPUT:
 .B[KB] T0
012F4 VALUE_LAST_PAGE:
 .B[KB] 4
012F8 VALUE_LIBRARY_SPECIFICATION:
 .B[KB] 6
012FE VALUE_LOG_QUEUE:
 .B[KB] 8
01306 VALUE_LOG_SPECIFICATION:
 .B[KB] 6
0130C VALUE_NOTE:
 .B[KB] 6
01312 VALUE_OPERATOR_REQUEST:
 .B[KB] 6
01318 VALUE_OWNER_UIC:
 .B[KB] 4
0131C VALUE_PAGE_SETUP_MODULES:
 .B[KB] 8
01322 VALUE_PARAMETER_1:
 .B[KB] 6
01328 VALUE_PARAMETER_2:
 .B[KB] 6
0132E VALUE_PARAMETER_3:
 .B[KB] 6
01334 VALUE_PARAMETER_4:

```
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 8
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_SC$NODE_NAME: .BLKB 6
0139A VALUE_WS$DEFAULT: .BLKB 2
0139C VALUE_WS$EXTENT: .BLKB 2
0139E VALUE_WS$QUOTA: .BLKB 2
013A0 VALUE_STORAGE_END: .BLKB 0
```

.PSECT CODE,NOWRT,2

```
4C 55 21 5F 48 43 54 41 42 00000 P.AAB: .ASCII \BATCH_!UL\
00009 .BLKB 3
00000009 0000C P.AAA: .LONG 9
00000000 00010 .ADDRESS P.AAB
2E 43 41 21 3A 4D 45 54 53 59 53 24 53 59 53 00014 P.AAD: .ASCII \SYS$SYSTEM:!AC.EXE\
45 58 45 00023
00026 .BLKB 2
00000012 00028 P.AAC: .LONG 18
00000000 0002C .ADDRESS P.AAD
```


54 55 4F 4E 49 47 4F 4C 08 00030 P.AAE: .ASCII <8>\LOGINOUT\ ;

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 ALLOCATE_MEMORY
.EXTRN COMPLETE_JOB, CREATE_SRQ_RECORD
.EXTRN DEALLOCATE_MEMORY
.EXTRN ENTER_PROCESS_DATA
.EXTRN FIND_PROCESS_DATA
.EXTRN PAUSE_SYMBIONT_TASK
.EXTRN READ_RECORD, RELEASE_RECORD
.EXTRN RESET_SYMBIONT_STREAM
.EXTRN RESUME_SYMBIONT_TASK
.EXTRN REWRITE_RECORD, START_SYMBIONT_TASK
.EXTRN STOP_SYMBIONT_TASK
.EXTRN UPDATE_GETQUI_DATA
.EXTRN ALL_PRIVILEGES, NLAO_DESC
.EXTRN EXSEPID_TO_PCB
.EXTRN SCH\$FORCEEXIT, SYSS\$FAO
.EXTRN SYSS\$CREPRC

			00FC 00000	.ENTRY	START EXECUTION, Save R2,R3,R4,R5,R6,R7	: 1168
	57	00000000G	00 9E 00002	MOVAB	SYSS\$FAO, R7	
	56	00000000'	EF 9E 00009	MOVAB	JBC_UIC, R6	
	5E	A0	AE 9E 00010	MOVAB	-96(SP), SP	
	52	10	AC D0 00014	MOVL	SJH, R2	: 1226
	53	08	AC D0 00018	MOVL	SMQ, R3	: 1205
	03	0C	A3 E8 0001C	BLBS	12(R3), 1\$	
			00AD 31 00020	BRW	7\$	
	6E		0F D0 00023	MOVL	#15, PRCNAM_DESC	: 1220
04	AE	08	AE 9E 00026	MOVAB	PRCNAM_BUFFER, PRCNAM_DESC+4	: 1221
		08	A2 DD 0002B	PUSHL	8(R2)	: 1226
		04	AE 9F 0002E	PUSHAB	PRCNAM_DESC	
		08	AE 9F 00031	PUSHAB	PRCNAM_DESC	
		9C	AF 9F 00034	PUSHAB	P.AAA	
	67		04 FB 00037	CALLS	#4, SYSS\$FAO	
18	AE		3F D0 0003A	MOVL	#63, IMAGE_DESC	: 1231
1C	AE	20	AE 9E 0003E	MOVAB	IMAGE_BUFFER, IMAGE_DESC+4	: 1232
		00B0	C2 95 00043	TSTB	176(R2)	: 1239
			06 12 00047	BNEQ	2\$	
	50	AB	AF 9E 00049	MOVAB	P.AAE, R0	
			05 11 0004D	BRB	3\$	
	50	00B0	C2 9E 0004F	MOVAB	176(R2), R0	
			50 DD 00054	PUSHL	R0	
		1C	AE 9F 00056	PUSHAB	IMAGE_DESC	
		20	AE 9F 00059	PUSHAB	IMAGE_DESC	
		90	AF 9F 0005C	PUSHAB	P.AAC	
	67		04 FB 0005F	CALLS	#4, SYSS\$FAO	
	55		10 D0 00062	MOVL	#16, STSFLG	: 1244
03	0E	A3	05 E0 00065	BBS	#5, 14(R3), 4\$	: 1245
		55	04 88 0006A	BISB2	#4, STSFLG	

54		6E 9E 0006D 4\$:	MOVAB	PRCNAM_DESC, PRCNAM	: 1250
		7E D4 00070 5\$:	CLRL	-(SP)	: 1266
		55 DD 00072	PUSHL	STSFLG	
		7E D4 00074	CLRL	-(SP)	
		66 DD 00076	PUSHL	JBC UIC	
7E	0114	C3 9A 00078	MOVZBL	276(R3), -(SP)	
		54 DD 0007D	PUSHL	PRCNAM	
		7E D4 0007F	CLRL	-(SP)	
	00000000G	EF 9F 00081	PUSHAB	ALL PRIVILEGES	
	20	AE 9F 00087	PUSHAB	PRCNAM_DESC	
	24	AE 9F 0008A	PUSHAB	PRCNAM_DESC	
	00000000G	EF 9F 0008D	PUSHAB	NLA0_DESC	
	44	AE 9F 00093	PUSHAB	IMAGE_DESC	
	0168	C2 9F 00096	PUSHAB	360(R2)	
00000000G	00	0D FB 0009A	CALLS	#13, SYSS\$CREPRC	
	0F	50 E8 000A1	BLBS	STATUS, 6\$	: 1269
0094	8F	50 B1 000A4	CMPW	STATUS, #148	: 1277
		67 12 000A9	BNEQ	10\$	
		54 D5 000AB	TSTL	PRCNAM	
		63 13 000AD	BEQL	10\$	
		54 D4 000AF	CLRL	PRCNAM	: 1279
		BD 11 000B1	BRB	5\$	: 1269
	0C	AC DD 000B3 6\$:	PUSHL	SJH_N	: 1289
	04	AC DD 000B6	PUSHL	SMQ_N	
	0168	C2 DD 000B9	PUSHL	360(R2)	
		01 DD 000BD	PUSHL	#1	
00000000G	EF	04 FB 000BF	CALLS	#4, ENTER_PROCESS_DATA	
	FF28	C6 D6 000C6	INCL	QUEUE_REFERENCE_COUNT	: 1290
10	A2	08 88 000CA	BISB2	#8, 16(R2)	: 1291
		37 11 000CE	BRB	9\$	: 1205
14	A3	C6 7D 000D0 7\$:	MOVQ	CUR TIME, 20(R3)	: 1302
	54	C2 D0 000D6	MOVL	236(R2), SQR_N	: 1303
		05 12 000DB	BNEQ	8\$	: 1304
	54	C2 D0 000DD	MOVL	244(R2), SQR_N	
	00F4	54 DD 000E2 8\$:	PUSHL	SQR_N	: 1305
00000000G	EF	01 FB 000E4	CALLS	#1, READ_RECORD	
		50 DD 000EB	PUSHL	SQR	: 1306
		14 BB 000ED	PUSHR	#*M<R2,R4>	
	0C	AC DD 000EF	PUSHL	SJH_N	
		53 DD 000F2	PUSHL	R3	
	04	AC DD 000F4	PUSHL	SMQ_N	
00000000G	EF	06 FB 000F7	CALLS	#6, START_SYMBIONT_TASK	
		54 DD 000FE	PUSHL	SQR_N	: 1307
00000000G	EF	01 FB 00100	CALLS	#1, RELEASE_RECORD	
	00DC	C2 7C 00107 9\$:	CLRQ	220(R2)	: 1313
	00E4	C2 D4 0010B	CLRL	228(R2)	: 1315
	50	01 D0 0010F	MOVL	#1, R0	: 1317
		04 00112 10\$:	RET		

; Routine Size: 275 bytes, Routine Base: CODE + 0039


```
281 1318 1 GLOBAL ROUTINE START_EXECUTOR_JOB(SMQ_N,SMQ,SJH_N,SJH): NOVALUE=
282 1319 1
283 1320 1 ++
284 1321 1
285 1322 1 FUNCTIONAL DESCRIPTION:
286 1323 1 This routine starts a job on an executor.
287 1324 1
288 1325 1 INPUT PARAMETERS:
289 1326 1 SMQ_N - Record number of SMQ.
290 1327 1 SMQ - Pointer to SMQ.
291 1328 1 SJH_N - Record number of SJH.
292 1329 1 SJH - Pointer to SJH.
293 1330 1
294 1331 1 IMPLICIT INPUTS:
295 1332 1 NONE
296 1333 1
297 1334 1 OUTPUT PARAMETERS:
298 1335 1 NONE
299 1336 1
300 1337 1 IMPLICIT OUTPUTS:
301 1338 1 NONE
302 1339 1
303 1340 1 ROUTINE VALUE:
304 1341 1 NONE
305 1342 1
306 1343 1 SIDE EFFECTS:
307 1344 1 NONE
308 1345 1
309 1346 1 --
310 1347 1
311 1348 2 BEGIN
312 1349 2 MAP
313 1350 2 SMQ: REF BBLOCK; ! Pointer to SMQ
314 1351 2 SJH: REF BBLOCK; ! Pointer to SJH
315 1352 2 LOCAL
316 1353 2 STATUS; ! Status return
317 1354 2
318 1355 2
319 1356 2 ! If this is a remote queue, enter a remote request.
320 1357 2
321 1358 3 IF SYSID_NEQ(THIS_SYSID, SMQ[SMQ$T_SYSID])
322 1359 2 THEN
323 1360 2 STATUS = CREATE_SRQ_RECORD(SRQ$K_START_JOB, .SMQ_N, .SMQ, .SJH_N, .SJH)
324 1361 2 ELSE
325 1362 2 STATUS = START_EXECUTION(.SMQ_N, .SMQ, .SJH_N, .SJH);
326 1363 2
327 1364 2
328 1365 2 IF .STATUS OR .STATUS EQL 0
329 1366 2 THEN
330 1367 3 BEGIN
331 1368 3 LOCAL
332 1369 3 SJH_NP, ! Record number of predecessor of SJH
333 1370 3 SJH_P: REF BBLOCK; ! Pointer to predecessor of SJH
334 1371 3
335 1372 3
336 1373 3 ! Enqueue the job to the current queue.
337 1374 3
```

```

: 338 1375 3 IF .SMQ[SMQ$L_CURRENT_LIST] EQL 0
: 339 1376 3 THEN
: 340 1377 3 SMQ[SMQ$L_CURRENT_LIST] = .SJH_N
: 341 1378 3 ELSE
: 342 1379 4 BEGIN
: 343 1380 4 SJH_P = READ_RECORD(SJH_NP = .SMQ[SMQ$L_CURRENT_LIST_END]);
: 344 1381 4 SJH_P[SYMSL_LINK] = .SJH_N;
: 345 1382 4 REWRITE_RECORD(.SJH_NP);
: 346 1383 3 END;
: 347 1384 3 SMQ[SMQ$L_CURRENT_LIST_END] = .SJH_N;
: 348 1385 3 SMQ[SMQ$B_CURRENT_JOB_COUNT] = .SMQ[SMQ$B_CURRENT_JOB_COUNT] + 1;
: 349 1386 3 SJH[SYMSL_LINK] = 0;
: 350 1387 3 FLAGS[FLAGS_V_INVALID_SJH] = FALSE;
: 351 1388 3 END
: 352 1389 2 ELSE
: 353 1390 2 BEGIN
: 354 1391 2 ! Fail the job.
: 355 1392 2 !
: 356 1393 2 !
: 357 1394 2 SJH[SJH$L_CONDITION_1] = .STATUS;
: 358 1395 2 COMPLETE_JOB(.SJH_N, .SJH, .SMQ, 0);
: 359 1396 2 FLAGS[FLAGS_V_INVALID_SJH] = TRUE;
: 360 1397 2 END;
: 361 1398 1 END;
```

			007C	00000	.ENTRY	START_EXECUTOR_JOB, Save R2,R3,R4,R5,R6	: 1318
	56	00000000	EF	9E 00002	MOVAB	FLAGS, R6	
	54	10	AC	D0 00009	MOVL	SJH, R4	: 1360
	55	0C	AC	D0 0000D	MOVL	SJH_N, R5	
	52	08	AC	D0 00011	MOVL	SMQ, R2	: 1358
0106	C2	08	A6	D1 00015	CMPL	THIS_SYSID, 262(R2)	
			08	12 0001B	BNEQ	1\$	
010A	C2	0C	A6	B1 0001D	CMPL	THIS_SYSID+4, 266(R2)	
			12	13 00023	BEQL	2\$	
			54	DD 00025	PUSHL	R4	: 1360
			24	BB 00027	PUSHR	#*M<R2,R5>	
		04	AC	DD 00029	PUSHL	SMQ_N	
			01	DD 0002C	PUSHL	#1	
00000000G	EF		05	FB 0002E	CALLS	#5, CREATE_SRQ_RECORD	
			0C	11 00035	BRB	3\$	
			54	DD 00037	PUSHL	R4	: 1362
			24	BB 00039	PUSHR	#*M<R2,R5>	
		04	AC	DD 0003B	PUSHL	SMQ_N	
FEAA	CF		04	FB 0003E	CALLS	#4, START_EXECUTION	
	04		50	E6 00043	BLBS	STATUS, 4\$	: 1365
			50	D5 00046	TSTL	STATUS	
			32	12 00048	BNEQ	7\$	
		48	A2	D5 0004A	TSTL	72(R2)	: 1375
			06	12 0004D	BNEQ	5\$	
48	A2		55	D0 0004F	MOVL	R5, 72(R2)	: 1377
			19	11 00053	BRB	6\$	
	53	4C	A2	D0 00055	MOVL	76(R2), SJH_NP	: 1380
			53	DD 00059	PUSHL	SJH_NP	

EXECUTOR
V04-002

Executor control

L 4
16-Sep-1984 00:03:38
14-Sep-1984 12:37:01

VAX-11 Bliss-32 V4.0-742
[JOBCTL.SRC]EXECUTOR.B32;2

Page 17
(4)

00000000G	EF		01	FB	0005B	CALLS	#1, READ_RECORD	:	
	60		55	D0	00062	MOVL	R5, (SJH_P)	:	1381
			53	DD	00065	PUSHL	SJH_NP	:	1382
00000000G	EF		01	FB	00067	CALLS	#1, REWRITE_RECORD	:	
4C	A2		55	D0	0006E	MOVL	R5, 76(R2)	:	1384
		0115	C2	96	00072	INCB	277(R2)	:	1385
			64	D4	00076	CLRL	(R4)	:	1386
	66		20	8A	00078	BICB2	#32, FLAGS	:	1387
				04	0007B	RET		:	1365
00DC	C4		50	D0	0007C	MOVL	STATUS, 220(R4)	:	1394
			7E	D4	00081	CLRL	-(SP)	:	1395
			52	DD	00083	PUSHL	R2	:	
			54	DD	00085	PUSHL	R4	:	
			55	DD	00087	PUSHL	R5	:	
00000000G	EF		04	FB	00089	CALLS	#4, COMPLETE_JOB	:	
	66		20	88	00090	BISB2	#32, FLAGS	:	1396
			04	00093	RET			:	1398

; Routine Size: 148 bytes, Routine Base: CODE + 014C

```

363 1399 1 ROUTINE DELETE_PROCESS_KERNEL(CTX)=
364 1400 1
365 1401 1 ++
366 1402 1
367 1403 1 FUNCTIONAL DESCRIPTION:
368 1404 1 This routine causes a process to exit with a specified exit status
369 1405 1 by using SCH$FORCEDEEXIT to exit each access mode in turn. This routine
370 1406 1 executes in kernel mode and partially at IPL$_SYNCH.
371 1407 1
372 1408 1 INPUT PARAMETERS:
373 1409 1 CTX - Context block.
374 1410 1
375 1411 1 IMPLICIT INPUTS:
376 1412 1 NONE
377 1413 1
378 1414 1 OUTPUT PARAMETERS:
379 1415 1 NONE
380 1416 1
381 1417 1 IMPLICIT OUTPUTS:
382 1418 1 NONE
383 1419 1
384 1420 1 ROUTINE VALUE:
385 1421 1 Completion status.
386 1422 1
387 1423 1 SIDE EFFECTS:
388 1424 1 AST queued to process.
389 1425 1
390 1426 1 --
391 1427 1
392 1428 2 BEGIN
393 1429 2 MAP
394 1430 2 CTX: REF VECTOR; ! Pointer to context block
395 1431 2 REGISTER
396 1432 2 EPID, ! EPID of process to delete
397 1433 2 PCB: REF BBLOCK, ! Pointer to PCB
398 1434 2 STATUS; ! Exit status
399 1435 2 STACKLOCAL
400 1436 2 MODE; ! Access mode
401 1437 2
402 1438 2
403 1439 2 ! Load parameters into registers or onto the kernel stack to avoid page fault
404 1440 2 ! at elevated IPL.
405 1441 2
406 1442 2 EPID = .CTX[0];
407 1443 2 MODE = .CTX[1];
408 1444 2 STATUS = .CTX[2];
409 1445 2
410 1446 2
411 1447 2 ! Raise IPL to SYNCH.
412 1448 2
413 1449 2 MTPR(LOCK_IPL, PR$_IPL);
414 1450 2
415 1451 2
416 1452 2 ! Do the real work.
417 1453 2
418 1454 2 PCB = EXE$EPID_TO_PCB(.EPID);
419 1455 2 IF
```



```
: 420      1456 3 BEGIN
: 421      1457 3 IF .PCB NEQ 0
: 422      1458 3 THEN
: 423      1459 3     .PCB[PCB$V_BATCH]
: 424      1460 3 ELSE
: 425      1461 3     FALSE
: 426      1462 3 END
: 427      1463 2 THEN
: 428      1464 2 BEGIN
: 429      1465 2
: 430      1466 2     ! Force the specified access mode to exit with the specified status.
: 431      1467 2     !
: 432      1468 2     SCH$FORCEDEXIT(MODE, .STATUS, .PCB);
: 433      1469 2
: 434      1470 2
: 435      1471 2     ! Lower IPL.
: 436      1472 2     !
: 437      1473 2     MTPR(%REF(0), PR$_IPL);
: 438      1474 2
: 439      1475 2
: 440      1476 2     ! If we delivered to kernel mode, clean up. Otherwise set a kernel
: 441      1477 2     ! mode timer to reenter this routine in two minutes for the next
: 442      1478 2     ! higher access mode.
: 443      1479 2     !
: 444      1480 2 IF .MODE NEQ PSL$C_KERNEL
: 445      1481 2 THEN
: 446      1482 4 BEGIN
: 447      1483 4     CTX[1] = .MODE - 1;
: 448      1484 4     $SETIMR(
: 449      1485 4         DAYTIM=UPLIT(-1200000000, -1),      ! 2 minute delta time
: 450      1486 4         ASTADR=DELETE_PROCESS_KERNEL,
: 451      1487 4         REQIDT=.CTX);
: 452      1488 4     END
: 453      1489 3 ELSE
: 454      1490 3     DEALLOCATE_MEMORY(.CTX);
: 455      1491 3
: 456      1492 3
: 457      1493 3     SSS$_NORMAL
: 458      1494 3     END
: 459      1495 2 ELSE
: 460      1496 2 BEGIN
: 461      1497 2
: 462      1498 2     ! Lower IPL.
: 463      1499 2     !
: 464      1500 2     MTPR(%REF(0), PR$_IPL);
: 465      1501 2
: 466      1502 2
: 467      1503 2     ! Clean up and return.
: 468      1504 2     !
: 469      1505 2     DEALLOCATE_MEMORY(.CTX);
: 470      1506 2     SSS$_NONEXPR
: 471      1507 2     END
: 472      1508 1 END;
```

FFFFFFFF B8797400 001E0 P.AAF: .LONG -1200000000, -1

.EXTRN SYS\$SETIMR

007C 00000 DELETE_PROCESS_KERNEL:

56	00000000G	EF	9E	00002	.WORD	Save R2,R3,R4,R5,R6	: 1399
55	04	AC	D0	00009	MOVAB	DEALLOCATE_MEMORY, R6	: 1442
50		65	D0	0000D	MOVL	CTX, R5	: 1443
	04	A5	DD	00010	MOVL	(R5), EPID	: 1444
52	08	A5	DD	00013	PUSHL	4(R5)	: 1449
12	0000V	CF	DA	00017	MOVL	8(R5), STATUS	: 1454
	00000000G	00	16	0001C	MTPR	LOCK IPL, #18	: 1457
51		50	D0	00022	JSB	EX\$EPID_TO_PCB	: 1459
		3C	13	00025	MOVL	R0, PCB	: 1468
37	25	A1	06	E1	BEQL	3\$	: 1473
50		6E	9E	0002C	BBC	#6, 37(PCB), 3\$	: 1480
54		51	D0	0002F	MOVAB	MODE, R0	: 1483
53		52	D0	00032	MOVL	PCB, R4	: 1487
	00000000G	00	16	00035	MOVL	STATUS, R3	
12		00	DA	0003B	JSB	SCH\$FORCEDEXIT	
		6E	D5	0003E	MTPR	#0, #18	
		18	13	00040	TSTL	MODE	
04	A5	6E	01	C3	BEQL	1\$	
		55	DD	00047	SUBL3	#1, MODE, 4(R5)	
	B4	AF	9F	00049	PUSHL	R5	
	A9	AF	9F	0004C	PUSHAB	DELETE_PROCESS_KERNEL	
		7E	D4	0004F	PUSHAB	P.AAF	
	00000000G	00	04	FB	CLRL	-(SP)	
		05	11	00058	CALLS	#4, SYS\$SETIMR	
		55	DD	0005A	BRB	2\$	: 1480
66		01	FB	0005C	PUSHL	R5	: 1490
50		01	D0	0005F	CALLS	#1, DEALLOCATE_MEMORY	: 1464
			04	00062	MOVL	#1, R0	: 1500
12		00	DA	00063	RET		: 1505
		55	DD	00066	MTPR	#0, #18	: 1496
66		01	FB	00068	PUSHL	R5	: 1508
50	08E8	8F	3C	0006B	CALLS	#1, DEALLOCATE_MEMORY	
		04	00070	MOVZWL	#2280, R0		
				RET			

; Routine Size: 113 bytes, Routine Base: CODE + 01E8

EXECUTOR
V04-002

Executor control

C 5
16-Sep-1984 00:03:38
14-Sep-1984 12:37:01

VAX-11 Bliss-32 V4.0-742
[JOBCTL.SRC]EXECUTOR.B32;2

Page 21
(6)

```
: 474      1509 1 OWN
: 475      1510 1      LOCK_IPL:      PSECT(CODE) INITIAL(IPL$_SYNCH);
: 476      1511 1
: 477      1512 1
: 478      L 1513 1 %IF LOCK_IPL + 4 - DELETE_PROCESS_KERNEL GTR 512
: 479      1514 1 %THEN %ERROR('Possible page fault at elevated IPL') %FI
```



```
: 481      1515 1 GLOBAL ROUTINE ABORT_EXECUTION(SMQ_N,SMQ,SJH_N,SJH)=
: 482      1516 1
: 483      1517 1 ++
: 484      1518 1
: 485      1519 1 FUNCTIONAL DESCRIPTION:
: 486      1520 1     This routine performs the actual communication that aborts a job on an
: 487      1521 1     executor.
: 488      1522 1
: 489      1523 1 INPUT PARAMETERS:
: 490      1524 1     SMQ_N           - Record number of SMQ.
: 491      1525 1     SMQ           - Pointer to SMQ.
: 492      1526 1     SJH_N           - Record number of SJH.
: 493      1527 1     SJH           - Pointer to SJH.
: 494      1528 1
: 495      1529 1 IMPLICIT INPUTS:
: 496      1530 1     NONE
: 497      1531 1
: 498      1532 1 OUTPUT PARAMETERS:
: 499      1533 1     NONE
: 500      1534 1
: 501      1535 1 IMPLICIT OUTPUTS:
: 502      1536 1     NONE
: 503      1537 1
: 504      1538 1 ROUTINE VALUE:
: 505      1539 1     Completion status.
: 506      1540 1
: 507      1541 1 SIDE EFFECTS:
: 508      1542 1     NONE
: 509      1543 1
: 510      1544 1 --
: 511      1545 1
: 512      1546 2 BEGIN
: 513      1547 2 MAP
: 514      1548 2     SMQ:           REF BBLOCK,      ! Pointer to SMQ
: 515      1549 2     SJH:           REF BBLOCK;      ! Pointer to SJH
: 516      1550 2 LOCAL
: 517      1551 2     STATUS;           ! Status return
: 518      1552 2
: 519      1553 2
: 520      1554 2 IF .SMQ[SMQ$V_BATCH]
: 521      1555 2 THEN
: 522      1556 3     BEGIN
: 523      1557 3     LOCAL
: 524      1558 3     CTX:           REF VECTOR,      ! Pointer to context area
: 525      1559 3     ARGVLIST:       VECTOR[2];      ! $CMKRN argument list
: 526      1560 3
: 527      1561 3
: 528      1562 3     CTX = ALLOCATE MEMORY();
: 529      1563 3     CTX[0] = .SJH[SJH$EXECUTOR_PID];
: 530      1564 3     CTX[1] = PSL$C_USER;
: 531      1565 3     CTX[2] = JBC$_JOBABORT OR STS$K_ERROR;
: 532      1566 3     IF .SJH[SJH$V_REQUEUE] THEN CTX[2] = JBC$_JOBREQUEUE OR STS$K_ERROR;
: 533      1567 3     ARGVLIST[0] = 1;
: 534      1568 3     ARGVLIST[1] = .CTX;
: 535      1569 3     STATUS = $CMKRN(ROUTIN=DELETE_PROCESS_KERNEL, ARGVLIST=ARGVLIST);
: 536      1570 3     IF NOT .STATUS THEN RETURN .STATUS;
: 537      1571 3     END
```



```
: 538
: 539
: 540
: 541
: 542
: 543
: 544
: 545
: 546

1572 2 ELSE
1573 2 STOP_SYMBIONT_TASK(.SMQ_N, .SMQ, .SJH_N, .SJH);
1574 2
1575 2
1576 2 ! Update SJH.
1577 2
1578 2 SJH[SJH$V_ABORTED] = TRUE;
1579 2 SSS_NORMAL
1580 1 END;
```

```
00000008 00259 .BLKB 3
00000008 0025C LOCK_IPL: .LONG 8
                                .EXTRN SYSSCMKRNL
                                .ENTRY ABORT_EXECUTION, Save R2
                                SUBL2 #8, SP
                                MOVL SJH, R2
                                MOVL SMQ, R0
                                BLBC 12(R0), 2$
                                CALLS #0, ALLOCATE_MEMORY
                                MOVL 360(R2), (CTX)
                                MOVL #3, 4(CTX)
                                MOVL #295042, 8(CTX)
                                BLBC 17(R2), 1$
                                MOVL #295138, 8(CTX)
                                MOVL #1, ARGLST
                                MOVL CTX, ARGLST+4
                                PUSHL SP
                                PUSHAB DELETE_PROCESS_KERNEL
                                CALLS #2, SYSSCMKRNL
                                BLBS STATUS, 3$
                                RET
                                PUSHL R2
                                PUSHL SJH_N
                                PUSHL R0
                                PUSHL SMQ_N
                                CALLS #4, STOP_SYMBIONT_TASK
                                BISB2 #1, 16(R2)
                                MOVL #1, R0
                                RET

0004 00000
08 C2 00002
5E 10 AC D0 00005
50 08 AC D0 00009
3C 0C A0 E9 0000D
00000000G EF 00 FB 00011
60 0168 C2 D0 00018
04 A0 00048082 03 D0 0001D
08 A0 00048082 8F D0 00021
08 08 000480E2 A2 E9 00029
08 A0 000480E2 8F D0 0002D
04 6E 01 D0 00035 1$:
AE AE 50 D0 00038
FF46 5E DD 0003C
00000000G 00 CF 9F 0003E
12 02 FB 00042
50 50 E8 00049
04 0004C
52 DD 0004D 2$:
0C AC DD 0004F
50 DD 00052
04 AC DD 00054
00000000G EF 04 FB 00057
10 A2 01 88 0005E 3$:
50 01 D0 00062
04 00065
```

; Routine Size: 102 bytes, Routine Base: CODE + 0260

0000 00000		.ENTRY		ABORT EXECUTOR_JOB, Save nothing		
50	08	AC	D0	00002	MOVL	SMQ, R0
0106	C0 00000000'	EF	D1	00006	CMPL	THIS_SYSID, 262(R0)
		0B	12	0000F	BNEQ	1\$
010A	C0 00000000'	EF	B1	00011	CMPL	THIS_SYSID+4, 266(R0)

1581
1622

EXECUTOR
V04-002

Executor control

G 5
16-Sep-1984 00:03:38
14-Sep-1984 12:37:01

VAX-11 Bliss-32 V4.0-742
[JOBCTL.SRC]EXECUTOR.B32;2

Page 25
(8)

	7E	0C	13	13	0001A	1\$:	BEQL	2\$		
			AC	7D	0001C		MOVQ	SJH_N, -(SP)		1624
		04	50	DD	00020		PUSHL	R0		
			AC	DD	00022		PUSHL	SMQ_N		
00000000G	EF		02	DD	00025		PUSHL	#2		
			05	FB	00027		CALLS	#5, CREATE_SRQ_RECORD		
	7E	0C	AC	7D	0002E	2\$:	RET			1626
			50	DD	00033		MOVQ	SJH_N, -(SP)		
		04	AC	DD	00035		PUSHL	R0		
FF5D	CF		04	FB	00038		PUSHL	SMQ_N		
			04	FB	0003D		CALLS	#4, ABORT_EXECUTION		
							RET			1627

; Routine Size: 62 bytes, Routine Base: CODE + 02C6

```
596 1628 1 GLOBAL ROUTINE PAUSE_EXECUTION(SMQ_N,SMQ)=
597 1629 1
598 1630 1 ++
599 1631 1
600 1632 1 FUNCTIONAL DESCRIPTION:
601 1633 1 This routine performs the actual communication that pauses a job on an
602 1634 1 executor.
603 1635 1
604 1636 1 INPUT PARAMETERS:
605 1637 1 SMQ_N - Record number of SMQ.
606 1638 1 SMQ - Pointer to SMQ.
607 1639 1
608 1640 1 IMPLICIT INPUTS:
609 1641 1 NONE
610 1642 1
611 1643 1 OUTPUT PARAMETERS:
612 1644 1 NONE
613 1645 1
614 1646 1 IMPLICIT OUTPUTS:
615 1647 1 NONE
616 1648 1
617 1649 1 ROUTINE VALUE:
618 1650 1 Completion status.
619 1651 1
620 1652 1 SIDE EFFECTS:
621 1653 1 NONE
622 1654 1
623 1655 1 --
624 1656 1
625 1657 2 BEGIN
626 1658 2 MAP
627 1659 2 SMQ: REF BBLOCK; ! Pointer to SMQ
628 1660 2
629 1661 2 LOCAL SJH_N, ! Record number of SJH
630 1662 2 SJH_NS, ! Record number of successor of SJH
631 1663 2 SJH: REF BBLOCK, ! Pointer to SJH
632 1664 2 FINAL_STATUS, ! Final status return
633 1665 2 STATUS; ! Status return
634 1666 2
635 1667 2
636 1668 2 FINAL_STATUS = SS$NORMAL;
637 1669 2
638 1670 2
639 1671 2 IF .SMQ[SMQ$V_BATCH]
640 1672 2 THEN
641 1673 3 BEGIN
642 1674 3
643 1675 3 ! Issue a suspend for all currently executing jobs.
644 1676 3
645 1677 3 SJH_N = .SMQ[SMQ$L_CURRENT_LIST];
646 1678 3 WHILE .SJH_N NEQ 0 DO
647 1679 4 BEGIN
648 1680 4 SJH = READ_RECORD(.SJH_N);
649 1681 4 STATUS = $SUSPND(PIDADR=SJH[SJH$L_EXECUTOR_PID]);
650 1682 4 IF NOT .STATUS AND .FINAL_STATUS THEN FINAL_STATUS = .STATUS;
651 1683 4 SJH_NS = .SJH[SYMS$L_LINK];
652 1684 4 RELEASE_RECORD(.SJH_N);
```


EXECUTOR
V04-002

Executor control

I 5
16-Sep-1984 00:03:38
14-Sep-1984 12:37:01

VAX-11 Bliss-32 V4.0-742
[JOBCTL.SRC]EXECUTOR.B32;2

Page 27
(9)

```
: 653      1685      4      SJH_N = .SJH_NS;  
: 654      1686      3      END;  
: 655      1687      3      SMQ[SMQ$V_PAUSED] = TRUE;  
: 656      1688      3      END  
: 657      1689      2      ELSE  
: 658      1690      2      PAUSE_SYMBIONT_TASK(.SMQ_N, .SMQ);  
: 659      1691      2  
: 660      1692      2  
: 661      1693      2      .FINAL_STATUS  
: 662      1694      1      END;
```

				.EXTRN	SYSSUSPND	
			00FC 00000	.ENTRY	PAUSE EXECUTION, Save R2,R3,R4,R5,R6,R7	: 1628
	56		01 DO 00002	MOVL	#1, FINAL_STATUS	: 1668
	52	08	AC DO 00005	MOVL	SMQ, R2	: 1671
	42	0C	A2 E9 00009	BLBC	12(R2), 4\$	
	54	48	A2 DO 0000D	MOVL	72(R2), SJH_N	: 1677
			36 13 00011 1\$:	BEQL	3\$	: 1678
			54 DD 00013	PUSHL	SJH_N	: 1680
00000000G	EF		01 FB 00015	CALLS	#1, READ_RECORD	
	53		50 DO 0001C	MOVL	R0, SJH	
			7E D4 0001F	CLRL	-(SP)	: 1681
		0168	C3 9F 00021	PUSHAB	360(SJH)	
00000000G	00		02 FB 00025	CALLS	#2, SYSSUSPND	
	55		50 DO 0002C	MOVL	R0, STATUS	
	06		55 E8 0002F	BLBS	STATUS, 2\$	: 1682
	03		56 E9 00032	BLBC	FINAL_STATUS, 2\$	
	56		55 DO 00035	MOVL	STATUS, FINAL_STATUS	
	57		63 DO 00038 2\$:	MOVL	(SJH), SJH_NS	: 1683
			54 DD 0003B	PUSHL	SJH_N	: 1684
00000000G	EF		01 FB 0003D	CALLS	#1, RELEASE_RECORD	
	54		57 DO 00044	MOVL	SJH_NS, SJH_N	: 1685
			C8 11 00047	BRB	1\$	: 1678
10	A2		04 88 00049 3\$:	BISB2	#4, 16(R2)	: 1687
			0C 11 0004D	BRB	5\$	: 1671
			52 DD 0004F 4\$:	PUSHL	R2	: 1690
		04	AC DD 00051	PUSHL	SMQ_N	
00000000G	EF		02 FB 00054	CALLS	#2, PAUSE_SYMBIONT_TASK	
	50		56 DO 0005B 5\$:	MOVL	FINAL_STATUS, R0	: 1694
			04 0005E	RET		

; Routine Size: 95 bytes, Routine Base: CODE + 0304

```

664 1695 1 GLOBAL ROUTINE RESUME_EXECUTION(SMQ_N,SMQ,FLAGS,ALIGNMENT_PAGES,RELATIVE_PAGE,SEARCH_LENGTH,SEARCH_ADDRESS)=
665 1696 1
666 1697 1 !++
667 1698 1
668 1699 1 FUNCTIONAL DESCRIPTION:
669 1700 1 This routine performs the actual communication that resumes a job on an
670 1701 1 executor.
671 1702 1
672 1703 1 INPUT PARAMETERS:
673 1704 1 SMQ_N - Record number of SMQ.
674 1705 1 SMQ - Pointer to SMQ.
675 1706 1 FLAGS - Resume control flags.
676 1707 1 ALIGNMENT_PAGES - Number of alignment pages (or 0).
677 1708 1 RELATIVE_PAGE - Relative page offset (or 0).
678 1709 1 SEARCH_LENGTH - Descriptor for search string (or 0).
679 1710 1 SEARCH_ADDRESS -
680 1711 1
681 1712 1 IMPLICIT INPUTS:
682 1713 1 NONE
683 1714 1
684 1715 1 OUTPUT PARAMETERS:
685 1716 1 NONE
686 1717 1
687 1718 1 IMPLICIT OUTPUTS:
688 1719 1 NONE
689 1720 1
690 1721 1 ROUTINE VALUE:
691 1722 1 Completion status.
692 1723 1
693 1724 1 SIDE EFFECTS:
694 1725 1 NONE
695 1726 1
696 1727 1 --
697 1728 1
698 1729 2 BEGIN
699 1730 2 MAP
700 1731 2 SMQ: REF BBLOCK; ! Pointer to SMQ
701 1732 2 LOCAL
702 1733 2 SJH_N, ! Record number of SJH
703 1734 2 SJH_NS, ! Record number of successor of SJH
704 1735 2 SJH: REF BBLOCK, ! Pointer to SJH
705 1736 2 FINAL_STATUS, ! Final status return
706 1737 2 STATUS; ! Status return
707 1738 2
708 1739 2
709 1740 2 FINAL_STATUS = SS$_NORMAL;
710 1741 2
711 1742 2
712 1743 2 IF .SMQ[SMQ$V_BATCH]
713 1744 2 THEN
714 1745 2 BEGIN
715 1746 2
716 1747 2 ! Issue a resume for all currently executing jobs.
717 1748 2 !
718 1749 2 SJH_N = .SMQ[SMQ$L_CURRENT_LIST];
719 1750 2 WHILE .SJH_N NEQ 0 DO
720 1751 2 BEGIN
```



```
: 721      1752  4      SJH = READ RECORD(.SJH_N);
: 722      1753  4      STATUS = $RESUME(PIDADR=SJH[SJH$ EXECUTOR_PID]);
: 723      1754  4      IF NOT .STATUS AND .FINAL_STATUS THEN FINAL_STATUS = .STATUS;
: 724      1755  4      SJH_NS = .SJH[SYMS$ LINK];
: 725      1756  4      RELEASE_RECORD(.SJH_N);
: 726      1757  4      SJH_N = .SJH_NS;
: 727      1758  3      END;
: 728      1759  3      SMQ[SMQ$V_PAUSED] = FALSE;
: 729      1760  3      END
: 730      1761  2  ELSE
: 731      1762  2      RESUME SYMBIONT_TASK(
: 732      1763  2          .SMQ_N, .SMQ,
: 733      1764  2          .FLAGS, .ALIGNMENT_PAGES, .RELATIVE_PAGE,
: 734      1765  2          .SEARCH_LENGTH, .SEARCH_ADDRESS);
: 735      1766  2
: 736      1767  2
: 737      1768  2  .FINAL_STATUS
: 738      1769  1  END;
```

				.EXTRN	SYSS\$RESUME	
			00FC 00000	.ENTRY	RESUME EXECUTION, Save R2,R3,R4,R5,R6,R7	: 1695
	56		01 D0 00002	MOVL	#1, FINAL_STATUS	: 1740
	52	08	AC D0 00005	MOVL	SMQ, R2	: 1743
	42	0C	A2 E9 00009	BLBC	12(R2), 4\$	
	54	48	A2 D0 0000D	MOVL	72(R2), SJH_N	: 1749
			36 13 00011 1\$:	BEQL	3\$	: 1750
			54 DD 00013	PUSHL	SJH_N	: 1752
00000000G	EF		01 FB 00015	CALLS	#1, READ_RECORD	
	53		50 D0 0001C	MOVL	R0, SJH	
			7E D4 0001F	CLRL	-(SP)	: 1753
		0168	C3 9F 00021	PUSHAB	360(SJH)	
00000000G	00		02 FB 00025	CALLS	#2, SYSS\$RESUME	
	55		50 D0 0002C	MOVL	R0, STATUS	
	06		55 E8 0002F	BLBS	STATUS, 2\$	: 1754
	03		56 E9 00032	BLBC	FINAL_STATUS, 2\$	
	56		55 D0 00035	MOVL	STATUS, FINAL_STATUS	
	57		63 D0 00038 2\$:	MOVL	(SJH), SJH_NS	: 1755
			54 DD 0003B	PUSHL	SJH_N	: 1756
00000000G	EF		01 FB 0003D	CALLS	#1, RELEASE_RECORD	
	54		57 D0 00044	MOVL	SJH_NS, SJH_N	: 1757
			C8 11 00047	BRB	1\$	: 1750
10	A2		04 8A 00049 3\$:	BICB2	#4, 16(R2)	: 1759
			17 11 0004D	BRB	5\$	: 1743
	7E	18	AC 7D 0004F 4\$:	MOVQ	SEARCH_LENGTH, -(SP)	: 1765
	7E	10	AC 7D 00053	MOVQ	ALIGNMENT_PAGES, -(SP)	: 1764
		0C	AC DD 00057	PUSHL	FLAGS	
			52 DD 0005A	PUSHL	R2	: 1763
		04	AC DD 0005C	PUSHL	SMQ_N	
00000000G	EF		07 FB 0005F	CALLS	#7, RESUME_SYMBIONT_TASK	
	50		56 D0 00066 5\$:	MOVL	FINAL_STATUS, R0	: 1769
			04 00069	RET		

; Routine Size: 106 bytes, Routine Base: CODE + 0363

EXECUTOR
V04-002

Executor control

L 5
16-Sep-1984 00:03:38
14-Sep-1984 12:37:01

VAX-11 Bliss-32 V4.0-742
[JOBCTL.SRC]EXECUTOR.B32;2

Page 30
(10)

GE
V0


```

: 740      1770 1 GLOBAL ROUTINE RESET_EXECUTOR_QUEUE(SMQ_N,SMQ): NOVALUE=
: 741      1771 1
: 742      1772 1 |++
: 743      1773 1 |
: 744      1774 1 | FUNCTIONAL DESCRIPTION:
: 745      1775 1 |     This routine resets an executor queue.
: 746      1776 1 |
: 747      1777 1 | INPUT PARAMETERS:
: 748      1778 1 |     SMQ_N           - Record number of SMQ.
: 749      1779 1 |     SMQ             - Pointer to SMQ.
: 750      1780 1 |
: 751      1781 1 | IMPLICIT INPUTS:
: 752      1782 1 |     NONE
: 753      1783 1 |
: 754      1784 1 | OUTPUT PARAMETERS:
: 755      1785 1 |     NONE
: 756      1786 1 |
: 757      1787 1 | IMPLICIT OUTPUTS:
: 758      1788 1 |     NONE
: 759      1789 1 |
: 760      1790 1 | ROUTINE VALUE:
: 761      1791 1 |     Completion status.
: 762      1792 1 |
: 763      1793 1 | SIDE EFFECTS:
: 764      1794 1 |     NONE
: 765      1795 1 |
: 766      1796 1 | --
: 767      1797 1 |
: 768      1798 2 BEGIN
: 769      1799 2 MAP
: 770      1800 2     SMQ:           REF BBLOCK;      ! Pointer to SMQ
: 771      1801 2
: 772      1802 2
: 773      1803 2 ! If the queue has a symbiont, send the symbiont a reset.
: 774      1804 2 !
: 775      1805 2 IF .SMQ[SMQ$L_STREAM_SCT] NEQ 0
: 776      1806 2 THEN
: 777      1807 2     RESET_SYMBIONT_STREAM(.SMQ_N, .SMQ);
: 778      1808 2
: 779      1809 2
: 780      1810 2 ! Reinitialize the queue header as appropriate.
: 781      1811 2 !
: 782      1812 2 SMQ[SMQ$L_STATUS] = 0;
: 783      1813 2 SMQ[SMQ$V_STOPPED] = TRUE;
: 784      1814 2 CLEAR TIME(SMQ[SMQ$Q_ACM_BEGTIM]);
: 785      1815 2 SMQ[SMQ$L_ACM_GETCNT] = 0;
: 786      1816 2 SMQ[SMQ$L_ACM_PAGECNT] = 0;
: 787      1817 2 SMQ[SMQ$L_ACM_QIOCNT] = 0;
: 788      1818 2 SMQ[SMQ$L_ACM_SYMCPUTIM] = 0;
: 789      1819 2 SMQ[SMQ$L_STREAM_SCT] = 0;
: 790      1820 2 SMQ[SMQ$B_CURRENT_JOB_COUNT] = 0;
: 791      1821 2 SMQ[SMQ$B_STREAM_INDEX] = 0;
: 792      1822 2
: 793      1823 2
: 794      1824 2 ! Process all current jobs.
: 795      1825 2 !
: 796      1826 2 WHILE .SMQ[SMQ$L_CURRENT_LIST] NEQ 0 DO
```

```

: 797      1827 3 BEGIN
: 798      1828 3 LOCAL
: 799      1829 3     SJH_N,           ! Record number of SJH
: 800      1830 3     SJH:           REF BBLOCK; ! Pointer to SJH
: 801      1831 3
: 802      1832 3
: 803      1833 3 ! Read and unlink the job header.
: 804      1834 3
: 805      1835 3 SJH = READ_RECORD(SJH_N = .SMQ[SMQ$L_CURRENT_LIST]);
: 806      1836 3 UPDATE_GETQUI_DATA(.SJH_N, .SJH);
: 807      1837 3 SMQ[SMQ$L_CURRENT_LIST] = .SJH[SYM$L_LINK];
: 808      1838 3
: 809      1839 3
: 810      1840 3 ! If this is a batch job, delete the process, but clean up data structures
: 811      1841 3 ! and complete the job immediately, as if the process had terminated.
: 812      1842 3
: 813      1843 3 IF .SMQ[SMQ$V_BATCH]
: 814      1844 3 THEN
: 815      1845 4 BEGIN
: 816      1846 4 LOCAL
: 817      1847 4     CTX:           REF VECTOR, ! Pointer to context area
: 818      1848 4     ARGVST:        VECTOR[2]; ! $CMKRN argument list
: 819      1849 4
: 820      1850 4
: 821      1851 4     CTX = ALLOCATE MEMORY();
: 822      1852 4     CTX[0] = .SJH[SJH$L_EXECUTOR_PID];
: 823      1853 4     CTX[1] = PSL$C_USER;
: 824      1854 4     CTX[2] = JBC$_QUERESSET OR STS$K_ERROR;
: 825      1855 4     ARGVST[0] = 1;
: 826      1856 4     ARGVST[1] = .CTX;
: 827      1857 4     $CMKRN(ROUTIN=DELETE_PROCESS_KERNEL, ARGVST=ARGVST);
: 828      1858 4     FIND_PROCESS_DATA(PDE_K_BATCH, .SJH[SJH$L_EXECUTOR_PID], TRUE);
: 829      1859 4     QUEUE_REFERENCE_COUNT = .QUEUE_REFERENCE_COUNT - 1;
: 830      1860 3 END;
: 831      1861 3
: 832      1862 3
: 833      1863 3 SJH[SJH$V_SYSTEM_FAILURE] = TRUE;
: 834      1864 3 COMPLETE_JOB(.SJH_N, .SJH, .SMQ, 0, JBC$_QUERESSET OR STS$K_ERROR);
: 835      1865 2 END;
: 836      1866 2 SMQ[SMQ$L_CURRENT_LIST_END] = 0;
: 837      1867 1 END;
```

			0E1C 00000	.ENTRY	RESET_EXECUTOR_QUEUE, Save R2,R3,R4,R9,R10,-;	1770
					R11	
	5E		08 C2 00002	SUBL2	#8, SP	
	52	08	AC D0 00005	MOVL	SMQ, R2	1805
		00FC	C2 D5 00009	TSTL	252(R2)	
			0C 13 0000D	BEQL	1\$	
			52 DD 0000F	PUSHL	R2	1807
		04	AC DD 00011	PUSHL	SMQ_N	
00000000G	EF		02 FB 00014	CALLS	#2, RESET_SYMBIONT_STREAM	
		10	A2 D4 0001B	CLRL	16(R2)	1812
	11	A2	02 88 0001E	BISB2	#2, 17(R2)	1813

		14	A2	7C	00022	CLRQ	20(R2)	:	1814
		1C	A2	7C	00025	CLRQ	28(R2)	:	1815
		24	A2	7C	00028	CLRQ	36(R2)	:	1817
		00FC	C2	D4	0002B	CLRL	252(R2)	:	1819
		0115	C2	94	0002F	CLRB	277(R2)	:	1820
		0117	C2	94	00033	CLRB	279(R2)	:	1821
		48	A2	D5	00037	TSTL	72(R2)	:	1826
			03	12	0003A	BNEQ	3\$	:	
			0081	31	0003C	BRW	5\$	:	
	54	48	A2	D0	0003F	MOVL	72(R2), SJH_N	:	1835
			54	DD	00043	PUSHL	SJH_N	:	
00000000G	EF		01	FB	00045	CALLS	#1, READ_RECORD	:	
	53		50	D0	0004C	MOVL	R0, SJH	:	
			53	DD	0004F	PUSHL	SJH	:	1836
			54	DD	00051	PUSHL	SJH_N	:	
00000000G	EF		02	FB	00053	CALLS	#2, UPDATE_GETQUI_DATA	:	
	48		63	D0	0005A	MOVL	(SJH), 72(R2)	:	1837
	41	0C	A2	E9	0005E	BLBC	12(R2), 4\$	:	1843
00000000G	EF		00	FB	00062	CALLS	#0, ALLOCATE_MEMORY	:	1851
	60	0168	C3	D0	00069	MOVL	360(SJH), (CTX)	:	1852
	04	A0	03	D0	0006E	MOVL	#3, 4(CTX)	:	1853
	08	A0	8F	D0	00072	MOVL	#295146, 8(CTX)	:	1854
	6E		01	D0	0007A	MOVL	#1, ARGLST	:	1855
	04	AE	50	D0	0007D	MOVL	CTX, ARGLST+4	:	1856
			5E	DD	00081	PUSHL	SP	:	1857
		FD94	CF	9F	00083	PUSHAB	DELETE_PROCESS_KERNEL	:	
00000000G	00		02	FB	00087	CALLS	#2, SYS\$CMKRN	:	
			01	DD	0008E	PUSHL	#1	:	1858
		0168	C3	DD	00090	PUSHL	360(SJH)	:	
			01	DD	00094	PUSHL	#1	:	
00000000G	EF		03	FB	00096	CALLS	#3, FIND_PROCESS_DATA	:	
		00000000'	EF	D7	0009D	DECL	QUEUE_REFERENCE_COUNT	:	1859
	11	A3	8F	88	000A3	BISB2	#64, T7(SJH)	:	1863
		000480EA	8F	DD	000A8	PUSHL	#295146	:	1864
			7E	D4	000AE	CLRL	-(SP)	:	
			52	DD	000B0	PUSHL	R2	:	
			53	DD	000B2	PUSHL	SJH	:	
			54	DD	000B4	PUSHL	SJH_N	:	
00000000G	EF		05	FB	000B6	CALLS	#5, COMPLETE_JOB	:	
		FF77	31	000BD	BRW	2\$		:	1826
		4C	A2	D4	000C0	CLRL	76(R2)	:	1866
			04	000C3	RET			:	1867

; Routine Size: 196 bytes, Routine Base: CODE + 03CD

EXECUTOR
V04-002

Executor control

C 6
16-Sep-1984 00:03:38
14-Sep-1984 12:37:01

VAX-11 Bliss-32 V4.0-742
[JOBCTL.SRC]EXECUTOR.B32;2

Page 34
(12)

: 839 1868 1 END
: 840 1869 0 ELUDOM

PSECT SUMMARY

Name	Bytes	Attributes
COMMON	5024	NOVEC, WRT, RD, NOEXE, NOSHR, LCL, REL, OVR, NOPIC, ALIGN(2)
CODE	1169	NOVEC, NOWRT, RD, EXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)

Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
_\$255\$DUA28:[SYSLIB]LIB.L32;1	18619	55	0	1000	00:01.4

COMMAND QUALIFIERS

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

: Size: 1097 code + 5096 data bytes
: Run Time: 00:22.7
: Elapsed Time: 01:45.5
: Lines/CPU Min: 4933
: Lexemes/CPU-Min: 41514
: Memory Used: 275 pages
: Compilation Complete

