

BBBBBBBBBBBBBB		AAAAAAAAAA	CCCCCCCCCCCC	KKK	KKK	UUU	UUU	PPPPPPPPPPPP
BBBBBBBBBBBBBB		AAAAAAAAAA	CCCCCCCCCCCC	KKK	KKK	UUU	UUU	PPPPPPPPPPPP
BBBBBBBBBBBBBB		AAAAAAAAAA	CCCCCCCCCCCC	KKK	KKK	UUU	UUU	PPPPPPPPPPPP
BBB	BBB	AAA	AAA	CCC	KKK	UUU	UUU	PPP
BBB	BBB	AAA	AAA	CCC	KKK	UUU	UUU	PPP
BBB	BBB	AAA	AAA	CCC	KKK	UUU	UUU	PPP
BBB	BBB	AAA	AAA	CCC	KKK	UUU	UUU	PPP
BBB	BBB	AAA	AAA	CCC	KKK	UUU	UUU	PPP
BBB	BBB	AAA	AAA	CCC	KKK	UUU	UUU	PPP
BBB	BBB	AAA	AAA	CCC	KKK	UUU	UUU	PPP
BBBBBBBBBBBBBB		AAA	AAA	CCC	KKKKKKKKKK	UUU	UUU	PPPPPPPPPPPP
BBBBBBBBBBBBBB		AAA	AAA	CCC	KKKKKKKKKK	UUU	UUU	PPPPPPPPPPPP
BBBBBBBBBBBBBB		AAA	AAA	CCC	KKKKKKKKKK	UUU	UUU	PPPPPPPPPPPP
BBB	BBB	AAAAAAAAAAAAAAAA	CCC	KKK	KKK	UUU	UUU	PPP
BBB	BBB	AAAAAAAAAAAAAAAA	CCC	KKK	KKK	UUU	UUU	PPP
BBB	BBB	AAAAAAAAAAAAAAAA	CCC	KKK	KKK	UUU	UUU	PPP
BBB	BBB	AAA	AAA	CCC	KKK	UUU	UUU	PPP
BBB	BBB	AAA	AAA	CCC	KKK	UUU	UUU	PPP
BBB	BBB	AAA	AAA	CCC	KKK	UUU	UUU	PPP
BBB	BBB	AAA	AAA	CCC	KKK	UUU	UUU	PPP
BBB	BBB	AAA	AAA	CCC	KKK	UUU	UUU	PPP
BBBBBBBBBBBBBB		AAA	AAA	CCCCCCCCCCCC	KKK	KKK	UUUUUUUUUUUUUUUU	PPP
BBBBBBBBBBBBBB		AAA	AAA	CCCCCCCCCCCC	KKK	KKK	UUUUUUUUUUUUUUUU	PPP
BBBBBBBBBBBBBB		AAA	AAA	CCCCCCCCCCCC	KKK	KKK	UUUUUUUUUUUUUUUU	PPP

SSSSSSSS SSSSSSSS SS SS SS SS SSSSSS SSSSSS SS SS SSSSSS SSSSSS	TTTTTTTT TTTTTTTT TT TT TT TT TT TT TT TT TT TT TT	AAAAAA AAAAAA AA AA AA AA AA AA AAAAAA AAAAAA AA AA AA AA	IIIIII IIIIII II II II II II II II II II IIIIII IIIIII	000000 000000 00 00 00 00 00 00 00 00 00 00 00 000000 000000	SSSSSSSS SSSSSSSS SS SS SS SS SSSSSS SSSSSS SS SS SSSSSS SSSSSS	YY YY YY YY YY YY YY YY YY YY YY YY YY YY YY	YY YY YY YY YY YY YY YY YY YY YY YY YY YY YY	SSSSSSSS SSSSSSSS SS SS SS SS SSSSSS SSSSSS SS SS SSSSSS SSSSSS	
LL LL LL LL LL LL LL LL LL LL LL LLLLLLLL LLLLLLLL	IIIIII IIIIII II II II II II II II II II IIIIII IIIIII	SSSSSSSS SSSSSSSS SS SS SS SS SSSSSS SSSSSS SS SS SS SS SSSSSS SSSSSS							

```
1 0001 0 MODULE STAIOSYS (%TITLE 'Standalone I/O system'
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 Backup/Restore
34 0034 1
35 0035 1 ABSTRACT:
36 0036 1 This module contains the replacements for I/O routines in the
37 0037 1 stand-alone environment.
38 0038 1
39 0039 1 ENVIRONMENT:
40 0040 1 VAX/VMS user mode.
41 0041 1 --
42 0042 1
43 0043 1 AUTHOR: M. Jack, CREATION DATE: 27-Dec-1980
44 0044 1
45 0045 1 MODIFIED BY:
46 0046 1
47 0047 1 V03-001 ACG0313 Andrew C. Goldstein, 12-Feb-1983 17:22
48 0048 1 Add routine subtitles
49 0049 1
50 0050 1 **
```

```
: 52      0051 1 REQUIRE 'SRC$:COMMON';
: 53      1157 1 LIBRARY 'SYS$LIBRARY:LIB';
: 54      1158 1
: 55      1159 1
: 56      1160 1 FORWARD ROUTINE
: 57      1161 1     ASSIGN CHANNEL,
: 58      1162 1     LIB$GET_COMMAND,
: 59      1163 1     LIB$PUT_OUTPUT,
: 60      1164 1     SYS$QIO,
: 61      1165 1     SYS$QIOW,
: 62      1166 1     SYS$CLOSE,
: 63      1167 1     SYS$CREATE,
: 64      1168 1     SYS$OPEN,
: 65      1169 1     SYS$CONNECT,
: 66      1170 1     SYS$PUT,
: 67      1171 1     DUMMY;
: 68      1172 1
: 69      1173 1
: 70      1174 1 EXTERNAL ROUTINE
: 71      1175 1     STA_QIO;
: 72      1176 1
: 73      1177 1
: 74      1178 1 OWN
: 75      1179 1     STA_TT_CHAN;
```

```
! Execute $ASSIGN service
! LIB$GET_COMMAND
! LIB$PUT_OUTPUT
! QIO service
! QIOW service
! CLOSE service
! CREATE service
! OPEN service
! CONNECT service
! PUT service
! Dummy service

! Standalone ACP QIO processor

! Channel assigned to SYS$INPUT/OUTPUT
```

```

: 77      1180 1 %SBTTL 'ASSIGN_CHANNEL - assign channel to device'
: 78      1181 1 ROUTINE ASSIGN_CHANNEL(DEVICE,CHANNEL)=
: 79      1182 1
: 80      1183 1 ++
: 81      1184 1
: 82      1185 1 FUNCTIONAL DESCRIPTION:
: 83      1186 1     This routine executes the $ASSIGN service.
: 84      1187 1
: 85      1188 1 INPUT PARAMETERS:
: 86      1189 1     DEVICE          - Descriptor for device name
: 87      1190 1     CHANNEL        - Pointer to where channel number will be stored
: 88      1191 1
: 89      1192 1 IMPLICIT INPUTS:
: 90      1193 1     NONE
: 91      1194 1
: 92      1195 1 OUTPUT PARAMETERS:
: 93      1196 1     NONE
: 94      1197 1
: 95      1198 1 IMPLICIT OUTPUTS:
: 96      1199 1     NONE
: 97      1200 1
: 98      1201 1 ROUTINE VALUE:
: 99      1202 1     Completion status.
: 100     1203 1
: 101     1204 1 SIDE EFFECTS:
: 102     1205 1     NONE
: 103     1206 1
: 104     1207 1 --
: 105     1208 1
: 106     1209 2 BEGIN
: 107     1210 2 MAP
: 108     1211 2     DEVICE:          REF BBLOCK;      ! Pointer to descriptor for device
: 109     1212 2 LOCAL
: 110     1213 2     IN_DESC:        VECTOR[2],        ! Descriptor for input device name
: 111     1214 2     OUT_DESC:       VECTOR[2],        ! Descriptor for BUFFER
: 112     1215 2     BUFFER:         VECTOR[64,BYTE],   ! Buffer for result of $TRNLOG
: 113     1216 2     P,              ! Temporary pointer
: 114     1217 2     STATUS;         ! Status variable
: 115     1218 2
: 116     1219 2
: 117     1220 2 ! Strip elements following colon from string if required.
: 118     1221 2
: 119     1222 2 IN_DESC[0] = .DEVICE[DSC$W_LENGTH];
: 120     1223 2 IF (P = CH$FIND.CH(.DEVICE[DSC$W_LENGTH], .DEVICE[DSC$A_POINTER], %C':')) NEQ 0
: 121     1224 2     THEN IN_DESC[0] = .P - .DEVICE[DSC$A_POINTER];
: 122     1225 2 IN_DESC[1] = .DEVICE[DSC$A_POINTER];
: 123     1226 2
: 124     1227 2
: 125     1228 2 ! Execute $TRNLOG service to handle process-permanent file.
: 126     1229 2
: 127     1230 2 OUT_DESC[0] = 64;
: 128     1231 2 OUT_DESC[1] = BUFFER;
: 129     1232 2 BUFFER[0] = 0;
: 130     1233 2 $TRNLOG(LOGNAM=IN_DESC, RSLLEN=OUT_DESC, RSLBUF=OUT_DESC);
: 131     1234 2
: 132     1235 2
: 133     1236 2 ! Handle process-permanent file.
```

```
1237 2 !
1238 2 IF .BUFFER[0] EQL %0'033'
1239 2 THEN
1240 2 BEGIN
1241 2     IN_DESC[0] = .OUT_DESC[0] - 4;
1242 2     IN_DESC[1] = .OUT_DESC[1] + 4;
1243 2 END;
1244 2
1245 2
1246 2 ! Execute $ASSIGN service, returning its result as value.
1247 2 !
1248 3 $ASSIGN(DEVNAM=IN_DESC, CHAN=.CHANNEL)
1249 1 END;
```

```
.TITLE STAIOSYS Standalone I/O system
.IDENT \V04-000\
```

```
.PSECT DATA,NOEXE,2
```

```
00000 STA_TT_CHAN:
.BLK 4
```

```
.EXTRN STA_QIO, SYS$TRNLOG
.EXTRN SYS$ASSIGN
```

```
.PSECT CODE,NOWRT,2
```

```
0004 00000 ASSIGN_CHANNEL:
```

			5E	B0	AE	9E	00002	.WORD	Save R2	1181
			52	04	AC	D0	00006	MOVAB	-80(SP), SP	
		48	AE		62	3C	0000A	MOVL	DEVICE, R2	1222
04	B2		62		3A	3A	0000E	MOVZWL	(R2), IN_DESC	
					02	12	00013	LOCC	#58, (R2), @4(R2)	1223
					51	D4	00015	BNEQ	1\$	
					51	D5	00017	CLRL	R1	
					51	D5	00017	TSTL	P	
					06	13	00019	BEQL	2\$	
48	AE		51	04	A2	C3	0001B	SUBL3	4(R2), P, IN_DESC	1224
		4C	AE	04	A2	D0	00021	MOVL	4(R2), IN_DESC+4	1225
		40	AE	40	8F	9A	00026	MOVZBL	#64, IN_DESC	1230
		44	AE		6E	9E	0002B	MOVAB	BUFFER, OUT_DESC+4	1231
					6E	94	0002F	CLRB	BUFFER	1232
					7E	7C	00031	CLRQ	-(SP)	1233
					7E	D4	00033	CLRL	-(SP)	
				4C	AE	9F	00035	PUSHAB	OUT_DESC	
				50	AE	9F	00038	PUSHAB	OUT_DESC	
				5C	AE	9F	0003B	PUSHAB	IN_DESC	
		00000000G	00		06	FB	0003E	CALLS	#6, SYS\$TRNLOG	
			1B		6E	91	00045	CMPB	BUFFER, #27	1238
					0C	12	00048	BNEQ	3\$	
48	AE	40	AE		04	C3	0004A	SUBL3	#4, OUT_DESC, IN_DESC	1241
4C	AE	44	AE		04	C1	00050	ADDL3	#4, OUT_DESC+4, IN_DESC+4	1242
					7E	7C	00056	CLRQ	-(SP)	1248
				08	AC	DD	00058	PUSHL	CHANNEL	
				54	AE	9F	0005B	PUSHAB	IN_DESC	
		00000000G	00		04	FB	0005E	CALLS	#4, SYS\$ASSIGN	

STAIOSYS
V04-000

Standalone I/O system
ASSIGN_CHANNEL - assign channel to device

B 7
16-Sep-1984 01:05:14
14-Sep-1984 11:54:05

VAX-11 Bliss-32 V4.0-742
[BACKUP.SRC]STAIOSYS.B32;1

Page 5
(3)

04 00065

RET

; 1249

; Routine Size: 102 bytes, Routine Base: CODE + 0000

```
: 148 1250 1 %SBTTL 'LIB$GET_COMMAND - get input from terminal'
: 149 1251 1 GLOBAL ROUTINE LIB$GET_COMMAND(GET_STR,PROMPT_STR,OUT_LEN)=
: 150 1252 1
: 151 1253 1 !++
: 152 1254 1
: 153 1255 1 FUNCTIONAL DESCRIPTION:
: 154 1256 1 This routine is a replacement for LIB$GET_COMMAND in the
: 155 1257 1 stand-alone environment. Note that it need only handle static
: 156 1258 1 output strings.
: 157 1259 1
: 158 1260 1 INPUT PARAMETERS:
: 159 1261 1 GET_STR - Pointer to static descriptor to receive the string.
: 160 1262 1 PROMPT_STR - Pointer to descriptor for prompt string.
: 161 1263 1 OUT_LEN - (Optional) Pointer to word to receive length.
: 162 1264 1
: 163 1265 1 IMPLICIT INPUTS:
: 164 1266 1 NONE
: 165 1267 1
: 166 1268 1 OUTPUT PARAMETERS:
: 167 1269 1 NONE
: 168 1270 1
: 169 1271 1 IMPLICIT OUTPUTS:
: 170 1272 1 NONE
: 171 1273 1
: 172 1274 1 ROUTINE VALUE:
: 173 1275 1 Completion status.
: 174 1276 1
: 175 1277 1 SIDE EFFECTS:
: 176 1278 1 NONE
: 177 1279 1
: 178 1280 1 !--
: 179 1281 1
: 180 1282 2 BEGIN
: 181 1283 2 MAP
: 182 1284 2 GET_STR: REF BBLOCK, ! Pointer to static descriptor
: 183 1285 2 PROMPT_STR: REF BBLOCK; ! Pointer to descriptor
: 184 1286 2 LOCAL
: 185 1287 2 STATUS, ! Status variable
: 186 1288 2 IOSB: VECTOR[4,WORD], ! I/O status block
: 187 1289 2 BUFFER: VECTOR[132,BYTE]; ! Input buffer
: 188 1290 2 BUILTIN
: 189 1291 2 ACTUALCOUNT;
: 190 1292 2
: 191 1293 2
: 192 1294 2 ! Assign the channel if required.
: 193 1295 2
: 194 1296 2 IF .STA_IT_CHAN EQL 0
: 195 1297 2 THEN
: 196 1298 2 BEGIN
: 197 1299 2 STATUS = ASSIGN_CHANNEL($DESCRIPTOR('SYS$INPUT'), STA_IT_CHAN);
: 198 1300 2 IF NOT .STATUS THEN RETURN .STATUS;
: 199 1301 2 END;
: 200 1302 2
: 201 1303 2
: 202 1304 2 ! Execute the QIO.
: 203 1305 2
: 204 P 1306 2 STATUS = $QIOW(
```

```

: 205 P 1307 2 FUNC=IOS READPROMPT,
: 206 P 1308 2 CHAN=STA TT_CHAN,
: 207 P 1309 2 IOSB=IOSB,
: 208 P 1310 2 P1=BUFFER,
: 209 P 1311 2 P2=132,
: 210 P 1312 2 P5=.PROMPT_STR[DSC$A_POINTER],
: 211 1313 2 P6=.PROMPT_STR[DSC$W_LENGTH]);
: 212 1314 2 IF .STATUS THEN STATUS = .IOSB[0];
: 213 1315 2 IF NOT .STATUS THEN RETURN .STATUS;
: 214 1316 2
: 215 1317 2
: 216 1318 2 ! Return the length if requested.
: 217 1319 2
: 218 1320 2 IF ACTUALCOUNT() GEQU 3
: 219 1321 2 THEN
: 220 1322 2 (.OUT_LEN)<0,16> = MINU(.IOSB[1], .GET_STR[DSC$W_LENGTH]);
: 221 1323 2
: 222 1324 2
: 223 1325 2 ! Copy the string to the output buffer and return.
: 224 1326 2
: 225 1327 2 CH$COPY(.IOSB[1], BUFFER, %C' ', .GET_STR[DSC$W_LENGTH], .GET_STR[DSC$A_POINTER]);
: 226 1328 2 $$$_NORMAL
: 227 1329 1 END;
```

```

54 55 50 4E 49 24 53 59 53 00066 P.AAB: .ASCII \SYSS$INPUT\
                                0006F .BLKB 1
                                00000009 00070 P.AAA: .LONG 9
                                00000000' 00074 .ADDRESS P.AAB
                                .EXTRN SYSS$QIOW
                                .ENTRY LIB$GET_COMMAND, Save R2,R3,R4,R5,R6
56 00000000' EF 9E 00002 MOVAB STA TT_CHAN, R6
5E FF74 CE 9E 00009 MOVAB -140(SP), SP
66 D5 0000E TSTL STA TT_CHAN
0D 12 00010 BNEQ 1$
56 DD 00012 PUSHL R6
E1 AF 9F 00014 PUSHAB P.AAA
FF6C CF 02 FB 00017 CALLS #2, ASSIGN_CHANNEL
55 50 E9 0001C BLBC STATUS, 4$
51 08 AC D0 0001F 1$: MOVL PROMPT_STR, R1
7E 61 3C 00023 MOVZWL (R1), =(SP)
04 A1 DD 00026 PUSHL 4(R1)
7E 7E 7C 00029 CLRQ -(SP)
84 8F 9A 0002B MOVZBL #132, -(SP)
14 AE 9F 0002F PUSHAB BUFFER
7E 7C 00032 CLRQ -(SP)
F8 AD 9F 00034 PUSHAB IOSB
37 DD 00037 PUSHL #55
66 DD 00039 PUSHL STA TT_CHAN
7E D4 0003B CLRL -(SP)
00000000G 00 UC FB 0003D CALLS #12, SYSS$QIOW
2D 50 E9 00044 BLBC STATUS, 4$
50 F8 AD 3C 00047 MOVZWL IOSB, STATUS
26 50 E9 0004B BLBC STATUS, 4$
```

STAIOSYS
V04-000

Standalone I/O system
LIB\$GET_COMMAND - get input from terminal

E 7
16-Sep-1984 01:05:14
14-Sep-1984 11:54:05

VAX-11 Bliss-32 V4.0-742
[BACKUP.SRC]STAIOSYS.B32;1

Page 8
(4)

		03		6C	91	0004E	CMPB	(AP), #3	: 1320
		50	FA	12	1F	00051	BLSSU	3\$	: 1322
		50	04	AD	3C	00053	MOVZWL	IOSB+2, R0	: 1322
		50	04	BC	B1	00057	CMPW	@GET_STR, R0	: 1322
		50	04	04	1E	0005B	BGEQU	2\$	: 1322
		0C		BC	3C	0005D	MOVZWL	@GET_STR, R0	: 1322
		50		50	B0	00061	MOVW	R0, @OUT_LEN	: 1327
60	20	50	04	AC	D0	00065	MOVL	GET_STR, R0	: 1327
		6E	FA	AD	2C	00069	MOVCS	IOSB+2, BUFFER, #32, (R0), @4(R0)	: 1329
		50	04	B0		0006F			: 1329
				01	D0	00071	MOVL	#1, R0	: 1329
				04	00074	4\$:	RET		: 1329

; Routine Size: 117 bytes, Routine Base: CODE + 0078

```
229 1330 1 %SBTTL 'LIB$PUT_OUTPUT - write to SYSS$OUTPUT'
230 1331 1 GLOBAL ROUTINE [LIB$PUT_OUTPUT(PUT_STR)=
231 1332 1
232 1333 1 !++
233 1334 1
234 1335 1 FUNCTIONAL DESCRIPTION:
235 1336 1 This routine is a replacement for LIB$PUT_OUTPUT in the
236 1337 1 stand-alone environment.
237 1338 1
238 1339 1 INPUT PARAMETERS:
239 1340 1 PUT_STR - Pointer to descriptor for string.
240 1341 1
241 1342 1 IMPLICIT INPUTS:
242 1343 1 NONE
243 1344 1
244 1345 1 OUTPUT PARAMETERS:
245 1346 1 NONE
246 1347 1
247 1348 1 IMPLICIT OUTPUTS:
248 1349 1 NONE
249 1350 1
250 1351 1 ROUTINE VALUE:
251 1352 1 Completion status.
252 1353 1
253 1354 1 SIDE EFFECTS:
254 1355 1 NONE
255 1356 1
256 1357 1 !--
257 1358 1
258 1359 2 BEGIN
259 1360 2 MAP
260 1361 2 PUT_STR: REF BBLOCK; ! Pointer to descriptor
261 1362 2 LOCAL
262 1363 2 STATUS, ! Status variable
263 1364 2 IOSB: VECTOR[4,WORD]; ! I/O status block
264 1365 2
265 1366 2
266 1367 2 ! Assign the channel if required.
267 1368 2
268 1369 2 IF .STA_TT_CHAN EQL 0
269 1370 2 THEN
270 1371 3 BEGIN
271 1372 3 STATUS = ASSIGN_CHANNEL($DESCRIPTOR('SYSS$INPUT'), STA_TT_CHAN);
272 1373 3 IF NOT .STATUS THEN RETURN .STATUS;
273 1374 2 END;
274 1375 2
275 1376 2
276 1377 2 ! Execute the QIO.
277 1378 2
278 P 1379 2 STATUS = $QIOW(
279 P 1380 2 FUNC=IOS$ WRITELBLK,
280 P 1381 2 CHAN=.STA_TT_CHAN,
281 P 1382 2 IOSB=IOSB,
282 P 1383 2 P1=.PUT_STR[DSC$A_POINTER],
283 P 1384 2 P2=.PUT_STR[DSC$W_LENGTH],
284 1385 2 P4=%C' ');
285 1386 2 IF .STATUS THEN STATUS = .IOSB[0];
```

STAIOSYS
V04-000

Standalone I/O system
LIB\$PUT_OUTPUT - write to SYS\$OUTPUT

G 7
16-Sep-1984 01:05:14
14-Sep-1984 11:54:05

VAX-11 Bliss-32 V4.0-742
[BACKUP.SRC]STAIOSYS.B32;1

Page 10
(5)

: 286
: 287

1387 2 .STATUS
1388 1 END;

54 55 50 4E 49 24 53 59 53 000ED P.AAD: .ASCII \SYS\$INPUT\
000F6 .BLKB 2
00000009 000F8 P.AAC: .LONG 9
00000000' 000FC .ADDRESS P.AAD

				0004 00000	.ENTRY LIB\$PUT_OUTPUT, Save R2	
52	00000000'	EF	9E	00002	MOVAB STA TT_CHAN, R2	1331
5E		08	C2	00009	SUBL2 #8, SP	
		62	D5	0000C	TSTL STA TT_CHAN	1369
		0D	12	0000E	BNEQ 1\$	
		52	DD	00010	PUSHL R2	1372
		AF	9F	00012	PUSHAB P.AAC	
FEE6	CF	02	FB	00015	CALLS #2, ASSIGN_CHANNEL	
	28	50	E9	0001A	BLBC STATUS, 2\$	1373
		7E	7C	0001D	CLRQ -(SP)	1385
		20	DD	0001F	PUSHL #32	
		7E	D4	00021	CLRL -(SP)	
	51	04	AC	00023	MOVL PUT_STR, R1	
	7E		61	3C	MOVZWL (R1), -(SP)	
		04	A1	DD	PUSHL 4(R1)	
			7E	7C	CLRQ -(SP)	
		20	AE	9F	PUSHAB IOSB	
			20	DD	PUSHL #32	
			62	DD	PUSHL STA TT_CHAN	
			7E	D4	CLRL -(SP)	
00000000G	00	0C	FB	00038	CALLS #12, SYS\$QIOW	
	03	50	E9	0003F	BLBC STATUS, 2\$	1386
	50	6E	3C	00042	MOVZWL IOSB, STATUS	
			04	00045	RET	1388

; Routine Size: 70 bytes, Routine Base: CODE + 0100

```
289 1389 1 %SBTTL 'SYSSQIO - intercept QIO service call'
290 1390 1 GLOBAL ROUTINE SYSSQIO(EFN,CHAN,FUNC,IOSB,ASTADR,ASTPRM,P1,P2,P3,P4,P5,P6)=
291 1391 1
292 1392 1 !++
293 1393 1
294 1394 1 FUNCTIONAL DESCRIPTION:
295 1395 1 This routine is a replacement for the QIO service in the
296 1396 1 stand-alone environment.
297 1397 1
298 1398 1 INPUT PARAMETERS:
299 1399 1 As for SYSSQIO.
300 1400 1
301 1401 1 IMPLICIT INPUTS:
302 1402 1 NONE
303 1403 1
304 1404 1 OUTPUT PARAMETERS:
305 1405 1 NONE
306 1406 1
307 1407 1 IMPLICIT OUTPUTS:
308 1408 1 NONE
309 1409 1
310 1410 1 ROUTINE VALUE:
311 1411 1 Completion status.
312 1412 1
313 1413 1 SIDE EFFECTS:
314 1414 1 NONE
315 1415 1
316 1416 1 !--
317 1417 1
318 1418 2 BEGIN
319 1419 2 MAP
320 1420 2 FUNC: BBLOCK; ! I/O function code
321 1421 2 BIND ROUTINE
322 1422 2 QIO_VECTOR= %X'800001C8'; ! Vector location for SYSSQIO
323 1423 2 BUILTIN
324 1424 2 CALLG,
325 1425 2 AP;
326 1426 2
327 1427 2
328 1428 2 IF
329 1429 2 .FUNC[IOSV_FCODE] LEQU IOS_LOGICAL OR
330 1430 2 .FUNC[IOSV_FCODE] EQL IOS_READPROMPT
331 1431 2 THEN
332 1432 2 BEGIN
333 1433 2
334 1434 2 Pass physical and logical I/O functions, and READPROMPT function,
335 1435 2 directly to the driver.
336 1436 2
337 1437 2 CALLG(.AP, QIO_VECTOR)
338 1438 2 END
339 1439 2 ELSE
340 1440 2 BEGIN
341 1441 2
342 1442 2 Read/write virtual and ACP I/O functions come here.
343 1443 2
344 1444 2 CALLG(.AP, STA_QIO)
345 1445 2 END
```

STAIOSYS
V04-000

Standalone I/O system
SYSSQIO - intercept QIO service call

1 7
16-Sep-1984 01:05:14
14-Sep-1984 11:54:05

VAX-11 Bliss-32 V4.0-742
[BACKUP.SRC]STAIOSYS.B32;1

Page 12
(6)

; 346

1446 1 END;

QIO_VECTOR= -2147483192

2F	0C	AC	06	0000 00000	.ENTRY	SYSSQIO, Save nothing	:	1390
				00 ED 00002	CMPZV	#0, #6, FUNC, #47	:	1429
37	0C	AC	06	08 1B 00008	BLEQU	1\$	:	
				00 ED 0000A	CMPZV	#0, #6, FUNC, #55	:	1430
				08 12 00010	BNEQ	2\$	:	
		800001C8	9F	6C FA 00012 1\$:	CALLG	(AP), @#X800001C8	:	1437
				04 00019	RET		:	1432
		00000000G	00	6C FA 0001A 2\$:	CALLG	(AP), STA_QIO	:	1444
				04 00021	RET		:	1446

; Routine Size: 34 bytes, Routine Base: CODE + 0146

```

: 348 1447 1 XSBTTL 'SYSSQIOW - intercept QIOW service call'
: 349 1448 1 GLOBAL ROUTINE SYSSQIOW(EFN,CHAN,FUNC,IOSB,ASTADR,ASTPRM,P1,P2,P3,P4,P5,P6)=
: 350 1449 1
: 351 1450 1 ++
: 352 1451 1
: 353 1452 1 FUNCTIONAL DESCRIPTION:
: 354 1453 1 This routine is a replacement for the QIOW service in the
: 355 1454 1 stand-alone environment.
: 356 1455 1
: 357 1456 1 INPUT PARAMETERS:
: 358 1457 1 As for SYSSQIOW.
: 359 1458 1
: 360 1459 1 IMPLICIT INPUTS:
: 361 1460 1 NONE
: 362 1461 1
: 363 1462 1 OUTPUT PARAMETERS:
: 364 1463 1 NONE
: 365 1464 1
: 366 1465 1 IMPLICIT OUTPUTS:
: 367 1466 1 NONE
: 368 1467 1
: 369 1468 1 ROUTINE VALUE:
: 370 1469 1 Completion status.
: 371 1470 1
: 372 1471 1 SIDE EFFECTS:
: 373 1472 1 NONE
: 374 1473 1
: 375 1474 1 --
: 376 1475 1
: 377 1476 2 BEGIN
: 378 1477 2 EXTERNAL ROUTINE
: 379 1478 2 SYSSWAITFR: ADDRESSING_MODE(GENERAL); ! Wait for event flag
: 380 1479 2 REGISTER
: 381 1480 2 R0=0;
: 382 1481 2 BUILTIN
: 383 1482 2 CALLG,
: 384 1483 2 AP;
: 385 1484 2
: 386 1485 2
: 387 1486 2 R0 = CALLG(.AP, SYSSQIO); ! Execute SQIO part
: 388 1487 2 IF NOT .R0 THEN RETURN .R0; ! If failed, return status
: 389 1488 2 CALLG(.AP, SYSSWAITFR) ! Execute $WAITFR part
: 390 1489 1 END;
```

```

                                .EXTRN SYSSWAITFR
                                .ENTRY SYSSQIOW, Save nothing
                                CALLG (AP), SYSSQIO
                                BLBC R0, 1$
                                CALLG (AP), SYSSWAITFR
                                RET
                                0000 0000
                                D8 AF 6C FA 0002
                                00000000G 07 50 E9 0006
                                00 6C FA 0009
                                04 00010 1$:
```

```

: 1448
: 1486
: 1487
: 1488
: 1489
```

; Routine Size: 17 bytes, Routine Base: CODE + 0168

```
392 1490 1 XSBTTL 'SYSS$CLOSE - RMS $CLOSE routine'
393 1491 1 GLOBAL ROUTINE SYSS$CLOSE(FAB)=
394 1492 1
395 1493 1 !++
396 1494 1
397 1495 1 FUNCTIONAL DESCRIPTION:
398 1496 1 This routine is a replacement for the CLOSE service in the
399 1497 1 stand-alone environment.
400 1498 1
401 1499 1 INPUT PARAMETERS:
402 1500 1 FAB - Pointer to the FAB.
403 1501 1
404 1502 1 IMPLICIT INPUTS:
405 1503 1 NONE
406 1504 1
407 1505 1 OUTPUT PARAMETERS:
408 1506 1 NONE
409 1507 1
410 1508 1 IMPLICIT OUTPUTS:
411 1509 1 NONE
412 1510 1
413 1511 1 ROUTINE VALUE:
414 1512 1 Completion status.
415 1513 1
416 1514 1 SIDE EFFECTS:
417 1515 1 NONE
418 1516 1
419 1517 1 !--
420 1518 1
421 1519 2 BEGIN
422 1520 2 MAP
423 1521 2 FAB: REF BBLOCK; ! Pointer to FAB
424 1522 2
425 1523 2
426 1524 2 ! This service deassigns the channel that was assigned by $CREATE or $OPEN.
427 1525 2
428 1526 2 $DASSGN(CHAN=FAB[FAB$L_CTX]);
429 1527 2 FAB[FAB$L_IFI] = 0;
430 1528 2 FAB[FAB$L_CTX] = 0;
431 1529 2 FAB[FAB$L_STS] = RMSS_NORMAL;
432 1530 2 FAB[FAB$L_STV] = 0;
433 1531 2 RMSS_NORMAL
434 1532 1 END;
```

				0004 00000	.EXTRN SYSS\$DASSGN	
	52	04	AC	D0 00002	.ENTRY SYSS\$CLOSE, Save R2	: 1491
		18	A2	DD 00006	MOVL FAB, R2	: 1526
00000000G	00		A1	FB 00009	PUSHL 24(R2)	
		02	A2	B4 00010	CALLS #1, SYSS\$DASSGN	
		18	A2	D4 00013	CLRW 2(R2)	: 1527
					CLRL 24(R2)	: 1528
08	A2	00010001	8F	D0 00016	MOVL #65537, 8(R2)	: 1529
		0C	A2	D4 0001E	CLRL 12(R2)	: 1530
	50	00010001	8F	D0 00021	MOVL #65537, R0	: 1532

STAIOSYS
V04-000

Standalone I/O system
SYSCLOSE - RMS SCLOSE routine

L 7
16-Sep-1984 01:05:14
14-Sep-1984 11:54:05

VAX-11 Bliss-32 V4.0-742
[BACKUP.SRC]STAIOSYS.B32;1

Page 15
(8)

04 00028

RET

;

; Routine Size: 41 bytes, Routine Base: CODE + 0179

```

436 1533 1 %SBTTL 'SYSS$CREATE - RMS $CREATE routine'
437 1534 1 GLOBAL ROUTINE SYSS$CREATE(FAB)=
438 1535 1
439 1536 1 ++
440 1537 1
441 1538 1 FUNCTIONAL DESCRIPTION:
442 1539 1 This routine is a replacement for the CREATE service in the
443 1540 1 stand-alone environment. It is limited to operation on terminals,
444 1541 1 line printers, and magnetic tape.
445 1542 1
446 1543 1 INPUT PARAMETERS:
447 1544 1 FAB - Pointer to the FAB.
448 1545 1
449 1546 1 IMPLICIT INPUTS:
450 1547 1 NONE
451 1548 1
452 1549 1 OUTPUT PARAMETERS:
453 1550 1 NONE
454 1551 1
455 1552 1 IMPLICIT OUTPUTS:
456 1553 1 NONE
457 1554 1
458 1555 1 ROUTINE VALUE:
459 1556 1 Completion status.
460 1557 1
461 1558 1 SIDE EFFECTS:
462 1559 1 NONE
463 1560 1
464 1561 1 --
465 1562 1
466 1563 2 BEGIN
467 1564 2 MAP
468 1565 2 FAB: REF BBLOCK; ! Pointer to FAB
469 1566 2 LOCAL
470 1567 2 NAM: REF BBLOCK, ! Pointer to NAM block
471 1568 2 DESC: VECTOR[2], ! Descriptor for expanded string
472 1569 2 STATUS; ! Status variable
473 1570 2
474 1571 2
475 1572 2 ! Parse the file specification if necessary.
476 1573 2
477 1574 2 IF NOT $PARSE(FAB=.FAB) THEN RETURN .FAB[FAB$$_STS];
478 1575 2
479 1576 2
480 1577 2 ! Check the device type.
481 1578 2
482 1579 2 IF
483 1580 2 NOT .BBLOCK[FAB[FAB$$_DEV], DEV$V_CCL] AND
484 1581 2 NOT .BBLOCK[FAB[FAB$$_DEV], DEV$V_SQD] AND
485 1582 2 .FAB[FAB$$_DEV] NEQ 0 ! Process-permanent file hack
486 1583 2 THEN
487 1584 3 BEGIN
488 1585 3 FAB[FAB$$_STS] = RMSS$_DEV;
489 1586 3 FAB[FAB$$_STV] = 0;
490 1587 3 RETURN RMSS$_DEV
491 1588 2 END;
492 1589 2
```

```

493 1590 2
494 1591 2 ! Create the resultant string.
495 1592 2 !
496 1593 2 NAM = .FAB[FAB$L_NAM];
497 1594 2 NAM[NAM$B_RSL] = .NAM[NAM$B_ESL];
498 1595 2 CH$MOVE(.NAM[NAM$B_ESL], .NAM[NAM$L_ESA], .NAM[NAM$L_RSA]);
499 1596 2
500 1597 2
501 1598 2 ! Assign the channel.
502 1599 2 !
503 1600 2 DESC[0] = .NAM[NAM$B_ESL];
504 1601 2 DESC[1] = .NAM[NAM$L_ESA];
505 1602 2 STATUS = ASSIGN_CHANNEL(DESC, FAB[FAB$L_CTX]);
506 1603 2 IF NOT .STATUS
507 1604 2 THEN
508 1605 3 BEGIN
509 1606 3 FAB[FAB$L_STS] = .STATUS;
510 1607 3 FAB[FAB$L_STV] = 0;
511 1608 3 RETURN .STATUS;
512 1609 2 END;
513 1610 2
514 1611 2
515 1612 2 ! Indicate success.
516 1613 2 !
517 1614 2 FAB[FAB$L_STS] = RMSS$NORMAL;
518 1615 2 FAB[FAB$L_STV] = .FAB[FAB$L_CTX];
519 1616 2 RMSS$NORMAL
520 1617 1 END;
```

```

                                00FC 00000
                                08 C2 00002
                                5E 04 AC D0 00005
                                57 57 DD 00009
                                00 01 FB 0000B
                                05 50 E8 00012
                                50 08 A7 D0 00015
                                1D 40 A7 01 E0 0001A 1$:
                                18 40 A7 05 E0 0001F
                                40 A7 D5 00024
                                13 13 00027
                                08 A7 000184C4 8F D0 00029
                                0C A7 D4 00031
                                50 000184C4 8F D0 00034
                                04 0003B
                                56 28 A7 D0 0003C 2$:
                                03 A6 0B A6 90 00040
                                50 0B A6 9A 00045
                                04 B6 0C 50 28 00049
                                6E 0B A6 9A 0004F
                                04 AE 0C A6 D0 00053
                                18 A7 9F 00058
                                04 AE 9F 0005B
```

.EXTRN SYSS\$PARSE

```

.ENTRY SYSS$CREATE, Save R2,R3,R4,R5,R6,R7
SUBL2 #8, SP
MOVL FAB, R7
PUSHL R7
CALLS #1, SYSS$PARSE
BLBS R0, 1$
MOVL 8(R7), R0
RET
BBS #1, 64(R7), 2$
BBS #5, 64(R7), 2$
TSTL 64(R7)
BEQL 2$
MOVL #99524, 8(R7)
CLRL 12(R7)
MOVL #99524, R0
RET
MOVL 40(R7), NAM
MOVB 11(NAM), 3(NAM)
MOVZBL 11(NAM), R0
MOVC3 R0, @12(NAM), @4(NAM)
MOVZBL 11(NAM), DESC
MOVL 12(NAM), DESC+4
PUSHAB 24(R7)
PUSHAB DESC
```

```

: 1534
: 1574
: 1580
: 1581
: 1582
: 1585
: 1586
: 1587
: 1593
: 1594
: 1595
: 1600
: 1601
: 1602
:
```

STAIOSYS
V04-000

Standalone I/O system
SYS\$CREATE - RMS \$CREATE routine

B 8
16-Sep-1984 01:05:14
14-Sep-1984 11:54:05

VAX-11 Bliss-32 V4.0-742
[BACKUP.SRC]STAIOSYS.B32;1

Page 18
(9)

FDFB	CF	02	FB	0005E	CALLS	#2, ASSIGN_CHANNEL
	08	50	E8	00063	BLBS	STATUS, 3\$
08	A7	50	D0	00066	MOVL	STATUS, 8(R7)
		OC	A7	D4 0006A	CLRL	12(R7)
			04	0006D	RET	
08	A7	00010001	8F	D0 0006E	3\$:	MOVL #65537, 8(R7)
OC	A7	18	A7	D0 00076	MOVL	24(R7), 12(R7)
	50	00C10001	8F	D0 0007B	MOVL	#65537, R0
			04	00082	RET	

: 1603
: 1606
: 1607
: 1608
: 1614
: 1615
: 1617
:

; Routine Size: 131 bytes, Routine Base: CODE + 01A2

```

522 1618 1 %SBTTL 'SYSS$OPEN - RMS $OPEN routine'
523 1619 1 GLOBAL ROUTINE SYSS$OPEN(FAB)=
524 1620 1
525 1621 1 ++
526 1622 1
527 1623 1 FUNCTIONAL DESCRIPTION:
528 1624 1 This routine is a replacement for the OPEN service in the
529 1625 1 stand-alone environment. It is limited to operation on terminals,
530 1626 1 line printers, and magnetic tape.
531 1627 1
532 1628 1 INPUT PARAMETERS:
533 1629 1 FAB - Pointer to the FAB.
534 1630 1
535 1631 1 IMPLICIT INPUTS:
536 1632 1 NONE
537 1633 1
538 1634 1 OUTPUT PARAMETERS:
539 1635 1 NONE
540 1636 1
541 1637 1 IMPLICIT OUTPUTS:
542 1638 1 NONE
543 1639 1
544 1640 1 ROUTINE VALUE:
545 1641 1 Completion status.
546 1642 1
547 1643 1 SIDE EFFECTS:
548 1644 1 NONE
549 1645 1
550 1646 1 --
551 1647 1
552 1648 2 BEGIN
553 1649 2 MAP
554 1650 2 FAB: REF BBLOCK; ! Pointer to FAB
555 1651 2 LOCAL
556 1652 2 NAM: REF BBLOCK, ! Pointer to NAM block
557 1653 2 DESC: VECTOR[2], ! Descriptor for expanded string
558 1654 2 STATUS; ! Status variable
559 1655 2
560 1656 2
561 1657 2 ! Parse the file specification if necessary.
562 1658 2
563 1659 2 IF NOT $PARSE(FAB=.FAB) THEN RETURN .FAB[FAB$L_STS];
564 1660 2
565 1661 2
566 1662 2 ! Check the device type.
567 1663 2
568 1664 2 IF
569 1665 2 NOT .BBLOCK[FAB[FAB$L_DEV], DEV$V_CCL] AND
570 1666 2 NOT .BBLOCK[FAB[FAB$L_DEV], DEV$V_SQD] AND
571 1667 2 .FAB[FAB$L_DEV] NEQ 0 ! Process-permanent file hack
572 1668 2 THEN
573 1669 3 BEGIN
574 1670 3 FAB[FAB$L_STS] = RMSS_DEV;
575 1671 3 FAB[FAB$L_STV] = 0;
576 1672 3 RETURN RMSS_DEV
577 1673 2 END;
578 1674 2
```

```

579 1675 2
580 1676 2 ! Create the resultant string.
581 1677 2
582 1678 2 NAM = .FAB[FAB$L_NAM];
583 1679 2 NAM[NAM$B_RSL] = .NAM[NAM$B_ESL];
584 1680 2 CH$MOVE(.NAM[NAM$B_ESL], .NAM[NAM$L_ESA], .NAM[NAM$L_RSA]);
585 1681 2
586 1682 2
587 1683 2 ! Assign the channel.
588 1684 2
589 1685 2 DESC[0] = .NAM[NAM$B_ESL];
590 1686 2 DESC[1] = .NAM[NAM$L_ESA];
591 1687 2 STATUS = ASSIGN_CHANNEL(DESC, FAB[FAB$L_CTX]);
592 1688 2 IF NOT .STATUS
593 1689 2 THEN
594 1690 2 BEGIN
595 1691 2 FAB[FAB$L_STS] = .STATUS;
596 1692 2 FAB[FAB$L_STV] = 0;
597 1693 2 RETURN .STATUS;
598 1694 2 END;
599 1695 2
600 1696 2
601 1697 2 ! Indicate success.
602 1698 2
603 1699 2 FAB[FAB$L_STS] = RMSS$ NORMAL;
604 1700 2 FAB[FAB$L_STV] = .FAB[FAB$L_CTX];
605 1701 2 RMSS$ NORMAL
606 1702 1 END;
```

				00FC 00000	.ENTRY	SYSSOPEN, Save R2,R3,R4,R5,R6,R7	1619
		5E		08 C2 00002	SUBL2	#8, SP	
		57	04	AC D0 00005	MOVL	FAB, R7	1659
				57 DD 00009	PUSHL	R7	
	00000000G	00		01 FB 0000B	CALLS	#1, SYSSPARSE	
		05		50 E8 00012	BLBS	R0, 1\$	
		50	08	A7 D0 00015	MOVL	8(R7), R0	
				04 00019	RET		
1D	40	A7		01 E0 0001A	BBS	#1, 64(R7), 2\$	1665
18	40	A7		05 E0 0001F	BBS	#5, 64(R7), 2\$	1666
			40	A7 D5 00024	TSTL	64(R7)	1667
				13 13 00027	BEQL	2\$	
	08	A7	000184C4	8F D0 00029	MOVL	#99524, 8(R7)	1670
			0C	A7 D4 00031	CLRL	12(R7)	1671
		50	000184C4	8F D0 00034	MOVL	#99524, R0	1672
				04 0003B	RET		
		56	28	A7 D0 0003C	MOVL	40(R7), NAM	1678
	03	A6	0B	A6 90 00040	MOVB	11(NAM), 3(NAM)	1679
		50	0B	A6 9A 00045	MOVZBL	11(NAM), R0	1680
04	B6	0C	B6	50 28 00049	MOVCL	R0, @12(NAM), @4(NAM)	
		6E	0B	A6 9A 0004F	MOVZBL	11(NAM), DESC	1685
		04	AE	0C A6 D0 00053	MOVL	12(NAM), DESC+4	1686
			18	A7 9F 00058	PUSHAB	24(R7)	1687
			04	AE 9F 0005B	PUSHAB	DESC	

STAIOSYS
V04-000

Standalone I/O system
SYSSOPEN - RMS \$OPEN routine

E 8
16-Sep-1984 01:05:14
14-Sep-1984 11:54:05

VAX-11 Bliss-32 V4.0-742
[BACKUP.SRC]STAIOSYS.B32;1

Page 21
(10)

FD78	CF		02	FB	0005E	CALLS	#2, ASSIGN_CHANNEL
	08		50	E8	00063	BLBS	STATUS, 3\$
08	A7		50	D0	00066	MOVL	STATUS, 8(R7)
		0C	A7	D4	0006A	CLRL	12(R7)
				04	0006D	RET	
08	A7	00010001	8F	D0	0006E	MOVL	#65537, 8(R7)
0C	A7	18	A7	D0	00076	MOVL	24(R7), 12(R7)
	50	00010001	8F	D0	0007B	MOVL	#65537, R0
				04	00082	RET	

: 1688
: 1691
: 1692
: 1693
: 1699
: 1700
: 1702
:

; Routine Size: 131 bytes, Routine Base: CODE + 0225

```
: 608 1703 1 %SBTTL 'SYSSCONNECT - RMS $CONNECT routine'
: 609 1704 1 GLOBAL ROUTINE SYSSCONNECT(RAB)=
: 610 1705 1
: 611 1706 1 !++
: 612 1707 1
: 613 1708 1 FUNCTIONAL DESCRIPTION:
: 614 1709 1 This routine is a replacement for the CONNECT service in the
: 615 1710 1 stand-alone environment.
: 616 1711 1
: 617 1712 1 INPUT PARAMETERS:
: 618 1713 1 RAB - Pointer to the RAB.
: 619 1714 1
: 620 1715 1 IMPLICIT INPUTS:
: 621 1716 1 NONE
: 622 1717 1
: 623 1718 1 OUTPUT PARAMETERS:
: 624 1719 1 NONE
: 625 1720 1
: 626 1721 1 IMPLICIT OUTPUTS:
: 627 1722 1 NONE
: 628 1723 1
: 629 1724 1 ROUTINE VALUE:
: 630 1725 1 Completion status.
: 631 1726 1
: 632 1727 1 SIDE EFFECTS:
: 633 1728 1 NONE
: 634 1729 1
: 635 1730 1 !--
: 636 1731 1
: 637 1732 2 BEGIN
: 638 1733 2 MAP
: 639 1734 2 RAB: REF BBLOCK; ! Pointer to RAB
: 640 1735 2
: 641 1736 2
: 642 1737 2 ! This service is a no-operation.
: 643 1738 2
: 644 1739 2 RAB[RAB$SL_STS] = RMSS_NORMAL;
: 645 1740 2 RAB[RAB$SL_STV] = 0;
: 646 1741 2 RMSS_NORMAL
: 647 1742 1 END;
```

```
0000 00000
08 50 04 AC D0 00002
AO 00010001 8F D0 00006
OC AO D4 0000E
50 00010001 8F D0 00011
04 00018
```

```
.ENTRY SYSSCONNECT, Save nothing
MOVL RAB, R0
MOVL #65537, 8(R0)
CLRL 12(R0)
MOVL #65537, R0
RET
```

```
: 1704
: 1739
: 1740
: 1742
:
```

; Routine Size: 25 bytes, Routine Base: CODE + 02A8

```

649 1743 1 %SBTTL 'SYSSPUT - RMS $PUT routine'
650 1744 1 GLOBAL ROUTINE SYSSPUT(RAB)=
651 1745 1
652 1746 1 ++
653 1747 1
654 1748 1 FUNCTIONAL DESCRIPTION:
655 1749 1 This routine is a replacement for the PUT service in the
656 1750 1 stand-alone environment. This service is provided to allow
657 1751 1 /LIST to operate. It is restricted to terminals and line printers.
658 1752 1
659 1753 1 INPUT PARAMETERS:
660 1754 1 RAB - Pointer to the RAB.
661 1755 1
662 1756 1 IMPLICIT INPUTS:
663 1757 1 NONE
664 1758 1
665 1759 1 OUTPUT PARAMETERS:
666 1760 1 NONE
667 1761 1
668 1762 1 IMPLICIT OUTPUTS:
669 1763 1 NONE
670 1764 1
671 1765 1 ROUTINE VALUE:
672 1766 1 Completion status.
673 1767 1
674 1768 1 SIDE EFFECTS:
675 1769 1 NONE
676 1770 1
677 1771 1 --
678 1772 1
679 1773 2 BEGIN
680 1774 2 MAP
681 1775 2 RAB: REF BBLOCK; ! Pointer to RAB
682 1776 2 LOCAL
683 1777 2 FAB: REF BBLOCK, ! Pointer to FAB
684 1778 2 STATUS, ! Status variable
685 1779 2 IOSB: VECTOR[4,WORD]; ! I/O status block
686 1780 2
687 1781 2
688 1782 2 ! Point to FAB.
689 1783 2
690 1784 2 FAB = .RAB[RAB$L_FAB];
691 1785 2
692 1786 2
693 1787 2 ! Make sure a channel has been assigned.
694 1788 2
695 1789 2 IF .FAB[FAB$L_CTX] EQL 0
696 1790 2 THEN
697 1791 2 BEGIN
698 1792 2 RAB[RAB$L_STS] = RMSS_DEV;
699 1793 2 RAB[RAB$L_STV] = 0;
700 1794 2 RETURN RMSS_DEV
701 1795 2 END;
702 1796 2
703 1797 2
704 1798 2 ! Execute the I/O.
705 1799 2
```

```
: 706
: 707
: 708
: 709
: 710
: 711
: 712
: 713
: 714
: 715
: 716
: 717

P 1800 2 STATUS = $QIOW(
P 1801 2 FUNC=IOS$ WRITELBLK,
P 1802 2 CHAN=.FAB[FAB$L_CTX],
P 1803 2 IOSB=IOSB,
P 1804 2 P1=.RAB[RAB$L_RBF],
P 1805 2 P2=.RAB[RAB$W_RSZ],
1806 2 P4=%C' ');
1807 2 IF STATUS THEN STATUS = .IOSB[0];
1808 2 RAB[RAB$L_STS] = STATUS;
1809 2 RAB[RAB$L_STV] = 0;
1810 2 .STATUS
1811 1 END;
```

			0004 00000	.ENTRY	SY\$SPUT, Save R2	
	5E		08 C2 00002	SUBL2	#8, SP	1744
	52	04	AC D0 00005	MOVL	RAB, R2	1784
	50	3C	A2 D0 00009	MOVL	60(R2), FAB	
		18	A0 D5 0000D	TSTL	24(FAB)	1789
			13 12 00010	BNEQ	1\$	
08	A2	000184C4	8F D0 00012	MOVL	#99524, 8(R2)	1792
		0C	A2 D4 0001A	CLRL	12(R2)	1793
	50	000184C4	8F D0 0001D	MOVL	#99524, R0	1794
			04 00024	RET		
			7E 7C 00025 1\$:	CLRQ	-(SP)	1806
			20 DD 00027	PUSHL	#32	
	7E	22	7E D4 00029	CLRL	-(SP)	
		28	A2 3C 0002B	MOVZWL	34(R2), -(SP)	
			A2 DD 0002F	PUSHL	40(R2)	
			7E 7C 00032	CLRQ	-(SP)	
		20	AE 9F 00034	PUSHAB	IOSB	
			20 DD 00037	PUSHL	#32	
		18	A0 DD 00039	PUSHL	24(FAB)	
			7E D4 0003C	CLRL	-(SP)	
00000000G	00		0C FB 0003E	CALLS	#12, SY\$QIOW	
	03		50 E9 00045	BLBC	STATUS, 2\$	1807
	50		6E 3C 00048	MOVZWL	IOSB, STATUS	
08	A2		50 D0 0004B 2\$:	MOVL	STATUS, 8(R2)	1808
		0C	A2 D4 0004F	CLRL	12(R2)	1809
			04 00052	RET		1811

; Routine Size: 83 bytes, Routine Base: CODE + 02C1

```
: 719 1812 1 %SBTTL 'DUMMY - various dummy RMS routines'
: 720 1813 1 ROUTINE DUMMY(RAB)=
: 721 1814 1
: 722 1815 1 ++
: 723 1816 1
: 724 1817 1 FUNCTIONAL DESCRIPTION:
: 725 1818 1 This routine serves as a stub for services that are referenced but are
: 726 1819 1 never called in the stand-alone environment, namely SYSS$FIND, SYSS$GET,
: 727 1820 1 and SYSS$REWIND. These services are used to handle save set files in
: 728 1821 1 the on-line version.
: 729 1822 1
: 730 1823 1 INPUT PARAMETERS:
: 731 1824 1 RAB - Pointer to the RAB.
: 732 1825 1
: 733 1826 1 IMPLICIT INPUTS:
: 734 1827 1 NONE
: 735 1828 1
: 736 1829 1 OUTPUT PARAMETERS:
: 737 1830 1 NONE
: 738 1831 1
: 739 1832 1 IMPLICIT OUTPUTS:
: 740 1833 1 NONE
: 741 1834 1
: 742 1835 1 ROUTINE VALUE:
: 743 1836 1 Completion status (RMSS$_IOP, invalid operation).
: 744 1837 1
: 745 1838 1 SIDE EFFECTS:
: 746 1839 1 NONE
: 747 1840 1
: 748 1841 1 --
: 749 1842 1
: 750 1843 2 BEGIN
: 751 1844 2 MAP
: 752 1845 2 RAB: REF BBLOCK; ! Pointer to RAB
: 753 1846 2
: 754 1847 2
: 755 1848 2 RAB[RAB$_STS] = RMSS$_IOP;
: 756 1849 2 RAB[RAB$_STV] = 0;
: 757 1850 2 RMSS$_IOP
: 758 1851 1 END;
```

```
0000 00000 DUMMY: .WORD Save nothing
08 50 04 AC D0 00002 MOVL RAB, R0
00 00018574 8F D0 00006 MOVL #99700, 8(R0)
50 00018574 0C A0 D4 0000E CLRL 12(R0)
8F D0 00011 MOVL #99700, R0
04 00018 RET
```

; Routine Size: 25 bytes, Routine Base: CODE + 0314

```
: 1813
: 1848
: 1849
: 1851
:
```

STAIOSYS
V04-000

Standalone I/O system
DUMMY - various dummy RMS routines

J 8
16-Sep-1984 01:05:14
14-Sep-1984 11:54:05

VAX-11 Bliss-32 V4.0-742
[BACKUP.SRC]STAIOSYS.B32;1

Page 26
(14)

```
: 760      1852 1 GLOBAL BIND ROUTINE
: 761      1853 1      SYSS$FIND      = DUMMY,
: 762      1854 1      SYSS$GET      = DUMMY,
: 763      1855 1      SYSS$REWIND   = DUMMY;
: 764      1856 1
: 765      1857 1
: 766      1858 1 END
: 767      1859 0 ELUDOM
```

```
SYSS$FIND==      DUMMY
SYSS$GET==        DUMMY
SYSS$REWIND==     DUMMY
```

PSECT SUMMARY

Name	Bytes	Attributes
DATA	4	NOVEC, WRT, RD, NOEXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
CODE	813	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	36	0	1000	00:02.2

COMMAND QUALIFIERS

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

```
: Size:      776 code + 41 data bytes
: Run Time:   00:23.3
: Elapsed Time: 01:11.8
: Lines/CPU Min: 4789
: Lexemes/CPU-Min: 33866
: Memory Used: 185 pages
: Compilation Complete
```

