

NNN		NNN	CCCCCCCCCCCC	PPPPPPPPPPPP	
NNN		NNN	CCCCCCCCCCCC	PPPPPPPPPPPP	
NNN		NNN	CCCCCCCCCCCC	PPPPPPPPPPPP	
NNN		NNN	CCC	PPP	PPP
NNN		NNN	CCC	PPP	PPP
NNN		NNN	CCC	PPP	PPP
NNNNNN		NNN	CCC	PPP	PPP
NNNNNN		NNN	CCC	PPP	PPP
NNNNNN		NNN	CCC	PPP	PPP
NNN	NNN	NNN	CCC	PPPPPPPPPPPP	
NNN	NNN	NNN	CCC	PPPPPPPPPPPP	
NNN	NNN	NNN	CCC	PPPPPPPPPPPP	
NNN	NNNNNN	NNN	CCC	PPP	
NNN	NNNNNN	NNN	CCC	PPP	
NNN	NNNNNN	NNN	CCC	PPP	
NNN	NNN	NNN	CCC	PPP	
NNN	NNN	NNN	CCC	PPP	
NNN	NNN	NNN	CCC	PPP	
NNN	NNN	NNN	CCCCCCCCCCCC	PPP	
NNN	NNN	NNN	CCCCCCCCCCCC	PPP	
NNN	NNN	NNN	CCCCCCCCCCCC	PPP	

```

NN      NN      CCCCCCCC  PPPPPPPP  SSSSSSSS  TTTTTTTTTT  AAAAAA  SSSSSSSS  HH      HH  LL
NN      NN      CCCCCCCC  PPPPPPPP  SSSSSSSS  TTTTTTTTTT  AAAAAA  SSSSSSSS  HH      HH  LL
NN      NN      CC        PP        PP  SS        TT        AA      AA  SS        HH      HH  LL
NN      NN      CC        PP        PP  SS        TT        AA      AA  SS        HH      HH  LL
NNNN    NN      CC        PP        PP  SS        TT        AA      AA  SS        HH      HH  LL
NNNN    NN      CC        PP        PP  SS        TT        AA      AA  SS        HH      HH  LL
NN      NN      CC        PPPPPPPP  SSSSSS    TT        AA      AA  SSSSSS  HHHHHHHHHH  LL
NN      NN      CC        PPPPPPPP  SSSSSS    TT        AA      AA  SSSSSS  HHHHHHHHHH  LL
NN      NNNN    CC        PP        SS        TT        AAAAAAAAAA  SS        HH      HH  LL
NN      NNNN    CC        PP        SS        TT        AAAAAAAAAA  SS        HH      HH  LL
NN      NN      CC        PP        SS        TT        AA      AA  SS        HH      HH  LL
NN      NN      CC        PP        SS        TT        AA      AA  SS        HH      HH  LL
NN      NN      CCCCCCCC  PP        SSSSSSSS  TT        AA      AA  SSSSSSSS  HH      HH  LLLLLLLLLL
NN      NN      CCCCCCCC  PP        SSSSSSSS  TT        AA      AA  SSSSSSSS  HH      HH  LLLLLLLLLL
                                     ....
                                     ....
                                     ....
                                     ....

LL      IIIIII  SSSSSSSS
LL      IIIIII  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
LLLLLLLLLLLL  IIIIII  SSSSSSSS
LLLLLLLLLLLL  IIIIII  SSSSSSSS
```



```
0001 0 %TITLE 'Show/List Parse States and Data'
0002 0 MODULE NCPSTASHL (IDENT = 'V04-000', LIST(NOOBJECT)) =
0003 1 BEGIN
0004 1
0005 1
0006 1 *****
0007 1 *
0008 1 *  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0009 1 *  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0010 1 *  ALL RIGHTS RESERVED.
0011 1 *
0012 1 *  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0013 1 *  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0014 1 *  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0015 1 *  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0016 1 *  OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0017 1 *  TRANSFERRED.
0018 1 *
0019 1 *  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0020 1 *  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0021 1 *  CORPORATION.
0022 1 *
0023 1 *  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0024 1 *  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0025 1 *
0026 1 *
0027 1 *****
0028 1
0029 1
0030 1 ++
0031 1 FACILITY:      Network Control Program (NCP)
0032 1
0033 1 ABSTRACT:
0034 1
0035 1      States and data for the parsing of NCP show and list commands
0036 1
0037 1 ENVIRONMENT:  VAX/VMS Operating System
0038 1
0039 1 AUTHOR:      Darrell Duffy , CREATION DATE: 14-September-79
0040 1
0041 1 MODIFIED BY:
0042 1
0043 1      V03-009 PRD0061      Paul R. DeStefano      05-Feb-1984
0044 1      Allow OBJECT parameter to accept both name and number.
0045 1
0046 1      V03-008 PRD0053      Paul R. DeStefano      05-Feb-1984
0047 1      Complete addition of and enable X25-Access parsing.
0048 1
0049 1      V03-007 RPG0007      Bob Grosso      19-Feb-1983
0050 1      Accept CIRCUIT circuit-id and KNOWN CIRCUITS after
0051 1      LOOP NODES.
0052 1
0053 1      V03-006 RPG0006      Bob Grosso      15-Nov-1982
0054 1      Accept all information types with SHOW ADJACENT...
0055 1      Accept CIRCUIT circuit-id and KNOWN CIRCUITS after
0056 1      ACTIVE NODES and KNOWN NODES.
0057 1
```

58	0058	1	V03-005	RPG0005	Bob Grosso	28-Sep-1982
59	0059	1		Add Show AREA.		
60	0060	1		Show Module Configurator, Console, Loader, Looper.		
61	0061	1				
62	0062	1	V03-004	RPG0004	Bob Grosso	14-Sep-1982
63	0063	1		Correct prompting. MODULE X-P is now a noiseword		
64	0064	1		when a DTE, or GROUP qualifier is present.		
65	0065	1		Fix SHOW X25-P DTE to prompt for DTE name.		
66	0066	1				
67	0067	1	V03-003	TMH0003	Tim Halvorsen	16-Aug-1982
68	0068	1		Fix SHOW X25-TRACE so that the tracepoint name is		
69	0069	1		sent in the NICE message if it is specified.		
70	0070	1				
71	0071	1	V03-002	RPG0002	Bob Grosso	03-Aug-1982
72	0072	1		Include prompting for X-P.		
73	0073	1		Include prompting for X-S.		
74	0074	1				
75	0075	1	V03-001	RPG0001	Bob Grosso	14-Jul-1982
76	0076	1		Add show module X25-Trace, X29-Server.		
77	0077	1				
78	0078	1	V006	RPG0006	Bob Grosso	13-May-1982
79	0079	1		Add show module X25-protocol and X25-server,		
80	0080	1		X25-Access.		
81	0081	1				
82	0082	1	V005	TMH0005	Tim Halvorsen	08-Nov-1981
83	0083	1		Fix SHOW LOGGING when SUMMARY follows KNOWN SINKS		
84	0084	1		on the command line.		
85	0085	1				
86	0086	1	V004	TMH0004	Tim Halvorsen	25-Aug-1981
87	0087	1		Add SHOW LINK nnn.		
88	0088	1				
89	0089	1	V003	TMH0003	Tim Halvorsen	05-Jul-1981
90	0090	1		Add SHOW MODULE.		
91	0091	1				
92	0092	1	V002	TMH0002	Tim Halvorsen	20-Jul-1981
93	0093	1		Remove special casing of 'LIN' as a LINE entity		
94	0094	1		and require full spelling of 'LINE' to distinguish		
95	0095	1		between lines and links. This makes the ambiguity		
96	0096	1		rules consistent.		
97	0097	1				
98	0098	1	V001	TMH0001	Tim Halvorsen	17-Jun-1981
99	0099	1		Indicate object & link as system-specific entity		
100	0100	1		types in their respective SDB's.		
101	0101	1		Add SHOW CIRCUIT.		
102	0102	1				


```

: 104      0103 1
: 105      0104 1
: 106      0105 1  INCLUDE FILES:
: 107      0106 1
: 108      0107 1
: 109      0108 1      LIBRARY 'OBJ$:NMALIBRY';
: 110      0109 1      LIBRARY 'OBJ$:NCPLIBRY';
: 111      0110 1      LIBRARY 'SYSSLIBRARY:TPAMAC';
: 112      0111 1      LIBRARY 'SYSSLIBRARY:STARLET';
: 113      0112 1
: 114      0113 1
: 115      0114 1  EXTERNAL REFERENCES:
: 116      0115 1
: 117      0116 1
: 118      0117 1      ACT_DFN      ! External symbols for action routines

```

```
120 0118 1 %SBTTL 'Parameter blocks'
121 0119 1
122 0120 1
123 0121 1 BIND DATA:
124 0122 1
125 0123 1
126 0124 1
127 0125 1 Parameter Blocks
128 0126 1
129 0127 1
130 0128 1
131 0129 1 For General Use
132 0130 1
133 0131 1
134 P 0132 1 BUILD_PCL
135 P 0133 1
136 P 0134 1 (DUMMY,
137 P 0135 1
138 P 0136 1 , END, , ,
139 P 0137 1 )
140 0138 1
141 0139 1
142 P 0140 1 BUILD_SDB
143 0141 1 (SAR,-NMA$C_ENT_ARE, VRB_ENT, DUMMY)
144 0142 1
145 0143 1
146 P 0144 1 BUILD_SDB
147 0145 1 (SLI,-NMA$C_ENT_LIN, VRB_ENT, DUMMY)
148 0146 1
149 0147 1
150 0148 1
151 0149 1 BIND PDB$G_SCS_ENT = ! Module Console
152 0150 1 UPLIT-BYTE(0, %ASCIC 'CONSOLE');
153 0151 1
154 P 0152 1 BUILD_SDB
155 0153 1 (SCS,-NMA$C_ENT_MOD, SCS_ENT, DUMMY)
156 0154 1
157 0155 1
158 0156 1 BIND PDB$G_SLD_ENT = ! Module Loader
159 0157 1 UPLIT-BYTE(0, %ASCIC 'LOADER');
160 0158 1
161 P 0159 1 BUILD_SDB
162 0160 1 (SLD,-NMA$C_ENT_MOD, SLD_ENT, DUMMY)
163 0161 1
164 0162 1
165 0163 1 BIND PDB$G_SLP_ENT = ! Module Looper
166 0164 1 UPLIT-BYTE(0, %ASCIC 'LOOPER');
167 0165 1
168 P 0166 1 BUILD_SDB
169 0167 1 (SLP,-NMA$C_ENT_MOD, SLP_ENT, DUMMY)
170 0168 1
171 0169 1
172 0170 1
173 P 0171 1 BUILD_SDB
174 0172 1 (SOB,-NMA$C_SENT_OBJ, VRB_ENT, DUMMY)
175 0173 1
176 0174 1
```



```

: 177 P 0175 1 BUILD_PBK
: 178 P 0176 1
: 179 P 0177 1 (SHL,
: 180 P 0178 1
: 181 P 0179 1 ADJ, LITB, NMASC_ENT_ADJ, VRB_ENT,
: 182 P 0180 1 ACT, LITB, NMASC_ENT_ACT, VRB_ENT,
: 183 P 0181 1 KWN, LITB, NMASC_ENT_KNO, VRB_ENT,
: 184 P 0182 1 LUP, LITB, NMASC_ENT_LOO, VRB_ENT,
: 185 P 0183 1
: 186 P 0184 1 ARE, AREA, , VRB_ENT,
: 187 P 0185 1 TKN, TKN, , VRB_ENT,
: 188 P 0186 1 NOD, NADR, , VRB_ENT,
: 189 P 0187 1 EXE, LITL, 0, VRB_ENT,
: 190 P 0188 1 INF, LITB, 0, ,
: 191 P 0189 1
: 192 P 0190 1 )
: 193 P 0191 1
: 194 P 0192 1 BUILD_PBK
: 195 P 0193 1
: 196 P 0194 1 (INF,
: 197 P 0195 1
: 198 P 0196 1 TO, TKN, , ,
: 199 P 0197 1
: 200 0198 1 )

```

```
.. 202 0199 1 :  
... 203 0200 1 : Show / List Node  
... 204 0201 1 :  
... 205 P 0202 1 BUILD_PCL  
... 206 P 0203 1 (SNO,  
... 207 P 0204 1  
... 208 P 0205 1 CIR, TKN, PCNO_DLI, ,  
... 209 P 0206 1  
... 210 P 0207 1 { END, . . ,  
... 211 0208 1  
... 212 0209 1  
... 213 0210 1  
... 214 P 0211 1 BUILD_PBK  
... 215 P 0212 1 (SNO,  
... 216 P 0213 1  
... 217 P 0214 1 KCI, LITB, NMA$C_ENT_KNO, SNO_CIR,  
... 218 P 0215 1 CIR, TKN, 0, ,  
... 219 P 0216 1  
... 220 0217 1 )  
... 221 0218 1  
... 222 P 0219 1 BUILD_SDB  
... 223 0220 1 (SNO, NMA$C_ENT_NOD, VRB_ENT, SNO)  
... 224 0221 1  
... 225 0222 1 :  
... 226 0223 1 : Show / List Circuit  
... 227 0224 1 :  
... 228 0225 1  
... 229 P 0226 1 BUILD_PCL  
... 230 P 0227 1  
... 231 P 0228 1 (SCI,  
... 232 P 0229 1  
... 233 P 0230 1 NOD, NADR, PCCI_ADJ, ,  
... 234 P 0231 1 { END, . . ,  
... 235 0232 1  
... 236 0233 1  
... 237 0234 1  
... 238 P 0235 1 BUILD_PBK  
... 239 P 0236 1  
... 240 P 0237 1 (SCI,  
... 241 P 0238 1  
... 242 P 0239 1 NOD, NADR, . . ,  
... 243 0240 1 )  
... 244 0241 1  
... 245 0242 1  
... 246 P 0243 1 BUILD_SDB  
... 247 0244 1 (SCI, NMA$C_ENT_CIR, VRB_ENT, SCI)
```



```
249 0245 1 |
250 0246 1 | Show List Logging
251 0247 1 |
252 0248 1 |
253 P 0249 1 BUILD_PCL
254 P 0250 1
255 P 0251 1 (SLO,
256 P 0252 1
257 P 0253 1 SNO, NADR, PCLO_SIN, ,
258 P 0254 1
259 P 0255 1 , END, , ,
260 P 0256 1 )
261 0257 1
262 0258 1
263 P 0259 1 BUILD_PBK
264 P 0260 1
265 P 0261 1 (SLO,
266 P 0262 1
267 P 0263 1 SKN, LITB, NMA$C_ENT_KNO, SLO_SNO,
268 P 0264 1 SEX, LITL, 0, SLO_SNO,
269 P 0265 1 SNO, NADR, , ,
270 P 0266 1 )
271 0267 1
272 0268 1
273 P 0269 1 BUILD_SDB
274 P 0270 1
275 0271 1 (SLO, NMA$C_ENT_LOG, VRB_ENT, SLO)
276 0272 1
277 0273 1 |
278 0274 1 | Show List Links by node
279 0275 1 |
280 0276 1 |
281 P 0277 1 BUILD_PCL
282 P 0278 1
283 P 0279 1 (SLK,
284 P 0280 1
285 P 0281 1 NOD, NADR, PCLK_NID, ,
286 P 0282 1
287 P 0283 1 , END, , ,
288 P 0284 1 )
289 0285 1
290 0286 1
291 P 0287 1 BUILD_PBK
292 P 0288 1
293 P 0289 1 (SLK,
294 P 0290 1
295 P 0291 1 ENT, NADR, , VRB_ENT, ! Link address, not a node name
296 P 0292 1 ! (but using same format as NADR)
297 P 0293 1 NOD, NADR, , ,
298 P 0294 1 )
299 0295 1
300 0296 1
301 P 0297 1 BUILD_SDB
302 P 0298 1
303 0299 1 (SLK, -NMA$C_SENT_LNK, VRB_ENT, SLK)
```

```
305      0300 1 1
306      0301 1 1
307      0302 1 1
308      P 0303 1 1 Show Module Configurator
309      P 0304 1 1 BUILD_PCL
310      P 0305 1 1 (SCF,
311      P 0306 1 1 CIR, TKN, PCCN_CIR, ,
312      P 0307 1 1
313      P 0308 1 1
314      P 0309 1 1 , END, , ,
315      0310 1 1
316      0311 1 1
317      P 0312 1 1 BUILD_PBK
318      P 0313 1 1 (SCF,
319      P 0314 1 1
320      P 0315 1 1 KCI, LITB, NMASC_ENT_KNO, SCF_CIR,
321      P 0316 1 1 CIR, TKN, 0, ,
322      P 0317 1 1
323      0318 1 1 )
324      0319 1 1
325      0320 1 1 BIND PDB$G_CNF_ENT = ! Module Configurator
326      0321 1 1 UPLIT-BYTE(0, %ASCIC 'CONFIGURATOR');
327      0322 1 1
328      P 0323 1 1 BUILD_SDB
329      P 0324 1 1
330      0325 1 1 (SCF, NMASC_ENT_MOD, CNF_ENT, SCF)
331      0326 1 1
332      0327 1 1
333      0328 1 1
334      0329 1 1
335      0330 1 1
336      P 0331 1 1 Show Module X25-Access
337      P 0332 1 1 BUILD_PCL
338      P 0333 1 1 (SAC,
339      P 0334 1 1
340      P 0335 1 1 NET, TKN, PCXA_NET, ,
341      P 0336 1 1
342      P 0337 1 1 , END, , ,
343      0338 1 1
344      0339 1 1
345      P 0340 1 1 BUILD_PBK
346      P 0341 1 1
347      P 0342 1 1 (SAC,
348      P 0343 1 1
349      P 0344 1 1 NET, TKN, NMASC_ENT_KNO, SAC_NET, ! known networks
350      P 0345 1 1 KNT, LITB, NMASC_ENT_KNO, SAC_ENT,
351      0346 1 1 )
352      0347 1 1
353      0348 1 1 BIND PDB$G_SAC_ENT = ! Access entity name
354      0349 1 1 UPLIT-BYTE(0, %ASCIC 'X25-ACCESS');
355      0350 1 1
356      P 0351 1 1 BUILD_SDB
357      P 0352 1 1
358      0353 1 1 (SAC, NMASC_ENT_MOD, SAC_ENT, SAC)
359      0354 1 1
```



```

: 361      0355 1  |
: 362      0356 1  | Show Module X25-Protocol
: 363      0357 1  |
: 364      P 0358 1  | BUILD_PCL
: 365      P 0359 1  |
: 366      P 0360 1  | (SPR,
: 367      P 0361 1  |
: 368      P 0362 1  | DTE, TKN, PCXP_DTE, ;
: 369      P 0363 1  | GRP, TKN, PCXP_GRP, ;
: 370      P 0364 1  |
: 371      P 0365 1  |   END, . . .
: 372      0366 1  | }
: 373      0367 1  |
: 374      P 0368 1  | BUILD_PBK
: 375      P 0369 1  |
: 376      P 0370 1  | (SPR,
: 377      P 0371 1  |
: 378      P 0372 1  | DTE, TKN,
: 379      P 0373 1  | KDT, LITB, 'NMA$C_ENT_KNO, SPR_DTE,      ! known DTEs
: 380      P 0374 1  | GRP, TKN,
: 381      P 0375 1  | KGR, LITB, 'NMA$C_ENT_KNO, SPR_GRP,      ! known Groups
: 382      0376 1  | )
: 383      0377 1  |
: 384      0378 1  | BIND PDB$G_SPR_ENT =                      ! Protocol entity name
: 385      0379 1  | UPLIT BYTE(0, %ASCIC 'X25-PROTOCOL');
: 386      0380 1  |
: 387      P 0381 1  | BUILD_SDB
: 388      P 0382 1  |
: 389      0383 1  | (SPR, NMA$C_ENT_MOD, SPR_ENT, SPR)

```

```
.. 391      0384 1
.. 392      0385 1
.. 393      0386 1
.. 394      0387 1
.. 395      P 0388 1
.. 396      P 0389 1
.. 397      P 0390 1
.. 398      P 0391 1
.. 399      P 0392 1
.. 400      P 0393 1
.. 401      P 0394 1
.. 402      0395 1
.. 403      0396 1
.. 404      P 0397 1
.. 405      P 0398 1
.. 406      P 0399 1
.. 407      P 0400 1
.. 408      P 0401 1
.. 409      P 0402 1
.. 410      0403 1
.. 411      0404 1
.. 412      0405 1
.. 413      0406 1
.. 414      0407 1
.. 415      0408 1
.. 416      P 0409 1
.. 417      P 0410 1
.. 418      0411 1
.. 419      0412 1
.. 420      0413 1
.. 421      0414 1
.. 422      0415 1
.. 423      0416 1
.. 424      0417 1
.. 425      0418 1
.. 426      0419 1
.. 427      0420 1
.. 428      0421 1
.. 429      0422 1
.. 430      0423 1
.. 431      0424 1
.. 432      0425 1
.. 433      P 0426 1
.. 434      P 0427 1
.. 435      0428 1
.. 436      0429 1

Show Module X25-Server
BUILD_PCL
(SSE,
DST, TKN, PCXS_DST, ,
, END, . . .
)
BUILD_PBK
(SSE,
DST, TKN, 'NMA$C_ENT_KNO, SSE_DST,      ! destinations
KDS, LITB, 'NMA$C_ENT_KNO, SSE_DST,
)
BIND
PDBG_SSE_ENT =      ! Server entity name
UPLIT BYTE(0, %ASCIC 'X25-SERVER');
BUILD_SDB
(SSE, NMA$C_ENT_MOD, SSE_ENT, SSE)

Show Module X29-Server

Use PCL and PKBs from X25-Server, only SDB and Entity PDB
are different

BIND
PDBG_S9S_ENT =      ! Server entity name
UPLIT BYTE(0, %ASCIC 'X29-SERVER');
BUILD_SDB
(S9S, NMA$C_ENT_MOD, S9S_ENT, SSE)
```



```

: 438      0430 1
: 439      0431 1
: 440      0432 1
: 441      0433 1
: 442      P 0434 1
: 443      P 0435 1
: 444      P 0436 1
: 445      P 0437 1
: 446      P 0438 1
: 447      P 0439 1
: 448      P 0440 1
: 449      0441 1
: 450      0442 1
: 451      0443 1
: 452      P 0444 1
: 453      P 0445 1
: 454      P 0446 1
: 455      P 0447 1
: 456      P 0448 1
: 457      P 0449 1
: 458      0450 1
: 459      0451 1
: 460      0452 1
: 461      0453 1
: 462      0454 1
: 463      0455 1
: 464      P 0456 1
: 465      P 0457 1
: 466      0458 1

          Show Module X25-Trace
          BUILD_PCL
          (STR,
          TPT, TKN,      PCXT_TPT, ,
          , END, . . .

          BUILD_PBK
          (STR,
          TPT, TKN,
          KTP, LITB, 'NMASC_ENT_KNO, STR_TPT,      ! known TRACEPOINTS
          )

          BIND PDB$G_STR_ENT =      ! Trace entity name
          UPLIT BYTE(0, %ASCIC 'X25-TRACE');

          BUILD_SDB
          (STR, NMASC_ENT_MOD, STR_ENT, STR)
```

```

468      0459 1 %SBTTL 'Prompt strings'
469      0460 1
470      0461 1
471      0462 1
472      0463 1
473      0464 1
474      0465 1 BIND
475      P 0466 1 PROMPT_STRINGS
476      P P 0467 1 (SHL,
477      P P 0468 1
478      P 0469 1 ACT, '(AREAS, CIRCUITS, LINES, LOGGING, NODES): ',
479      P 0470 1 ADJ, '(NODES): ',
480      L 0471 1 ENT, %STRING ('(ACTIVE, ADJACENT, AREA, CIRCUIT, EXECUTOR, ', CRLF,
481      P 0472 1 'KNOWN, LINE, LOGGING, LOOP, MODULE, NODE, OBJECT): '),
482      P P 0473 1 KWN, '(AREAS, CIRCUITS, LINES, LINKS, LOGGING, NODES, OBJECTS): ',
483      L 0474 1 MOD, %STRING ('Module (CONFIGURATOR, CONSOLE, LOADER, ', CRLF,
484      L 0475 1 'LOOPER, X25-ACCESS, X25-PROTOCOL, ', CRLF,
485      P 0476 1 'X25-SERVER, X25-TRACE, X29-SERVER): '),
486      P 0477 1
487      0478 2 )
488      0479 1 ,
489      0480 1
490      P 0481 1 PROMPT_STRINGS
491      P P 0482 1 (SAR,
492      P P 0483 1
493      P 0484 1 ENT, 'Area number (integer): ',
494      P 0485 1
495      0486 2 )
496      0487 1 ,
497      0488 1
498      P 0489 1 PROMPT_STRINGS
499      P P 0490 1 (SCI,
500      P P 0491 1
501      P 0492 1 ENT, 'Circuit name (16 characters): ',
502      P 0493 1
503      0494 1 ),
504      0495 1
505      P 0496 1 PROMPT_STRINGS
506      P P 0497 1 (SLI,
507      P P 0498 1
508      P 0499 1 ENT, 'Line ID (dev-c-u.t): ',
509      P 0500 1
510      0501 1 ),
511      0502 1
512      P 0503 1 PROMPT_STRINGS
513      P P 0504 1 (SLK,
514      P P 0505 1
515      P 0506 1 ENT, 'Link address (0-65535): ',
516      P 0507 1
517      0508 2 )
518      0509 1 ;

```



```
520 0510 1
521 0511 1 BIND
522 P 0512 1 PROMPT_STRINGS
523 P 0513 1 (SLO,
524 P 0514 1
525 P 0515 1 ENT, 'Type of logging (CONSOLE, FILE, MONITOR): ',
526 P 0516 1 SNO, 'Sink node (node-id, EXECUTOR): ',
527 P 0517 1
528 0518 2 )
529 0519 1 ,
530 0520 1
531 P 0521 1 PROMPT_STRINGS
532 P 0522 1 (SNO,
533 P 0523 1
534 P 0524 1 ENT, 'Node ID (node-name, address): ',
535 P 0525 1
536 0526 2 )
537 0527 1 ,
538 0528 1
539 P 0529 1 PROMPT_STRINGS
540 P 0530 1 (SOB,
541 P 0531 1
542 P 0532 1 ENT, 'Object name (8 characters): ',
543 P 0533 1
544 0534 1 ),
545 0535 1
546 P 0536 1 PROMPT_STRINGS
547 P 0537 1 (SCF,
548 P 0538 1
549 P 0539 1 DAT, '(CIRCUIT, KNOWN CIRCUITS): ',
550 P 0540 1 CIR, 'CIRCUIT name (16 characters): ',
551 P 0541 1
552 0542 1 ),
553 0543 1
554 P 0544 1 PROMPT_STRINGS
555 P 0545 1 (SAC,
556 P 0546 1
557 P 0547 1 DAT, '(NETWORK name, KNOWN NETWORKS): ',
558 P 0548 1 NET, 'NETWORK name (16 Characters): ',
559 P 0549 1
560 0550 1 ),
561 0551 1
562 P 0552 1 PROMPT_STRINGS
563 P 0553 1 (SPR,
564 P 0554 1
565 P 0555 1 DAT, '(DTE, GROUP, KNOWN DTES, KNOWN GROUPS): ',
566 P 0556 1 DTE, '(DTE name): ',
567 P 0557 1 GRP, '(GROUP name): ',
568 P 0558 1
569 0559 1 ),
570 0560 1
571 P 0561 1 PROMPT_STRINGS
572 P 0562 1 (SSE,
573 P 0563 1
574 P 0564 1 DAT, '(DESTINATION, KNOWN DESTINATIONS): ',
575 P 0565 1 DST, '(DESTINATION name): ',
576 P 0566 1
```

```

577      0567 1      ),
578      0568 1
579      P 0569 1      PROMPT_STRINGS
580      P 0570 1      (STR,
581      P 0571 1
582      P 0572 1      DAT, '(TRACEPOINT, KNOWN TRACEPOINTS): ',
583      P 0573 1      TPT, '(TRACEPOINT name): ',
584      P 0574 1
585      0575 1      ),
586      0576 1
587      P 0577 1      PROMPT_STRINGS
588      P 0578 1      (S9S,
589      P 0579 1
590      P 0580 1      DAT, '(DESTINATION, KNOWN DESTINATIONS): ',
591      P 0581 1      DST, '(DESTINATION name): ',
592      P 0582 1
593      0583 2      )
594      0584 1      ;

```

.....


```
596 0585 1 XSBTTL 'State Table Entry'
597 0586 1
598 0587 1 $INIT_STATE (NCP$G_STTBL_SHL, NCP$G_KYTBL_SHL);
599 0588 1
600 0589 1
601 0590 1
602 0591 1
603 0592 1
604 0593 1
605 0594 1
606 0595 1
607 0596 1
608 P 0597 1 COMMAND PROMPT
609 P 0598 1 (SHL, ENT, NCP$ INVKEY,
610 P 0599 1
611 P 0600 1 ( (SE INFO_TYPES), ST_SHL_ENT),
612 P 0601 1 ('ACTIVE', ST_SHL_ACT, ACT$SAVPRM, , , PBK$G_SHL_ACT),
613 P 0602 1 ('ADJACENT', ST_SHL_ADJ, ACT$SAVPRM, , , PBK$G_SHL_ADJ),
614 P 0603 1 ('AREA', ST_SAR_ENT),
615 P 0604 1 ('CIRCUIT', ST_SCI_ENT),
616 P 0605 1 ('CONFIGURATOR', ST_SCF_DAT),
617 P 0606 1 ('CONSOLE', ST_SCS_ENT),
618 P 0607 1 ('DTE', ST_SPR_DTE, NMASC_ENT_MOD, NCP$GL_OPTION),
619 P 0608 1 ('GROUP', ST_SPR_GRP, NMASC_ENT_MOD, NCP$GL_OPTION),
620 P 0609 1 ('EXECUTOR', ST_SNO_GO, ACT$SAVPRM, NMASC_ENT_MOD,
621 P 0610 1 NCP$GL_OPTION, PBK$G_SHL_EXE),
622 P 0611 1 ('KNOWN', ST_SHL_KWN, ACT$SAVPRM, , , PBK$G_SHL_KWN),
623 P 0612 1 ('LINE', ST_SLI_ENT),
624 P 0613 1 ('LINK', ST_SLK_ENT),
625 P 0614 1 ('LOADER', ST_SLD_ENT),
626 P 0615 1 ('LOGGING', ST_SLO_ENT),
627 P 0616 1 ('LOOP', ST_SHL_LUP, ACT$SAVPRM, , , PBK$G_SHL_LUP),
628 P 0617 1 ('LOOPER', ST_SLP_ENT),
629 P 0618 1 ('MODULE', ST_SHL_MOD),
630 P 0619 1 ('NODE', ST_SNO_ENT),
631 P 0620 1 ('OBJECT', ST_SOB_ENT),
632 P 0621 1 ('TRACEPOINT', ST_STR_TPT, NMASC_ENT_MOD, NCP$GL_OPTION),
633 P 0622 1 ('X25', ST_SHL_X25),
634 P 0623 1 ('X29', ST_SHL_X29),
635 P 0624 1
636 0625 1 )
```

```
638 0626 1 %SBTTL 'Active, Adjacent, Known and Loop Entities'
639 0627 1
640 0628 1
641 0629 1
642 0630 1
643 0631 1
644 0632 1
645 0633 1
646 0634 1
647 0635 1
648 P 0636 1
649 P 0637 1
650 P 0638 1
651 P 0639 1
652 P 0640 1
653 P 0641 1
654 P 0642 1
655 P 0643 1
656 P 0644 1
657 0645 1
658 0646 1
659 0647 1
660 0648 1
661 0649 1
662 0650 1
663 P 0651 1
664 P 0652 1
665 P 0653 1
666 P 0654 1
667 P 0655 1
668 0656 1
669 0657 1
670 P 0658 1
671 P 0659 1
672 P 0660 1
673 P 0661 1
674 P 0662 1
675 0663 1
676 0664 1
677 P 0665 1
678 P 0666 1
679 0667 1
680 0668 1
681 P 0669 1
682 P 0670 1
683 0671 1

%SBTTL 'Active, Adjacent, Known and Loop Entities'

For each, dispatch or prompt if end of string

Active Entities

COMMAND PROMPT
(SHL, ACT, NCP$ INVKEY,
('AREAS', ST_SAR_GO),
('CIRCUITS', ST_SCI_GO),
('LINES', ST_SLI_GO),
('LOGGING', ST_SLO_GO),
('NODES', ST_ADJ_NOD) ! Use same path for Active nodes as for Adjacent nodes
)

Adjacent Entities

COMMAND PROMPT
(SHL, ADJ, NCP$ INVKEY,
('NODES', ST_ADJ_NOD)
)

$STATE (ST_ADJ_NOD,
('CIRCUIT', ST_ADJ_CIR),
('KNOWN', ST_ADJ_KCI),
(TPAS_LAMBDA, ST_SNO_GO),
(TPAS_EOS, ST_SNO_GOT);
)

$STATE (ST_ADJ_CIR,
((SE_CIRC_ID), ST_SNO_GO, ACT$SAVPRM, , , PBK$G_SNO_CIR)
);

$STATE (ST_ADJ_KCI,
('CIRCUITS', ST_SNO_GO, ACT$SAVPRM, , , PBK$G_SNO_KCI)
);
```



```
685 0672 1 !
686 0673 1 !
687 0674 1 !
688 0675 1 !
689 P 0676 1 COMMAND PROMPT
690 P 0677 1 (SHL, KWN, NCP$_INVKEY,
691 P 0678 1
692 P 0679 1 ('AREAS', ST_SAR_GO),
693 P 0680 1 ('CIRCUITS', ST_SCI_GO),
694 P 0681 1 ('DTES', ST_SPR_DOIT, ACT$SAVPRM, NMA$C_ENT_MOD, NCP$GL_OPTION, PBK$G_SPR_KDT),
695 P 0682 1 ('GROUPS', ST_SPR_DOIT, ACT$SAVPRM, NMA$C_ENT_MOD, NCP$GL_OPTION, PBK$G_SPR_KGR),
696 P 0683 1 ('LINES', ST_SLI_GO),
697 P 0684 1 ('LINKS', ST_SLK_KWN),
698 P 0685 1 ('LOGGING', ST_SLO_GO),
699 P 0686 1 ('NODES', ST_ADJ_NOD), ! Use same path for KNOWN NODES as for ADJACENT NODES
700 P 0687 1 ('OBJECTS', ST_SOB_GO),
701 P 0688 1 ('TRACEPOINTS', ST_STR_DOIT, ACT$SAVPRM, NMA$C_ENT_MOD, NCP$GL_OPTION, PBK$G_STR_KTP)
702 P 0689 1 )
703 0690 1
704 0691 1
705 0692 1 !
706 0693 1 !
707 0694 1 !
708 0695 1 !
709 P 0696 1 $STATE (ST_SHL_LUP,
710 P 0697 1 ('NODES', ST_LUP_NOD), ! Ignore noise word
711 P 0698 1 (TPAS_LAMBDA, ST_SNO_GO) ! Nodes only are valid
712 0699 1 );
713 0700 1
714 P 0701 1 $STATE (ST_LUP_NOD,
715 P 0702 1 ('CIRCUIT', ST_LUP_CIR),
716 P 0703 1 ('KNOWN', ST_LUP_KCI),
717 P 0704 1 (TPAS_LAMBDA, ST_SNO_GO),
718 P 0705 1 (TPAS_EOS, ST_SNO_GO),
719 0706 1 );
720 0707 1
721 P 0708 1 $STATE (ST_LUP_CIR,
722 P 0709 1 (SE_CIRC_ID), ST_SNO_GO, ACT$SAVPRM, , , PBK$G_SNO_CIR)
723 0710 1 );
724 0711 1
725 P 0712 1 $STATE (ST_LUP_KCI,
726 P 0713 1 ('CIRCUITS', ST_SNO_GO, ACT$SAVPRM, , , PBK$G_SNO_KCI)
727 0714 1 );
```

```

: 729      0715 1 %SBTTL 'Show / List Area'
: 730      0716 1
: 731      0717 1
: 732      0718 1
: 733      0719 1
: 734      0720 1
: 735      0721 1
: 736      0722 1
: 737      0723 1
: 738      0724 1
: 739      P 0725 1
: 740      P 0726 1
: 741      P 0727 1
: 742      P 0728 1
: 743      P 0729 1
: 744      0730 1
: 745      0731 1
: 746      0732 1
: 747      0733 1
: 748      0734 1
: 749      0735 1
: 750      P 0736 1
: 751      P 0737 1
: 752      0738 1
: 753      0739 1
: 754      0740 1
: 755      0741 1
: 756      0742 1
: 757      0743 1
: 758      P 0744 1
: 759      P 0745 1
: 760      0746 1
: 761      0747 1
: 762      P 0748 1
: 763      P 0749 1
: 764      0750 1

      %SBTTL 'Show / List Area'

      Show / List Area

      Collect the Area number or prompt

      COMMAND PROMPT
      (SAR, ENT, NCP$INVVAL,
      ( (SE_AREA_NUM), , ACT$SAVPRM, , , PBK$G_SHL_ARE)
      )

      Collect the information type

      $STATE (ST_SAR_GO,
      ( (ST_INF_TYPE2) )
      );

      Perform the function

      $STATE (
      (TPAS_LAMBDA, , , NMASC_ENT_ARE, NCP$GL_OPTION, )
      );

      $STATE (
      (TPAS_EOS, TPAS_EXIT, ACT$VRB_SHOLIS, , , SDB$G_SAR)
      );
```



```

766 0751 1 %SBTTL 'Show / List Circuit'
767 0752 1
768 0753 1
769 0754 1 Show / List Circuit
770 0755 1
771 0756 1
772 0757 1
773 0758 1 Collect the circuit name or prompt
774 0759 1
775 0760 1
776 P 0761 1 COMMAND PROMPT
777 P 0762 1 (SCI, ENT, NCP$INVVAL,
778 P 0763 1
779 P 0764 1 ( (SE_CIRC_ID), , ACT$SAVPRM, , , PBK$G_SHL_TKN)
780 P 0765 1
781 0766 1 )
782 0767 1
783 0768 1
784 0769 1 Collect the information type
785 0770 1
786 0771 1
787 P 0772 1 $STATE (ST_SCI_GO,
788 P 0773 1 ( (ST_INF_TYPE2))
789 0774 1 );
790 0775 1
791 P 0776 1 $STATE (
792 P 0777 1 ('ADJACENT'),
793 P 0778 1 (TPAS_LAMBDA, ST_SCI_GONE),
794 0779 1 );
795 0780 1
796 P 0781 1 $STATE (
797 P 0782 1 ('NODE')
798 0783 1 );
799 0784 1
800 P 0785 1 $STATE (
801 P 0786 1 ((SE_NODE_ID), , ACT$SAVPRM, , , PBK$G_SCI_NOD),
802 0787 1 );
803 0788 1
804 P 0789 1 $STATE (
805 P 0790 1 ( (ST_INF_TYPE2) )
806 0791 1 );
807 0792 1
808 0793 1
809 0794 1 Perform the function
810 0795 1
811 0796 1
812 P 0797 1 $STATE (ST_SCI_GONE,
813 P 0798 1 (TPAS_LAMBDA, , , NMASC_ENT_CIR, NCP$GL_OPTION, )
814 0799 1 );
815 0800 1
816 P 0801 1 $STATE (
817 P 0802 1 (TPAS_EOS, TPAS_EXIT, ACT$VRB_SHOLIS, , , SDB$G_SCI)
818 0803 1 );
```

```

820      0804 1 %SBTTL 'Show / List Line'
821      0805 1
822      0806 1
823      0807 1 Show / List Line
824      0808 1
825      0809 1
826      0810 1
827      0811 1 Collect the line ID or prompt
828      0812 1
829      0813 1
830      P 0814 1 COMMAND PROMPT
831      P 0815 1 (SLI, ENT, NCP$_INVVAL,
832      P 0816 1
833      P 0817 1 ( (SE_LINE_ID), , ACT$SAVPRM, , , PBK$G_SHL_TKN)
834      P 0818 1
835      0819 1 )
836      0820 1
837      0821 1
838      0822 1 Collect the information type
839      0823 1
840      0824 1
841      P 0825 1 $STATE (ST_SLI_GO,
842      P 0826 1 ( (ST_INF_TYPE2) )
843      0827 1 );
844      0828 1
845      0829 1
846      0830 1 Perform the function
847      0831 1
848      0832 1
849      P 0833 1 $STATE (
850      P 0834 1 (TPAS_LAMBDA, , , NMAC$ENT_LIN, NCP$GL_OPTION, )
851      0835 1 );
852      0836 1
853      P 0837 1 $STATE (
854      P 0838 1 (TPAS_EOS, TPAS_EXIT, ACT$VRB_SHOLIS, , , SDB$G_SLI)
855      0839 1 );

```



```
.. 857      0840 1 %SBTTL 'Show / List Links'
.. 858      0841 1
.. 859      0842 1
.. 860      0843 1 Show / List Links
.. 861      0844 1
.. 862      0845 1
.. 863      0846 1
.. 864      0847 1 Collect the Link address or prompt
.. 865      0848 1
.. 866      0849 1
.. 867      P 0850 1 COMMAND PROMPT
.. 868      P 0851 1 (SLK, ENT, NCP$_INVVAL,
.. 869      P 0852 1
.. 870      P 0853 1 ((SE_LINK_ID), ST_SLK_GO, ACT$SAVPRM, , , PBK$G_SLK_ENT)
.. 871      0854 1 )
.. 872      0855 1
.. 873      0856 1
.. 874      0857 1 Show known links (and optionally select only those with a given node)
.. 875      0858 1
.. 876      0859 1
.. 877      P 0860 1 $STATE (ST_SLK_KWN,
.. 878      P 0861 1 ('WITH'),
.. 879      P 0862 1 (TPAS_LAMBDA, ST_SLK_GO)
.. 880      0863 1 );
.. 881      0864 1
.. 882      P 0865 1 $STATE (
.. 883      P 0866 1 ('NODE'),
.. 884      0867 1 );
.. 885      0868 1
.. 886      P 0869 1 $STATE (
.. 887      P 0870 1 ((SE_NODE_ID), ST_SLK_GO, ACT$SAVPRM, , , PBK$G_SLK_NOD),
.. 888      0871 1 )
.. 889      0872 1
.. 890      0873 1
.. 891      0874 1 Collect the information type
.. 892      0875 1
.. 893      0876 1
.. 894      P 0877 1 $STATE (ST_SLK_GO,
.. 895      P 0878 1 ( (ST_INF_TYPE1) )
.. 896      0879 1 );
.. 897      0880 1
.. 898      0881 1
.. 899      0882 1 Perform the function
.. 900      0883 1
.. 901      0884 1
.. 902      P 0885 1 $STATE (
.. 903      P 0886 1 (TPAS_LAMBDA, , , NMAC_SENT_LNK, NCP$GL_OPTION)
.. 904      0887 1 );
.. 905      0888 1
.. 906      P 0889 1 $STATE (
.. 907      P 0890 1 (TPAS_EOS, TPAS_EXIT, ACT$VRB_SHOLIS, , , SDB$G_SLK)
.. 908      0891 1 );
.. 909      0892 1
```

```

: 911      0893 1 %SBTTL 'Show / List Logging'
: 912      0894 1
: 913      0895 1
: 914      0896 1
: 915      0897 1
: 916      0898 1
: 917      0899 1
: 918      0900 1
: 919      0901 1
: 920      0902 1
: 921      P 0903 1
: 922      P 0904 1
: 923      P 0905 1
: 924      P 0906 1
: 925      P 0907 1
: 926      0908 1
: 927      0909 1
: 928      0910 1
: 929      0911 1
: 930      0912 1
: 931      0913 1
: 932      P 0914 1
: 933      P 0915 1
: 934      0916 1
: 935      0917 1
: 936      0918 1
: 937      0919 1
: 938      0920 1
: 939      0921 1
: 940      P 0922 1
: 941      P 0923 1
: 942      P 0924 1
: 943      P 0925 1
: 944      0926 1
: 945      0927 1
: 946      0928 1
: 947      0929 1
: 948      0930 1
: 949      0931 1
: 950      P 0932 1
: 951      P 0933 1
: 952      P 0934 1
: 953      0935 1

      1 %SBTTL 'Show / List Logging'
      1
      1
      1 Show / List Logging
      1
      1
      1 Collect the logging type
      1
      1
      1 COMMAND PROMPT
      1 (SLO, ENT, NCP$_INVVAL,
      1 ( (SE_LOG_TYP) )
      1 )
      1
      1
      1 Now the information type
      1
      1
      1 $STATE (ST_SLO_GO,
      1 ( (ST_INF_TYPE3) )
      1 );
      1
      1
      1 Now any remaining qualifiers
      1
      1
      1 $STATE (
      1 ('SINK', ST_SLO_NOD),
      1 ('KNOWN', ST_SLO_SKN),
      1 (TPAS_LAMBDA, ST_SLO_DOIT),
      1 );
      1
      1
      1 Parse rest of SINK NODE xxx
      1
      1
      1 $STATE (ST_SLO_NOD,
      1 ('NODE'),
      1 (TPAS_LAMBDA)
      1 );

```



```

: 955      0936 1
: 956      0937 1
: 957      0938 1
: 958      0939 1
: 959      0940 1
: 960      P 0941 1
: 961      P 0942 1
: 962      P 0943 1
: 963      P 0944 1
: 964      P 0945 1
: 965      P 0946 1
: 966      0947 1
: 967      0948 1
: 968      0949 1
: 969      0950 1
: 970      0951 1
: 971      0952 1
: 972      P 0953 1
: 973      P 0954 1
: 974      P 0955 1
: 975      0956 1
: 976      0957 1
: 977      P 0958 1
: 978      P 0959 1
: 979      0960 1
: 980      0961 1
: 981      0962 1
: 982      0963 1
: 983      0964 1
: 984      0965 1
: 985      P 0966 1
: 986      P 0967 1
: 987      0968 1
: 988      0969 1
: 989      P 0970 1
: 990      P 0971 1
: 991      0972 1
: 992      0973 1
: 993      P 0974 1
: 994      P 0975 1
: 995      0976 1

      The node id for the sink node

COMMAND PROMPT
(SLO, SNO, NCP$_INVVAL,
('EXECUTOR', ST_SLO_DOIT, ACT$_SAVPRM, , PBK$_SLO_SEX),
( (SE_NODE_ID), ST_SLO_DOIT, ACT$_SAVPRM, , , PBK$_SLO_SNO)
)

Parse rest of KNOWN SINKS

$STATE (ST_SLO_SKN,
('SINKS'),
(TPAS_LAMBDA)
);

$STATE (
(TPAS_LAMBDA, , ACT$_SAVPRM, , , PBK$_SLO_SKN)
);

Perform the function

$STATE (ST_SLO_DOIT,
((ST_INF_TYPE3))
);
! Collect info type here too

$STATE (
(TPAS_LAMBDA, , , NMASC_ENT_LOG, NCP$_GL_OPTION)
);

$STATE (
(TPAS_EOS, TPAS_EXIT, ACT$_VRB_SHOLIS, , , SDB$_SLO)
);
```

```

: 997      0977 1 %SBTTL 'Show / List Node'
: 998      0978 1
: 999      0979 1
1000      0980 1      Show / List Node
1001      0981 1
1002      0982 1
1003      0983 1
1004      0984 1      Obtain the node id
1005      0985 1
1006      0986 1
1007      P 0987 1      COMMAND PROMPT
1008      P 0988 1      (SNO, ENT, NCP$_INVVAL,
1009      P 0989 1
1010      P 0990 1      ( (SE_NODE_ID), , ACT$SAVPRM, , , PBK$G_SHL_NOD)
1011      P 0991 1
1012      0992 1      )
1013      0993 1
1014      0994 1
1015      0995 1      Now the information type
1016      0996 1
1017      0997 1
1018      P 0998 1 $STATE (ST_SNO GO
1019      P 0999 1      ( (ST_INF_TYPE2) )
1020      1000 1      );
1021      1001 1
1022      1002 1
1023      1003 1      Now perform the function
1024      1004 1
1025      1005 1
1026      P 1006 1 $STATE (
1027      P 1007 1      (TPAS_LAMBDA, , , NMASC_ENT_NOD, NCP$GL_OPTION, )
1028      1008 1      );
1029      1009 1
1030      P 1010 1 $STATE (
1031      P 1011 1      (TPAS_EOS, TPAS_EXIT, ACT$VRB_SHOLIS, , , SDB$G_SNO)
: 1032      1012 1      );
```



```
: 1034      1013 1 %SBTTL 'Show / List Objects'
: 1035      1014 1
: 1036      1015 1
: 1037      1016 1
: 1038      1017 1
: 1039      1018 1
: 1040      1019 1
: 1041      1020 1
: 1042      1021 1
: 1043      1022 1
: 1044      1023 1
: 1045      1024 1
: 1046      1025 1
: 1047      1026 1
: 1048      1027 1
: 1049      1028 1
: 1050      1029 1
: 1051      1030 1
: 1052      1031 1
: 1053      1032 1
: 1054      1033 1
: 1055      1034 1
: 1056      1035 1
: 1057      1036 1
: 1058      1037 1
: 1059      1038 1
: 1060      1039 1
: 1061      1040 1
: 1062      1041 1
: 1063      1042 1
: 1064      1043 1
: 1065      1044 1
: 1066      1045 1
: 1067      1046 1
: 1068      1047 1
: 1069      1048 1
: 1070      1049 1
```

1013 1 %SBTTL 'Show / List Objects'

1014 1

1015 1

1016 1

1017 1

1018 1

1019 1

1020 1

1021 1

1022 1

1023 1

1024 1

1025 1

1026 1

1027 1

1028 1

1029 1

1030 1

1031 1

1032 1

1033 1

1034 1

1035 1

1036 1

1037 1

1038 1

1039 1

1040 1

1041 1

1042 1

1043 1

1044 1

1045 1

1046 1

1047 1

1048 1

1049 1

```
: 1072      1050 1 %SBTTL 'Show / List Configurator'
: 1073      1051 1
: 1074      1052 1
: 1075      1053 1
: 1076      1054 1
: 1077      1055 1
: 1078      1056 1
: 1079      1057 1
: 1080      1058 1
: 1081      1059 1
: 1082      P 1060 1
: 1083      P 1061 1
: 1084      P 1062 1
: 1085      P 1063 1
: 1086      P 1064 1
: 1087      P 1065 1
: 1088      1066 1
: 1089      1067 1
: 1090      P 1068 1
: 1091      P 1069 1
: 1092      1070 1
: 1093      1071 1
: 1094      P 1072 1
: 1095      P 1073 1
: 1096      P 1074 1
: 1097      1075 1
: 1098      1076 1
: 1099      1077 1
: 1100      1078 1
: 1101      1079 1
: 1102      1080 1
: 1103      P 1081 1
: 1104      P 1082 1
: 1105      1083 1
: 1106      1084 1
: 1107      1085 1
: 1108      1086 1
: 1109      1087 1
: 1110      1088 1
: 1111      P 1089 1
: 1112      P 1090 1
: 1113      1091 1
: 1114      1092 1
: 1115      P 1093 1
: 1116      P 1094 1
: 1117      1095 1
: 1118      1096 1

%SBTTL 'Show / List Configurator'

Show / List Configurator

Obtain the Configurator qualifier type or prompt

COMMAND PROMPT
(SCF, DAT, NCP$_INVVAL,
('KNOWN', ST_CNF_KWN),
('CIRCUIT', ST_CNF_CIR)
)

$STATE (ST_CNF_KWN,
('CIRCUITS', ST_SCF_GO, ACT$SAVPRM, , , PBK$G_SCF_KCI)
);

$STATE (ST_CNF_CIR,
( (SE_CIRC_ID), ACT$SAVPRM, , , PBK$G_SCF_CIR),
(TPAS_EOS, ST_CNF_CIR, ACT$PRMPT, , , PMT$G_SCF_CIR)
);

Obtain the information type desired

$STATE (ST_SCF_GO,
( (ST_INF_TYPE1) ),
);

Perform the function

$STATE (
(TPAS_LAMBDA, , , NMASC_ENT_MOD, NCP$GL_OPTION, )
);

$STATE (
(TPAS_EOS, TPAS_EXIT, ACT$VRB_SHOLIS, , , SDB$G_SCF)
);
```



```
: 1120      1097 1 %SBTTL 'Show / List Console'
: 1121      1098 1
: 1122      1099 1
: 1123      1100 1
: 1124      1101 1
: 1125      1102 1
: 1126      P 1103 1 $STATE (ST_SCS_ENT,      !      Obtain the information type desired
: 1127      P 1104 1      ( (ST_INF_TYPE1) ),
: 1128      1105 1      );
: 1129      1106 1
: 1130      1107 1
: 1131      1108 1
: 1132      1109 1
: 1133      1110 1
: 1134      P 1111 1 $STATE (
: 1135      P 1112 1      (TPAS_LAMBDA, , , NMASC_ENT_MOD, NCP$GL_OPTION, )
: 1136      1113 1      );
: 1137      1114 1
: 1138      P 1115 1 $STATE (
: 1139      P 1116 1      (TPAS_EOS, TPAS_EXIT, ACT$VRB_SHOLIS, , , SDB$G_SCS)
: 1140      1117 1      );
: 1141      1118 1
```

```
: 1143      1119 1 %SBTTL  'Show / List Loader'
: 1144      1120 1
: 1145      1121 1
: 1146      1122 1
: 1147      1123 1
: 1148      1124 1
: 1149      P 1125 1 $STATE (ST_SLD_ENT,          !      Obtain the information type desired
: 1150      P 1126 1      ( (ST_INF_TYPE1) ),
: 1151      1127 1      );
: 1152      1128 1
: 1153      1129 1
: 1154      1130 1
: 1155      1131 1
: 1156      1132 1
: 1157      P 1133 1 $STATE (
: 1158      P 1134 1      (TPAS_LAMBDA, , , NMASC_ENT_MOD, NCP$GL_OPTION, )
: 1159      1135 1      );
: 1160      1136 1
: 1161      P 1137 1 $STATE (
: 1162      P 1138 1      (TPAS_EOS, TPAS_EXIT, ACT$VRB_SHOLIS, , , SDB$G_SLD)
: 1163      1139 1      );
: 1164      1140 1
```



```
: 1166      1141 1 %SBTTL 'Show / List Looper'
: 1167      1142 1
: 1168      1143 1
: 1169      1144 1
: 1170      1145 1
: 1171      1146 1
: 1172      P 1147 1 $STATE (ST_SLP_ENT,          ! Obtain the information type desired
: 1173      P 1148 1      ( (ST_INF_TYPE1) ),
: 1174      1149 1      );
: 1175      1150 1
: 1176      1151 1
: 1177      1152 1
: 1178      1153 1
: 1179      1154 1
: 1180      P 1155 1 $STATE (
: 1181      P 1156 1      (TPAS_LAMBDA, , , NMASC_ENT_MOD, NCP$GL_OPTION, )
: 1182      1157 1      );
: 1183      1158 1
: 1184      P 1159 1 $STATE (
: 1185      P 1160 1      (TPAS_EOS, TPAS_EXIT, ACT$VRB_SHOLIS, , , SDB$G_SLP)
: 1186      1161 1      );
: 1187      1162 1
```

```
: 1189      1163 1 %SBTTL 'Show / List Module'
: 1190      1164 1
: 1191      1165 1
: 1192      1166 1
: 1193      1167 1
: 1194      1168 1
: 1195      1169 1
: 1196      1170 1
: 1197      1171 1
: 1198      1172 1
: 1199      P 1173 1
: 1200      P 1174 1
: 1201      P 1175 1
: 1202      P 1176 1
: 1203      P 1177 1
: 1204      P 1178 1
: 1205      P 1179 1
: 1206      P 1180 1
: 1207      P 1181 1
: 1208      1182 1
: 1209      1183 1
: 1210      P 1184 1
: 1211      1185 1
: 1212      P 1186 1
: 1213      P 1187 1
: 1214      P 1188 1
: 1215      P 1189 1
: 1216      P 1190 1
: 1217      1191 1
: 1218      1192 1
: 1219      P 1193 1
: 1220      1194 1
: 1221      P 1195 1
: 1222      P 1196 1
: 1223      1197 1

      %SBTTL 'Show / List Module'

      Show / List Module

      Obtain the module name or prompt

      COMMAND PROMPT
      (SHL, MOD, NCP$_INVVAL,

      ('CONFIGURATOR', ST_SCF_DAT),
      ('CONSOLE', ST_SCS_ENT),
      ('LOADER', ST_SLD_ENT),
      ('LOOPER', ST_SLP_ENT),
      ('X25', ST_SHL_X25),
      ('X29', ST_SHL_X29)
      )

      $STATE (ST_SHL_X25,
      ('-T'));

      $STATE (
      ('ACCESS', ST_SAC_GO,, NMASC_ENT_MOD, NCP$GL_OPTION),
      ('PROTOCOL', ST_SPR_GO,, NMASC_ENT_MOD, NCP$GL_OPTION),
      ('SERVER', ST_SSE_GO,, NMASC_ENT_MOD, NCP$GL_OPTION),
      ('TRACE', ST_STR_GO,, NMASC_ENT_MOD, NCP$GL_OPTION)
      );

      $STATE (ST_SHL_X29,
      ('-T'));

      $STATE (
      ('SERVER', ST_S9S_GO,, NMASC_ENT_MOD, NCP$GL_OPTION)
      );
```



```
1225 1198 1 |
1226 1199 1 | SHOW MODULE X25-ACCESS
1227 1200 1 |
1228 1201 1 |
1229 P 1202 1 $STATE (ST_SAC_GO,
1230 P 1203 1 (TPAS_EOS, ST_SAC_PMT_DAT),
1231 P 1204 1 (TPAS_LAMBDA, ST_SAC_DAT)
1232 1205 1 );
1233 1206 1
1234 P 1207 1 $STATE (ST_SAC_PMT_DAT,
1235 1208 1 (TPAS_LAMBDA, ACT$PRMPT, , PMT$G_SAC_DAT));
1236 1209 1
1237 P 1210 1 $STATE (ST_SAC_DAT,
1238 P 1211 1 ((ST_INF_TYPE1))
1239 1212 1 );
1240 1213 1
1241 P 1214 1 $STATE (
1242 P 1215 1 ('KNOWN', ST_SAC_DAT_KNW)
1243 P 1216 1 ('NETWORK', ST_SAC_DAT_NET),
1244 1217 1 );
1245 1218 1
1246 1219 1
1247 P 1220 1 $STATE (ST_SAC_DAT_NET,
1248 1221 1 (TPAS_LAMBDA));
1249 1222 1
1250 1223 1 |
1251 1224 1 | Collect the NETWORK name or prompt
1252 1225 1 |
1253 1226 1 |
1254 P 1227 1 COMMAND PROMPT
1255 P 1228 1 (SAC, NET, NCP$_INVVAL,
1256 P 1229 1
1257 P 1230 1 ((SE_NET_NAME), ST_SAC_DOIT, ACT$SAVPRM, , , PBK$G_SAC_NET),
1258 1231 1 )
1259 1232 1
1260 1233 1
1261 P 1234 1 $STATE (ST_SAC_DAT_KNW,
1262 P 1235 1 (TPAS_EOS, TPAS_EXIT, ACT$SAVPRM, , , PBK$G_SAC_KNT),
1263 P 1236 1 ('NETWORKS', , ACT$SAVPRM, , , PBK$G_SAC_KNT),
1264 P 1237 1 (TPAS_LAMBDA, ST_SAC_DOIT)
1265 1238 1 );
1266 1239 1
1267 1240 1
1268 P 1241 1 $STATE (ST_SAC_DOIT,
1269 P 1242 1 ((ST_INF_TYPE1))
1270 1243 1 );
1271 1244 1
1272 P 1245 1 $STATE (
1273 P 1246 1 (TPAS_EOS, TPAS_EXIT, ACT$VRB_SHOLIS, , , SDB$G_SAC)
1274 1247 1 );
1275 1248 1
```

```
: 1277      1249 1 |  
: 1278      1250 1 |  
: 1279      1251 1 |  
: 1280      1252 1 |  
: 1281      P 1253 1 $STATE (ST_SPR_GO,  
: 1282      P 1254 1 (TPAS_EOS, ST_SPR_PMT_DAT),  
: 1283      P 1255 1 (TPAS_LAMBDA, ST_SPR_DAT)  
: 1284      1256 1 );  
: 1285      1257 1 |  
: 1286      1258 1 |  
: 1287      1259 1 |  
: 1288      1260 1 |  
: 1289      1261 1 |  
: 1290      P 1262 1 $STATE (ST_SPR_PMT_DAT,  
: 1291      1263 1 (TPAS_LAMBDA,, ACT$PRMPT,,, PMT$G_SPR_DAT));  
: 1292      1264 1 |  
: 1293      P 1265 1 $STATE (ST_SPR_DAT,  
: 1294      P 1266 1 ((ST_INF_TYPE1))  
: 1295      1267 1 );  
: 1296      1268 1 |  
: 1297      P 1269 1 $STATE (  
: 1298      P 1270 1 ('DTE', ST_SPR_DTE),  
: 1299      P 1271 1 ('GROUP', ST_SPR_GRP),  
: 1300      P 1272 1 ('KNOWN', ST_SPR_KNW),  
: 1301      P 1273 1 (TPAS_LAMBDA, ST_SPR_DOIT)  
: 1302      1274 1 );  
: 1303      1275 1 |  
: 1304      P 1276 1 $STATE (ST_SPR_KNW,  
: 1305      P 1277 1 ('DTE', ST_SPR_DOIT, ACT$SAVPRM, , PBK$G_SPR_KDT),  
: 1306      P 1278 1 ('GROUP', ST_SPR_DOIT, ACT$SAVPRM, , , PBK$G_SPR_KGR),  
: 1307      P 1279 1 (TPAS_LAMBDA, ST_SPR_DOIT)  
: 1308      1280 1 );  
: 1309      1281 1 |  
: 1310      1282 1 |  
: 1311      1283 1 |  
: 1312      1284 1 |  
: 1313      P 1285 1 |  
: 1314      P 1286 1 |  
: 1315      P 1287 1 |  
: 1316      P 1288 1 |  
: 1317      1289 1 |  
: 1318      1290 1 |  
: 1319      1291 1 |  
: 1320      1292 1 |  
: 1321      1293 1 |  
: 1322      1294 1 |  
: 1323      P 1295 1 |  
: 1324      P 1296 1 |  
: 1325      P 1297 1 |  
: 1326      P 1298 1 |  
: 1327      1299 1 |  
: 1328      1300 1 |  
: 1329      P 1301 1 $STATE (ST_SPR_DOIT,  
: 1330      P 1302 1 ((ST_INF_TYPE1))  
: 1331      1303 1 );  
: 1332      1304 1 |  
: 1333      P 1305 1 $STATE (,
```


NCPSTASHL
V04-000

Show/List Parse States and Data
Show / List Module

M 6
16-Sep-1984 01:30:43
14-Sep-1984 12:48:32

VAX-11 Bliss-32 V4.0-742
[NCP.SRC]NCPSTASHL.B32;1

Page 33
(29)

: 1334
: 1335

P 1306 1
1307 1

(TPAS_EOS, TPAS_EXIT, ACTSVRB_SHOLIS, , , SDB\$G_SPR)
);

NCP
V04

```

1337 1308 1 | SHOW MODULE X25-SERVER
1338 1309 1 |
1339 1310 1 |
1340 P 1311 1 $STATE (ST_SSE_GO,
1341 P 1312 1 (TPAS_EOS, ST_SSE_PMT_DAT),
1342 P 1313 1 (TPAS_LAMBDA, ST_SSE_DAT)
1343 1314 1 );
1344 1315 1 |
1345 1316 1 | Obtain the database type or prompt
1346 1317 1 |
1347 1318 1 |
1348 1319 1 |
1349 P 1320 1 $STATE (ST_SSE_PMT_DAT,
1350 1321 1 (TPAS_LAMBDA, ACT$PRMPT, , , PMT$G_SSE_DAT));
1351 1322 1 |
1352 P 1323 1 $STATE (ST_SSE_DAT,
1353 P 1324 1 ((ST_INF_TYPE1))
1354 1325 1 );
1355 1326 1 |
1356 P 1327 1 $STATE (
1357 P 1328 1 ('KNOWN', ST_SSE_KNW),
1358 P 1329 1 ('DESTINATION', ST_SSE_DST),
1359 P 1330 1 (TPAS_LAMBDA, ST_SSE_DOIT)
1360 1331 1 );
1361 1332 1 |
1362 P 1333 1 $STATE (ST_SSE_KNW,
1363 P 1334 1 ('DESTINATIONS', ST_SSE_DOIT, ACT$SAVPRM, , , PBK$G_SSE_KDS),
1364 P 1335 1 (TPAS_LAMBDA, ST_SSE_DOIT)
1365 1336 1 );
1366 1337 1 |
1367 1338 1 |
1368 1339 1 | Collect the DESTINATION name or prompt
1369 1340 1 |
1370 1341 1 |
1371 P 1342 1 COMMAND PROMPT
1372 P 1343 1 (SSE, DST, NCP$_INVVAL,
1373 P 1344 1
1374 P 1345 1 ((SE_DEST_NAME), ST_SSE_DOIT, ACT$SAVPRM, , , PBK$G_SSE_DST)
1375 1346 1 )
1376 1347 1 |
1377 P 1348 1 $STATE (ST_SSE_DOIT,
1378 P 1349 1 ((ST_INF_TYPE1))
1379 1350 1 );
1380 1351 1 |
1381 P 1352 1 $STATE (
1382 P 1353 1 (TPAS_EOS, TPAS_EXIT, ACT$VRB_SHOLIS, , , SDB$G_SSE)
1383 1354 1 );

```



```
: 1385      1355 1 |  
: 1386      1356 1 |  
: 1387      1357 1 |  
: 1388      1358 1 |  
: 1389      P 1359 1 $STATE (ST_STR_GO,  
: 1390      P 1360 1 ((ST_INF_TYPE1))  
: 1391      1361 1 );  
: 1392      1362 1 |  
: 1393      P 1363 1 $STATE (  
: 1394      P 1364 1 ('KNOWN', ST_STR_KNW),  
: 1395      P 1365 1 ('TRACEPOINT', ST_STR_TPT),  
: 1396      P 1366 1 (TPAS_LAMBDA, ST_STR_DOIT)  
: 1397      1367 1 );  
: 1398      1368 1 |  
: 1399      P 1369 1 $STATE (ST_STR_KNW,  
: 1400      P 1370 1 ('TRACEPOINTS', ST_STR_DOIT, ACT$SAVPRM, , , PBK$G_STR_KTP),  
: 1401      P 1371 1 (TPAS_LAMBDA, ST_STR_DOIT)  
: 1402      1372 1 );  
: 1403      1373 1 |  
: 1404      1374 1 |  
: 1405      1375 1 |  
: 1406      1376 1 |  
: 1407      P 1377 1 |  
: 1408      P 1378 1 |  
: 1409      P 1379 1 |  
: 1410      P 1380 1 |  
: 1411      1381 1 |  
: 1412      1382 1 |  
: 1413      1383 1 |  
: 1414      P 1384 1 $STATE (ST_STR_DOIT,  
: 1415      P 1385 1 ((ST_INF_TYPE1))  
: 1416      1386 1 );  
: 1417      1387 1 |  
: 1418      P 1388 1 $STATE (  
: 1419      P 1389 1 (TPAS_EOS, TPAS_EXIT, ACT$VRB_SHOLIS, , , SDB$G_STR)  
: 1420      1390 1 );
```

```
: 1422      1391 1 |  
: 1423      1392 1 |      SHOW MODULE X29-SERVER  
: 1424      1393 1 |  
: 1425      1394 1 |  
: 1426      P 1395 1 $STATE (ST_S9S_GO,  
: 1427      P 1396 1      (TPAS_EOS, ST_S9S_PMT_DAT),  
: 1428      P 1397 1      (TPAS_LAMBDA, ST_S9S_DAT)  
: 1429      1398 1      );  
: 1430      1399 1 |  
: 1431      1400 1 |  
: 1432      1401 1 |      Obtain the database type or prompt  
: 1433      1402 1 |  
: 1434      1403 1 |  
: 1435      P 1404 1 $STATE (ST_S9S_PMT_DAT,  
: 1436      1405 1      (TPAS_LAMBDA, ACT$PRMPT, PMT$G_S9S_DAT));  
: 1437      1406 1 |  
: 1438      P 1407 1 $STATE (ST_S9S_DAT,  
: 1439      P 1408 1      ((ST_INF_TYPE1))  
: 1440      1409 1      );  
: 1441      1410 1 |  
: 1442      P 1411 1 $STATE (  
: 1443      P 1412 1      ('KNOWN', ST_S9S_KNW),  
: 1444      P 1413 1      ('DESTINATION', ST_S9S_DST),  
: 1445      P 1414 1      (TPAS_LAMBDA, ST_S9S_DOIT)  
: 1446      1415 1      );  
: 1447      1416 1 |  
: 1448      P 1417 1 $STATE (ST_S9S_KNW,  
: 1449      P 1418 1      ('DESTINATIONS', ST_S9S_DOIT, ACT$SAVPRM, , PBK$G_SSE_KDS),  
: 1450      P 1419 1      (TPAS_LAMBDA, ST_S9S_DOIT)  
: 1451      1420 1      );  
: 1452      1421 1 |  
: 1453      1422 1 |  
: 1454      1423 1 |      Collect the DESTINATION name or prompt  
: 1455      1424 1 |  
: 1456      P 1425 1      COMMAND PROMPT  
: 1457      P 1426 1      (S9S, DST, NCP$_INVVAL,  
: 1458      P 1427 1      ((SE_DEST_NAME), ST_S9S_DOIT, ACT$SAVPRM, , PBK$G_SSE_DST)  
: 1459      P 1428 1      )  
: 1460      1429 1 |  
: 1461      1430 1 |  
: 1462      P 1431 1 $STATE (ST_S9S_DOIT,  
: 1463      P 1432 1      ((ST_INF_TYPE1))  
: 1464      1433 1      );  
: 1465      1434 1 |  
: 1466      P 1435 1 $STATE (  
: 1467      P 1436 1      (TPAS_EOS, TPAS_EXIT, ACT$VRB_SHOLIS, , SDB$G_S9S)  
: 1468      1437 1      );
```



```
: 1470      1438 1 %SBTTL 'Subexpressions'
: 1471      1439 1
: 1472      1440 1
: 1473      1441 1
: 1474      1442 1
: 1475      1443 1
: 1476      1444 1
: 1477      1445 1
: 1478      1446 1
: 1479      1447 1
: 1480      P 1448 1 $STATE (ST_INF TYPE1,
: 1481      P 1449 1      ( (SE_INFO_TYPES), ST_INF TYPE1),
: 1482      P 1450 1      ( (SE_INFO_TO), ST_INF TYPE1),
: 1483      P 1451 1      (TPAS_LAMBDA, TPAS_EXIT)
: 1484      1452 1      );
: 1485      1453 1
: 1486      1454 1
: 1487      1455 1
: 1488      1456 1
: 1489      1457 1
: 1490      P 1458 1 $STATE (ST_INF TYPE2,
: 1491      P 1459 1      ( (SE_INFO_TYPES), ST_INF TYPE2),
: 1492      P 1460 1      ( (SE_INFO_TO), ST_INF TYPE2),
: 1493      P 1461 1      (TPAS_LAMBDA, TPAS_EXIT)
: 1494      1462 1      );
: 1495      1463 1
: 1496      1464 1
: 1497      1465 1
: 1498      1466 1
: 1499      1467 1
: 1500      P 1468 1 $STATE (ST_INF TYPE3,
: 1501      P 1469 1      ( (SE_INFO_TYPES), ST_INF TYPE3),
: 1502      P 1470 1      ( (SE_INFO_TO), ST_INF TYPE3),
: 1503      P 1471 1      (TPAS_LAMBDA, TPAS_EXIT)
: 1504      1472 1      );
```

```
: 1506      1473 1
: 1507      1474 1
: 1508      1475 1
: 1509      1476 1
: 1510      1477 1
: 1511 P 1478 1 $STATE (SE INFO TYPES
: 1512 P 1479 1 ('CHARACTERISTICS', TPAS_EXIT, ACT$SAVPRM,
: 1513 P 1480 1 NMA$C_OPINF_CHA ^ $BITPOSITION(NMA$V_OPT_INF),
: 1514 P 1481 1 NCP$G_OPTION, PBK$G_SHL_INF),
: 1515 P 1482 1 ('STATUS', TPAS_EXIT, ACT$SAVPRM,
: 1516 P 1483 1 NMA$C_OPINF_STA ^ $BITPOSITION(NMA$V_OPT_INF),
: 1517 P 1484 1 NCP$G_OPTION, PBK$G_SHL_INF),
: 1518 P 1485 1 ('SUMMARY', TPAS_EXIT, ACT$SAVPRM,
: 1519 P 1486 1 NMA$C_OPINF_SUM ^ $BITPOSITION(NMA$V_OPT_INF),
: 1520 P 1487 1 NCP$G_OPTION, PBK$G_SHL_INF),
: 1521 P 1488 1 ('EVENTS', TPAS_EXIT, ACT$SAVPRM,
: 1522 P 1489 1 NMA$C_OPINF_EVE ^ $BITPOSITION(NMA$V_OPT_INF),
: 1523 P 1490 1 NCP$G_OPTION, PBK$G_SHL_INF),
: 1524 P 1491 1 ('COUNTERS', TPAS_EXIT, ACT$SAVPRM,
: 1525 P 1492 1 NMA$C_OPINF_COU ^ $BITPOSITION(NMA$V_OPT_INF),
: 1526 P 1493 1 NCP$G_OPTION, PBK$G_SHL_INF),
: 1527      1494 1 );
: 1528      1495 1
: 1529      1496 1
: 1530      1497 1
: 1531      1498 1
: 1532      1499 1
: 1533 P 1500 1 $STATE (SE INFO_TO,
: 1534 P 1501 1 ('TO'),
: 1535      1502 1 );
: 1536      1503 1
: 1537 P 1504 1 $STATE (,
: 1538 P 1505 1 ('(SE_FILE_ID), TPAS_EXIT, ACT$SAVPRM, , , PBK$G_INF_TO),
: 1539      1506 1 );
```



```

: 1541      1507 1 %SBTTL 'Define Subexpressions from Library'
: 1542      1508 1
: 1543      1509 1
: 1544      1510 1
: 1545      1511 1
: 1546      1512 1
: 1547      1513 1
: 1548      1514 1
: 1549      1515 1
: 1550      1516 1
: 1551      1517 1
: 1552      1518 1
: 1553      1519 1
: 1554      1520 1
: 1555      1521 1
: 1556      1522 1
: 1557      1523 1
: 1558      1524 1
: 1559      1525 1
: 1560      1526 1

Define subexpressions from library
Any additions must have a macro defined in module NCPLIBRY.

SEM_AREA_NUM      ! Node area number
SEM_FILE_ID       ! File id strings
SEM_LINE_ID       ! Line id strings
SEM_LOG_TYP       ! Logging entity type
SEM_NODE_ID       ! Node id strings
SEM_OBJECT_ID     ! Object name/number
SEM_CIRC_ID       ! Circuit name
SEM_LINK_ID       ! Link address
SEM_DTE_NUMBER    ! DTE number
SEM_GRP_NAME      ! Group name
SEM_DEST_NAME     ! Destination name
SEM_TRCPNT_NAME   ! Tracepoint name
SEM_NET_NAME      ! Network name

```

NCPSTASHL
V04-000

Show/List Parse States and Data
Object Listing of Parse Table

6 7
16-Sep-1984 01:30:43
14-Sep-1984 12:48:32

VAX-11 Bliss-32 V4.0-742
[NCP.SRC]NCPSTASHL.B32;1

Page 40
(36)

: 1562
: 1563
: 1564
: 1565

1527 1 %SBTTL 'Object Listing of Parse Table'
1528 1
1529 1 END
1530 0 ELUDOM

!End of module

0271

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY