

CCCCCCCCCCCC	DDDDDDDDDDDD	UUU	UUU
CCCCCCCCCCCC	DDDDDDDDDDDD	UUU	UUU
CCCCCCCCCCCC	DDDDDDDDDDDD	UUU	UUU
CCC	DDD	UUU	UUU
CCC	DDD	UUU	UUU
CCC	DDD	UUU	UUU
CCC	DDD	UUU	UUU
CCC	DDD	UUU	UUU
CCC	DDD	UUU	UUU
CCC	DDD	UUU	UUU
CCC	DDD	UUU	UUU
CCC	DDD	UUU	UUU
CCC	DDD	UUU	UUU
CCC	DDD	UUU	UUU
CCC	DDD	UUU	UUU
CCC	DDD	UUU	UUU
CCC	DDD	UUU	UUU
CCCCCCCCCCCC	DDDDDDDDDDDD	UUUUUUUUUUUUUUUU	UUUUUUUUUUUUUUUU
CCCCCCCCCCCC	DDDDDDDDDDDD	UUUUUUUUUUUUUUUU	UUUUUUUUUUUUUUUU
CCCCCCCCCCCC	DDDDDDDDDDDD	UUUUUUUUUUUUUUUU	UUUUUUUUUUUUUUUU

GGGGGGGG	EEEEEEEEEE	NN	NN	CCCCCCCC	000000	DDDDDDDD	EEEEEEEEEE	11
GGGGGGGG	EEEEEEEEEE	NN	NN	CCCCCCCC	000000	DDDDDDDD	EEEEEEEEEE	11
GG	EE	NN	NN	CC	00	DD	EE	1111
GG	EE	NN	NN	CC	00	DD	EE	1111
GG	EE	NNNN	NN	CC	00	DD	EE	11
GG	EE	NNNN	NN	CC	00	DD	EE	11
GG	EEEEEEEE	NN	NN	CC	00	DD	EEEEEEEE	11
GG	EEEEEEEE	NN	NN	CC	00	DD	EEEEEEEE	11
GG	EE	NN	NNNN	CC	00	DD	EE	11
GG	EE	NN	NNNN	CC	00	DD	EE	11
GG	EE	NN	NN	CC	00	DD	EE	11
GG	EE	NN	NN	CC	00	DD	EE	11
GGGGGG	EEEEEEEEEE	NN	NN	CCCCCCCC	000000	DDDDDDDD	EEEEEEEEEE	111111
GGGGGG	EEEEEEEEEE	NN	NN	CCCCCCCC	000000	DDDDDDDD	EEEEEEEEEE	111111

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
LLLLLLLLLL	IIIIII	SSSSSSSS
LLLLLLLLLL	IIIIII	SSSSSSSS


```
1 0001 0 MODULE gencode1 (IDENT='V04-000',
2 0002 0 ADDRESSING_MODE(EXTERNAL=GENERAL))
3 0003 1 = BEGIN
4 0004 1
5 0005 1 *****
6 0006 1 *
7 0007 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY *
8 0008 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS. *
9 0009 1 * ALL RIGHTS RESERVED. *
10 0010 1 *
11 0011 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED *
12 0012 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE *
13 0013 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER *
14 0014 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY *
15 0015 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY *
16 0016 1 * TRANSFERRED. *
17 0017 1 *
18 0018 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE *
19 0019 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT *
20 0020 1 * CORPORATION. *
21 0021 1 *
22 0022 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS *
23 0023 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL. *
24 0024 1 *
25 0025 1 *
26 0026 1 *****
27 0027 1
28 0028 1 ++
29 0029 1 Facility: Command Definition Utility, Table Generator Module 1
30 0030 1
31 0031 1 Abstract: This module is one of a few modules that is responsible
32 0032 1 for generating the blocks that make up the DCL tables.
33 0033 1 The blocks are generated by traversing the intermediate
34 0034 1 representation of the CLD file created by the parsing
35 0035 1 modules.
36 0036 1
37 0037 1 It is recommended that you read over the CLITABDEF.SDL file
38 0038 1 before reading this code.
39 0039 1
40 0040 1 Environment: Standard CDU environment.
41 0041 1
42 0042 1 Author: Paul C. Anagnostopoulos
43 0043 1 Creation: 8 December 1982
44 0044 1
45 0045 1 Modifications:
46 0046 1 --
47 0047 1
48 0048 1
49 0049 1 library 'sys$library:lib';
50 0050 1 require 'clitabdef';
51 0375 1 require 'cdureq';
```

53	0789	1	!	T A B L E O F C O N T E N T S
54	0790	1	!	-----
55	0791	1	!	-----
56	0792	1	!	forward routine
57	0793	1	!	cdu\$generate_table_blocks: novalue,
58	0794	1	!	cdu\$report_semantic_error: novalue,
59	0795	1	!	cdu\$remember_reference: novalue,
60	0796	1	!	cdu\$resolve_references: novalue;
61	0797	1	!	
62	0798	1	!	
63	0799	1	!	E X T E R N A L R E F E R E N C E S
64	0800	1	!	-----
65	0801	1	!	-----
66	0802	1	!	external routine
67	0803	1	!	cdu\$create_node,
68	0804	1	!	cdu\$generate_command,
69	0805	1	!	cdu\$report_listing_line,
70	0806	1	!	cdu\$lookup_child,
71	0807	1	!	cdu\$generate_type,
72	0808	1	!	lib\$signal;
73	0809	1	!	
74	0810	1	!	external
75	0811	1	!	cdu\$gl_cld_errors: long,
76	0812	1	!	cdu\$gl_root_node: ref node,
77	0813	1	!	cdu\$gl_table: pointer;

GENCODE1
V04-000

N 8
15-Sep-1984 23:36:54
14-Sep-1984 11:58:20

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[CCDU.SRC]GENCODE1.B32;1 Page 3 (3)

```
: 79      0814 1 !      O W N   S T O R A G E
: 80      0815 1 !      -----
: 81      0816 1 !
: 82      0817 1 ! The following item is the head of the linked list of resolution nodes.
: 83      0818 1
: 84      0819 1 own
: 85      0820 1      resolution_list: long;
```

```

: 87      0821 1 :      O V E R V I E W   O F   C O D E   G E N E R A T I O N
: 88      0822 1 :      -----
: 89      0823 1 :
: 90      0824 1 : CLI table blocks are generated by traversing the intermediate
: 91      0825 1 : representation tree built by the parsing routines. There is a generation
: 92      0826 1 : routine for each of the table block formats, which is responsible for
: 93      0827 1 : looking at the subtree representing the construct to be converted into a
: 94      0828 1 : block, pulling out the necessary information, and stashing it in the
: 95      0829 1 : table block.
: 96      0830 1 :
: 97      0831 1 : Each generation routine operates by first allocating space for the
: 98      0832 1 : largest possible table block. It then traverses the subtree representing
: 99      0833 1 : its construct, filling in the table block. If it encounters a node which
: 100     0834 1 : represents another construct, it calls that construct's generation
: 101     0835 1 : routine to do its thing. All the blocks are linked together on the fly
: 102     0836 1 : via Table-Relative Offsets (TRO).
: 103     0837 1 :
: 104     0838 1 : Note that when the generation process is complete, the table blocks are
: 105     0839 1 : spread all over memory. Before the table is written out, it must be
: 106     0840 1 : collected into one contiguous area. This is done in module TABLE.

```



```
108 0841 1 !++
109 0842 1 ! Description: This routine is responsible for driving the generation of
110 0843 1 ! table blocks for the CLD file that has just been parsed.
111 0844 1 ! It scan the children of the top-level node in the
112 0845 1 ! intermediate representation, looking for verb, syntax,
113 0846 1 ! and type definitions.
114 0847 1
115 0848 1 ! Parameters: None.
116 0849 1
117 0850 1 ! Returns: Nothing.
118 0851 1
119 0852 1 ! Notes:
120 0853 1 ! --
121 0854 1
122 0855 1 GLOBAL ROUTINE cdu$generate_table_blocks : novalue
123 0856 2 = BEGIN
124 0857 2
125 0858 2 local
126 0859 2 child: ref node;
127 0860 2
128 0861 2
129 0862 2 ! Clear the head of the resolution node linked list. We will link nodes
130 0863 2 ! onto this list as we generate blocks.
131 0864 2
132 0865 2 resolution_list = 0;
133 0866 2
134 0867 2 ! Simply scan the children of the root node, looking for definitions.
135 0868 2
136 P 0869 2 scan_children(cdu$gl_root_node,child,
137 P 0870 2
138 P 0871 2 ! Case on the type of child node.
139 P 0872 2
140 P 0873 2 case .child[node_w_type] from 0 to node_k_max_type of set
141 P 0874 2
142 P 0875 2 [node_k_ident,
143 P 0876 2 node_k_module]:
144 P 0877 2
145 P 0878 2 ! The above nodes can be ignored.
146 P 0879 2
147 P 0880 2 ;
148 P 0881 2
149 P 0882 2 [node_k_define_verb,
150 P 0883 2 node_k_define_syntax]:
151 P 0884 2
152 P 0885 2 ! Call a routine to generate all blocks for the verb
153 P 0886 2 ! or syntax change definition.
154 P 0887 2
155 P 0888 2 cdu$generate_command(.child);
156 P 0889 2
157 P 0890 2 [node_k_define_type]:
158 P 0891 2
159 P 0892 2 ! Call a routine to generate all blocks for the type
160 P 0893 2 ! definition.
161 P 0894 2
162 P 0895 2 cdu$generate_type(.child);
163 P 0896 2
164 P 0897 2 [inrange,
```

0855
0865
0904

[illegible]

Address	Disassembly	Comment
00000000G 00	8F DD 00086 4\$:	PUSHL #CDUS_INTINYNODE
	01 FB 0008C	CALLS #1, LIB\$SIGNAL
	14 11 00093	BRB 7\$
00000000G 00	52 DD 00095 5\$:	PUSHL CHILD
	01 FB 00097	CALLS #1, CDUS\$GENERATE_COMMAND
	09 11 0009E	BRB 7\$
00000000G 00	52 DD 000A0 6\$:	PUSHL CHILD
	01 FB 000A2	CALLS #1, CDUS\$GENERATE_TYPE
52 04	A2 D0 000A9 7\$:	MOVL 4(CHILD), CHILD
	FF61 31 000AD	BRW 1\$
0000V CF	00 FB 000B0 8\$:	CALLS #0, CDUS\$RESOLVE_REFERENCES
	04 000B5	RET

```
; Routine Size: 182 bytes,    Routine Base: $CODE$ + 0000
```

0910
0914

GENCODE1
V04-000

F 9
15-Sep-1984 23:36:54
14-Sep-1984 11:58:20

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[CDU.SRC]GENCODE1.B32;1 Page 8 (5)


```
: 183      0915 1  !++
: 184      0916 1  Description: This routine is called when a semantic error is encountered.
: 185      0917 1  It signals the error so that it will appear on the
: 186      0918 1  terminal. It also includes the error in the listing file,
: 187      0919 1  if any.
: 188      0920 1
: 189      0921 1  Parameters: Standard $PUTMSG argument list.
: 190      0922 1
: 191      0923 1  Returns: Nothing.
: 192      0924 1
: 193      0925 1  Notes: You may want to compare this to CDU$REPORT_SYNTAX_ERROR.
: 194      0926 1  --
: 195      0927 1
: 196      0928 1 GLOBAL ROUTINE cdu$report_semantic_error      : novalue
: 197      0929 2 = BEGIN
: 198      0930 2
: 199      0931 2 builtin
: 200      0932 2     argptr,
: 201      0933 2     callg;
: 202      0934 2
: 203      0935 2
: 204      0936 2 ! Signal the error.
: 205      0937 2
: 206      0938 2 callg(argptr(),lib$signal);
: 207      0939 2
: 208      0940 2 ! Include the error message in the listing file.
: 209      0941 2
: 210      0942 2 callg(argptr(),cdu$report_listing_line);
: 211      0943 2
: 212      0944 2 ! Keep track of the number of semantic errors.
: 213      0945 2
: 214      0946 2 increment(cdu$gl_cld_errors);
: 215      0947 2
: 216      0948 2 return;
: 217      0949 2
: 218      0950 1 END;
```

		0000 0000	.ENTRY	CDU\$REPORT_SEMANTIC_ERROR, Save nothing	: 0928
00000000G	00	6C FA 00002	CALLG	(AP), LIB\$SIGNAL	: 0938
00000000G	00	6C FA 00009	CALLG	(AP), CDU\$REPORT_LISTING_LINE	: 0942
	00000000G	00 D6 00010	INCL	CDU\$GL_CLD_ERRORS	: 0946
		04 00016	RET		: 0950

; Routine Size: 23 bytes, Routine Base: \$CODE\$ + 00B6

220	0951	1	!	
221	0952	1	!	
222	0953	1	!	
223	0954	1	!	DEFINITIONS IN A CLD FILE CAN MAKE REFERENCES TO OTHER DEFINITIONS IN
224	0955	1	!	THE FILE. THESE REFERENCES CANNOT BE RESOLVED AS THE CLD IS PARSED,
225	0956	1	!	BECAUSE DEFINITIONS DO NOT HAVE TO APPEAR BEFORE REFERENCES TO THEM.
226	0957	1	!	THEREFORE, THE REFERENCES MUST BE RESOLVED DURING CODE GENERATION.
227	0958	1	!	
228	0959	1	!	
229	0960	1	!	WHEN A REFERENCE IS ENCOUNTERED DURING CODE GENERATION, A REFERENCE
230	0961	1	!	RESOLUTION NODE IS CREATED. THIS NODE CONTAINS THE FOLLOWING INFORMATION:
231	0962	1	!	
232	0963	1	!	o This sister pointer is used to chain all of the resolution nodes
233	0964	1	!	on a list, so that we can process them quickly after code
234	0965	1	!	generation.
235	0966	1	!	
236	0967	1	!	o The child pointer is used to reference the top-level node of the
237	0968	1	!	definition being referenced. After code generation, this node
238	0969	1	!	will contain the TRO of the table block being referenced.
239	0970	1	!	
240	0971	1	!	o The code pointer is used to reference the longword which is to
241	0972	1	!	contain the reference. We can fill in this longword after code


```

: 243 0973 1 ++
: 244 0974 1 Description: This routine is called to remember a definition reference
: 245 0975 1 which must be resolved after code generation is completed.
: 246 0976 1 A resolution node is created and used to remember the
: 247 0977 1 information needed to resolve the reference later.
: 248 0978 1
: 249 0979 1 Parameters: referencor By reference, the longword to contain the
: 250 0980 1 reference to the table block containing
: 251 0981 1 the definition.
: 252 0982 1 definition By reference, the node representing the
: 253 0983 1 definition being referenced.
: 254 0984 1
: 255 0985 1 Returns: Nothing.
: 256 0986 1
: 257 0987 1 Notes:
: 258 0988 1 --
: 259 0989 1
: 260 0990 1 GLOBAL ROUTINE cdu$remember_reference(referencor: pointer,
: 261 0991 1 definition: ref node) : novalue
: 262 0992 2 = BEGIN
: 263 0993 2
: 264 0994 2 local
: 265 0995 2 resolution: ref node;
: 266 0996 2
: 267 0997 2
: 268 0998 2 ! Create a resolution node to remember the reference for later processing.
: 269 0999 2 ! Link the node on the front of the list of resolution nodes.
: 270 1000 2
: 271 1001 2 resolution = cdu$create_node(node_k_resolution);
: 272 1002 2 resolution[node_l_sister] = .resolution_list;
: 273 1003 2 resolution_list = .resolution;
: 274 1004 2
: 275 1005 2 ! Remember the referencing longword in the code pointer, and the referenced
: 276 1006 2 ! definition node in the child pointer.
: 277 1007 2
: 278 1008 2 resolution[node_l_code] = .referencor;
: 279 1009 2 resolution[node_l_child] = .definition;
: 280 1010 2
: 281 1011 2 return;
: 282 1012 2
: 283 1013 1 END;

```

			0000 00000	.ENTRY	CDUS\$REMEMBER_REFERENCE, Save nothing	: 0990
			2F DD 00002	PUSHL	#47	: 1001
00000000G	00		01 FB 00004	CALLS	#1, CDUS\$CREATE NODE	
04 AO	0000'		CF D0 0000B	MOVL	RESOLUTION_LIST, 4(RESOLUTION)	: 1002
0000'	CF		50 D0 00011	MOVL	RESOLUTION, RESOLUTION_LIST	: 1003
0C AO	04		AC D0 00016	MOVL	REFERENCOR, 12(RESOLUTION)	: 1008
08 AO	08		AC D0 0001B	MOVL	DEFINITION, 8(RESOLUTION)	: 1009
			04 00020	RET		: 1013

; Routine Size: 33 bytes, Routine Base: \$CODE\$ + 00CD

GENCODE1
V04-000

J 9
15-Sep-1984 23:36:54
14-Sep-1984 11:58:20

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[CDU.SRC]GENCODE1.B32;1 Page 12
(8)


```
285 1014 1 !++
286 1015 1 ! Description: This routine is called after code generation is completed.
287 1016 1 ! It finishes up the task of resolving references to
288 1017 1 ! definitions by scanning the list of resolution nodes and
289 1018 1 ! storing the final TRO of the referenced block into the
290 1019 1 ! referencing longword.
291 1020 1
292 1021 1 ! Parameters: None.
293 1022 1
294 1023 1 ! Returns: Nothing.
295 1024 1
296 1025 1 ! Notes:
297 1026 1 --
298 1027 1
299 1028 1 GLOBAL ROUTINE cdu$resolve_references : novalue
300 1029 2 = BEGIN
301 1030 2
302 1031 2 local
303 1032 2 resolution: ref node,
304 1033 2 definition: ref node,
305 1034 2 referencor: pointer;
306 1035 2
307 1036 2
308 1037 2 ! Scan all of the resolution nodes that were created as references were
309 1038 2 ! discovered during code generation.
310 1039 2
311 1040 2 resolution = .resolution_list;
312 1041 3 while .resolution neqa 0 do (
313 1042 3
314 1043 3 ! The child pointer references a node representing the definition
315 1044 3 ! being referenced. That node now contains the TRO of the
316 1045 3 ! definition block.
317 1046 3
318 1047 3 definition = .resolution[node_l_child];
319 1048 3
320 1049 3 ! The code pointer points at a longword in some table block
321 1050 3 ! which is to receive the TRO of the referenced definition block.
322 1051 3 ! Store the TRO in the longword.
323 1052 3
324 1053 3 referencor = .resolution[node_l_code];
325 1054 3 referencor[0,0,32,0] = .definition[node_l_code];
326 1055 3
327 1056 3 ! Go on to the next resolution node.
328 1057 3
329 1058 3 resolution = .resolution[node_l_sister];
330 1059 2 );
331 1060 2
332 1061 2 return;
333 1062 2
334 1063 1 END;
```

```
50 0000' 0004 00000 .ENTRY CDUS$RESOLVE_REFERENCES, Save R2
CF DO 00002 MOVL RESOLUTION_LIST, RESOLUTION
```

```
: 1028
: 1040
```

GENCODE1
V04-000

L 9
15-Sep-1984 23:36:54
14-Sep-1984 11:58:20

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[CDU.SRC]GENCODE1.B32;1
Page 14
(9)

51	08	0E	13	00007	1\$:	BEQL	2\$	:	1041
62	0C	A0	7D	00009		MOVQ	8(RESOLUTION), DEFINITION	:	1047
50	04	A1	D0	0000D		MOVL	12(DEFINITION), (REFERENCOR)	:	1054
		A0	D0	00011		MOVL	4(RESOLUTION), RESOLUTION	:	1058
		F0	11	00015		BRB	1\$	:	1041
		04	00017	2\$:		RET		:	1063

; Routine Size: 24 bytes, Routine Base: \$CODE\$ + 00EE

: 335 1064 1
: 336 1065 1 END
: 337 1066 0 ELUDOM

.EXTRN LIB\$SIGNAL

PSECT SUMMARY

Name	Bytes	Attributes
\$OWNS	4	NOVEC, WRT, RD, NOEXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
\$CODE\$	262	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	4	0	1000	00:01.8

COMMAND QUALIFIERS

; BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS\$:GENCODE1/OBJ=OBJ\$:GENCODE1 MSRC\$:GENCODE1/UPDATE=(ENH\$:GENCODE1)

; Size: 262 code + 4 data bytes
; Run Time: 00:11.2
; Elapsed Time: 00:25.8
; Lines/CPU Min: 5690
; Lexemes/CPU-Min: 22339
; Memory Used: 111 pages
; Compilation Complete

0043

AH-BT13A-SE
 VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY