

RRRRRRRR	MM	MM	333333	XX	XX	AAAAAA	LL	LL	000000	
RRRRRRRR	MM	MM	333333	XX	XX	AAAAAA	LL	LL	000000	
RR	RR	MMMM	33	XX	XX	AA	AA	LL	00	00
RR	RR	MMMM	33	XX	XX	AA	AA	LL	00	00
RR	RR	MM	33	XX	XX	AA	AA	LL	00	00
RR	RR	MM	33	XX	XX	AA	AA	LL	00	00
RRRRRRRR	MM	MM	33	XX	XX	AA	AA	LL	00	00
RRRRRRRR	MM	MM	33	XX	XX	AA	AA	LL	00	00
RR	RR	MM	33	XX	XX	AAAAAAAAAA	LL	LL	00	00
RR	RR	MM	33	XX	XX	AAAAAAAAAA	LL	LL	00	00
RR	RR	MM	33	XX	XX	AA	AA	LL	00	00
RR	RR	MM	33	XX	XX	AA	AA	LL	00	00
RR	RR	MM	33	XX	XX	AA	AA	LL	00	00
RR	RR	MM	333333	XX	XX	AA	AA	LLLLLLLLLL	000000	000000
RR	RR	MM	333333	XX	XX	AA	AA	LLLLLLLLLL	000000	000000

```

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

```



```
1 0001 0 MODULE RM3XALLO (LANGUAGE (BLISS32) ,
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 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
9 0009 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
10 0010 1 * ALL RIGHTS RESERVED.
11 0011 1 *
12 0012 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
13 0013 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
14 0014 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
15 0015 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
16 0016 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
17 0017 1 * TRANSFERRED.
18 0018 1 *
19 0019 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
20 0020 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
21 0021 1 * CORPORATION.
22 0022 1 *
23 0023 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
24 0024 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
25 0025 1 *
26 0026 1 *
27 0027 1 *****
28 0028 1
29 0029 1 ++
30 0030 1
31 0031 1 FACILITY: RMS32 index sequential file organization
32 0032 1
33 0033 1 ABSTRACT:
34 0034 1 This routine fills in the area definitions from the XAB
35 0035 1
36 0036 1
37 0037 1 ENVIRONMENT:
38 0038 1
39 0039 1 VAX/VMS OPERATING SYSTEM
40 0040 1
41 0041 1 --
42 0042 1
43 0043 1
44 0044 1 AUTHOR: D. M. BOUSQUET
45 0045 1 CREATION DATE: 10-AUG-78 10:33
46 0046 1
47 0047 1
48 0048 1 MODIFIED BY
49 0049 1
50 0050 1 V03-012 SHZ0001 Stephen H. Zalewski 16-Apr-1984
51 0051 1 Do not attempt to read the area_vbn if this is a foreign
52 0052 1 device. This is needed to make copying of an index file
53 0053 1 to a foreign device work.
54 0054 1
55 0055 1 V03-011 DAS0001 David Solomon 25-Mar-1984
56 0056 1 Fix broken branches.
57 0057 1
```



```

58 0058 1 | V03-010 MCN0002 Maria del C. Nasr 31-Mar-1983
59 0059 1 | Reorganize linkages.
60 0060 1 |
61 0061 1 | V03-009 MCN0001 Maria del C. Nasr 08-Nov-1982
62 0062 1 | Return new field in area descriptor which stores
63 0063 1 | the area's total allocation: AREASL_TOTAL_ALLOC.
64 0064 1 | Also, now that the information in the area descriptor is
65 0065 1 | consistent, we can return the ALN value.
66 0066 1 |
67 0067 1 | V03-008 KBT0238 Keith B. Thompson 23-Aug-1982
68 0068 1 | Reorganize psects
69 0069 1 |
70 0070 1 | V02-0J7 KBT0001 K B Thompson 30-Dec-1981
71 0071 1 | Fill in the ALQ field with SOMETHING.
72 0072 1 |
73 0073 1 | V02-006 CDS0002 C Saether 9-Aug-1981
74 0074 1 | Use alternate linkage for RELEASE.
75 0075 1 |
76 0076 1 | V02-005 REFORMAT D M WALP 24-JUL-1980
77 0077 1 |
78 0078 1 | V02-004 CDS0001 C D SAETHER 11-MAR-1980
79 0079 1 | Return OK_NOP if area vbn zero (prologue not read)
80 0080 1 |
81 0081 1 | *****
82 0082 1 |
83 0083 1 | LIBRARY 'RMSLIB:RMS';
84 0084 1 |
85 0085 1 | REQUIRE 'RMSSRC:RMSIDXDEF';
86 0150 1 |
87 0151 1 | ! define default psects for code
88 0152 1 |
89 0153 1 | PSECT
90 0154 1 | CODE = RMSRMS3(PSECT_ATTR),
91 0155 1 | PLIT = RMSRMS3(PSECT_ATTR);
92 0156 1 |
93 0157 1 | ! Linkages
94 0158 1 |
95 0159 1 | LINKAGE
96 0160 1 | L_CHKSUM,
97 0161 1 | L_RELEASE_FAB,
98 0162 1 | L_CACHE,
99 0163 1 | L_FABREG,
100 0164 1 | L_LINK_7_10_11;
101 0165 1 |
102 0166 1 | ! External Routines
103 0167 1 |
104 0168 1 | EXTERNAL ROUTINE
105 0169 1 | RMSCHKSUM : RL$CHKSUM,
106 0170 1 | RMSRELEASE : RL$RELEASE FAB ADDRESSING MODE( LONG RELATIVE ),
107 0171 1 | RMSCACHE : RL$CACHE ADDRESSING MODE( LONG RELATIVE ),
108 0172 1 | RMSRND_DEV : RL$LINK_7_10_11 ADDRESSING_MODE( GENERAL );
```

```

110 0173 1 GLOBAL ROUTINE RM$XALLO3 (XAB) : RL$FABREG =
111 0174 1
112 0175 1 ++
113 0176 1
114 0177 1 FUNCTIONAL DESCRIPTION:
115 0178 1
116 0179 1 This routine checks to make sure that the Area Descriptors
117 0180 1 are never greater than the maximum number of descriptors
118 0181 1 defined for the file, (found in the IFAB).
119 0182 1
120 0183 1 It then fills in the ALLOCATION XAB from the area descriptor
121 0184 1
122 0185 1 CALLING SEQUENCE:
123 0186 1 RM$XALLO3(XAB)
124 0187 1
125 0188 1 INPUT PARAMETERS:
126 0189 1 XAB - Pointer to the allocation XAB we are processing
127 0190 1
128 0191 1 IMPLICIT INPUTS:
129 0192 1 $XAB
130 0193 1 [XAB$B_AID] - Area ID
131 0194 1
132 0195 1 $AREADEF
133 0196 1 [AREASB_AOP] - Area id for this descriptor
134 0197 1 [AREASB_ALN] - Alignment options
135 0198 1 [AREASL_CNBLK] - Number of blocks in current extent
136 0199 1 [AREASL_NXBLK] - Number of blocks in next extent
137 0200 1 [AREASW_DEQ] - Extend allocation alignment
138 0201 1 [AREASB_ARBKTSZ] - Bucket size for area
139 0202 1 [AREASW_VOLUME] - Volume number
140 0203 1 [AREASC_BLN] - Block length for area desc., 64 bytes
141 0204 1
142 0205 1 $STRUCT IFAB
143 0206 1 [IFBSB_AMAX] - Maximum number of area descriptors in file
144 0207 1 [IFBSB_AVBN] - Start vbn of first area descriptor
145 0208 1 [IFBSB_ORGCASE] - File organization
146 0209 1 [IFBSC_IDX] - Constant for index files
147 0210 1
148 0211 1
149 0212 1
150 0213 1 OUTPUT PARAMETERS:
151 0214 1 XAB - Untouched by this routine
152 0215 1
153 0216 1 IMPLICIT OUTPUTS:
154 0217 1 $XAB
155 0218 1 [XAB$B_AOP] - Alignment options
156 0219 1 [XAB$B_ALN] - Alignment
157 0220 1 [XAB$S_ALQ] - Allocation
158 0221 1 [XAB$W_DEQ] - Default extend quantity
159 0222 1 [XAB$B_BKZ] - Bucket size in blocks
160 0223 1 [XAB$W_VOL] - Volume number
161 0224 1
162 0225 1 ROUTINE VALUE:
163 0226 1 RMSERR
164 0227 1 (AID) - Invalid area id
165 0228 1 (SUC) - Success
166 0229 1
```



```
167 0230 1 |
168 0231 1 | SIDE EFFECTS:
169 0232 1 |     NONE
170 0233 1 |
171 0234 1 | --
172 0235 1 |
173 0236 2 | BEGIN
174 0237 2 |
175 0238 2 | EXTERNAL REGISTER
176 0239 2 |     COMMON_FAB_STR;
177 0240 2 |
178 0241 2 | GLOBAL REGISTER
179 0242 2 |     R_IDX_DFN;
180 0243 2 |
181 0244 2 | MAP
182 0245 2 |     XAB      : REF BBLOCK;
183 0246 2 |
184 0247 2 | LOCAL
185 0248 2 |     AREA_VBN,
186 0249 2 |     AREA_DESC      : REF BBLOCK;
187 0250 2 |
188 0251 2 | ! Just to make sure this is an indexed file
189 0252 2 | !
190 0253 2 | IF .IFAB[IFB$B_ORGCASE] EQL IFB$C_IDX
191 0254 2 | THEN
192 0255 2 |     BEGIN
193 0256 2 |
194 0257 2 |         ! Before we do anything let's check validity of the AID, all we really
195 0258 2 |         ! care about is that it isn't larger than the largest defined for this
196 0259 2 |         ! field. Don't care if areas are contiguous or if AID's are in
197 0260 2 |         ! ascending order.
198 0261 2 |
199 0262 2 |
200 0263 2 | IF .XAB[XAB$B_AID] GTRU .IFAB[IFB$B_AMAX]
201 0264 2 | THEN
202 0265 2 |     RETURN RMSERR(AID);
203 0266 2 |
204 0267 2 | ! Now to compute the area_vbn and the descriptor to work on If AVBN is
205 0268 2 | ! zero, the prologue wasn't read on OPEN (block i/o)
206 0269 2 | !
207 0270 2 |
208 0271 2 | IF (AREA_VBN = .IFAB[IFB$B_AVBN]) EQL 0
209 0272 2 | OR NOT RMSRND_DEV()
210 0273 2 | THEN
211 0274 2 |     RETURN RMSSUC(OK_NOP);
212 0275 2 |
213 0276 2 | AREA_VBN = .AREA_VBN + .XAB[XAB$B_AID]/8;
214 0277 2 | AREA_DESC = .XAB[XAB$B_AID] AND %X'00000007';
215 0278 2 |
216 0279 2 | ! Now to read in the area_vbn
217 0280 2 | !
218 0281 4 | BEGIN
219 0282 4 |
220 0283 4 | GLOBAL REGISTER
221 0284 4 |     COMMON_IO_STR;
222 0285 4 |
223 0286 4 | LOCAL
```

```

: 224      0287 4      STATUS;
: 225      0288 4
: 226      0289 4      STATUS = RM$CACHE(.AREA_VBN, 512, 0);
: 227      0290 4
: 228      0291 4      ! If error then return with error code in status
: 229      0292 4      !
: 230      0293 4
: 231      0294 4      IF NOT .STATUS
: 232      0295 4      THEN
: 233      0296 4          RETURN .STATUS;
: 234      0297 4
: 235      0298 4      ! Now to check bucket and release
: 236      0299 4
: 237      0300 4      RETURN_ON_ERROR ( RM$CHKSUM(), RM$RELEASE(0) );
: 238      0301 4
: 239      0302 4      ! Now to calculate the offset into the vbn
: 240      0303 4
: 241      0304 4      AREA_DESC = .AREA_DESC*AREA$C_BLN + .BKT_ADDR;
: 242      0305 4
: 243      0306 4      ! Now to fill in the various fields
: 244      0307 4
: 245      0308 4      XAB[XAB$B_AOP] = .AREA_DESC[AREA$B_AOP];
: 246      0309 4      XAB[XAB$B_ALN] = .AREA_DESC[AREA$B_ALN];
: 247      0310 4      XAB[XAB$L_ALQ] = .AREA_DESC[AREA$L_TOTAL_ALLOC];
: 248      0311 4      XAB[XAB$W_DEQ] = .AREA_DESC[AREA$W_DEQ];
: 249      0312 4      XAB[XAB$B_BKZ] = .AREA_DESC[AREA$B_ARBKTSZ];
: 250      0313 4      XAB[XAB$W_VOL] = .AREA_DESC[AREA$W_VOLUME];
: 251      0314 4      RM$RELEASE(0);
: 252      0315 3      END;
: 253      0316 3      ! end of GLOBAL REGISTER and STATUS def
: 254      0317 2      END;
: 255      0318 2
: 256      0319 2      ! Now to return the value of the routine if all went well
: 257      0320 2
: 258      0321 2      RETURN RM$SUC(SUC);
: 259      0322 2
: 260      0323 1      END;

```

```

.TITLE RM3XALLO
.IDENT \V04-000\

.EXTRN RM$CHKSUM, RM$RELEASE
.EXTRN RM$CACHE, RM$RND_DEV

.PSECT RM$RMS3,NOWRT, GBL, PIC,2

```

		00FC	8F	BB 00000	RM\$XALLO3::		
					PUSHR	#^M<R2,R3,R4,R5,R6,R7>	: 0173
	5E		04	C2 00004	SUBL2	#4, SP	
	02	23	AA	91 00007	CMPB	35(IFAB), #2	: 0253
			03	13 0000B	BEQL	1\$	
			0091	31 0000D	BRW	7\$	
	56	20	AE	D0 00010	MOVL	XAB, R6	: 0263
00B1	CA	17	A6	91 00014	CMPB	23(R6), 177(IFAB)	
			07	1B 0001A	BLEQU	2\$	
	50	83F4	8F	3C 0001C	MOVZWL	#33780, R0	: 0265

57	17	A6	6E	00B0	4E	11	00021	BRB	5\$			
					CA	9A	00023	MOVZBL	176(IFAB), AREA_VBN	:	0271	
					09	13	00028	BEQL	3\$	:		
				00000000G	00	16	0002A	JSB	RM\$RND_DEV	:	0272	
			07		50	E8	00030	BLBS	R0, 4\$	:		
			50	8059	8F	3C	00033	MOVZWL	#32857, R0	:	0274	
					6A	11	00038	BRB	8\$	:		
			50	17	A6	9A	0003A	MOVZBL	23(R6), R0	:	0276	
			50		08	C6	0003E	DIVL2	#8, R0	:		
			6E		50	C0	00041	ADDL2	R0, AREA_VBN	:		
			03		00	EF	00044	EXTZV	#0, #3, 23(R6), AREA_DESC	:	0277	
					53	D4	0004A	CLRL	R3	:	0289	
			52	0200	8F	3C	0004C	MOVZWL	#512, R2	:		
			51		6E	D0	00051	MOVL	AREA_VBN, R1	:		
				00000000G	EF	16	00054	JSB	RM\$CACHE	:		
			47		50	E9	0005A	BLBC	STATUS, 8\$	:	0294	
					0000G	30	0005D	BSBW	RM\$CHKSUM	:	0300	
			6E		50	D0	00060	MOVL	R0, STATUS	:		
			0D		6E	E8	00063	BLBS	STATUS, 6\$	:		
					53	D4	00066	CLRL	R3	:		
				00000000G	EF	16	00068	JSB	RM\$RELEASE	:		
			50		6E	D0	0006E	MOVL	STATUS, R0	:		
					31	11	00071	BRB	8\$	:		
		50	57		06	78	00073	ASHL	#6, AREA_DESC, R0	:	0304	
		57			55	C1	00077	ADDL3	BKT_ADDR, R0, AREA_DESC	:		
			08	A6	07	A7	90	0007B	MOVB	7(AREA_DESC), 8(R6)	:	0308
			09	A6	06	A7	90	00080	MOVB	6(AREA_DESC), 9(R6)	:	0309
			10	A6	32	A7	D0	00085	MOVL	50(AREA_DESC), 16(R6)	:	0310
			14	A6	24	A7	B0	0008A	MOVW	36(AREA_DESC), 20(R6)	:	0311
			16	A6	03	A7	90	0008F	MOVB	3(AREA_DESC), 22(R6)	:	0312
			0A	A6	04	A7	B0	00094	MOVW	4(AREA_DESC), 10(R6)	:	0313
					53	D4	00099	CLRL	R3	:	0314	
				00000000G	EF	16	0009B	JSB	RM\$RELEASE	:		
			50		01	D0	000A1	MOVL	#1, R0	:	0321	
			5E		04	C0	000A4	ADDL2	#4, SP	:	0323	
				00FC	8F	BA	000A7	POPR	#^M<R2,R3,R4,R5,R6,R7>	:		
					05	000AB	RSB			:		

; Routine Size: 172 bytes, Routine Base: RM\$RMS3 + 0000

:	261	0324	1
:	262	0325	1 END
:	263	0326	1
:	264	0327	0 ELUDOM

PSECT SUMMARY

Name	Bytes	Attributes
RM\$RMS3	172	NOVEC, NOWRT, RD, EXE, NOSHR, GBL, REL, CON, PIC, ALIGN(2)

Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
_\$255\$DUA28:[RMS.OBJ]RMS.L32;1	3109	48	1	154	00:00.3

COMMAND QUALIFIERS

```

:      BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:RM3XALLO/OBJ=OBJ$:RM3XALLO MSRC$:RM3XALLO/UPDATE=(ENH$:RM3XALLO)
:
: Size:      172 code + 0 data bytes
: Run Time:   00:05.4
: Elapsed Time: 00:17.1
: Lines/CPU Min: 3660
: Lexemes/CPU-Min: 15100
: Memory Used: 78 pages
: Compilation Complete

```


0329 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

