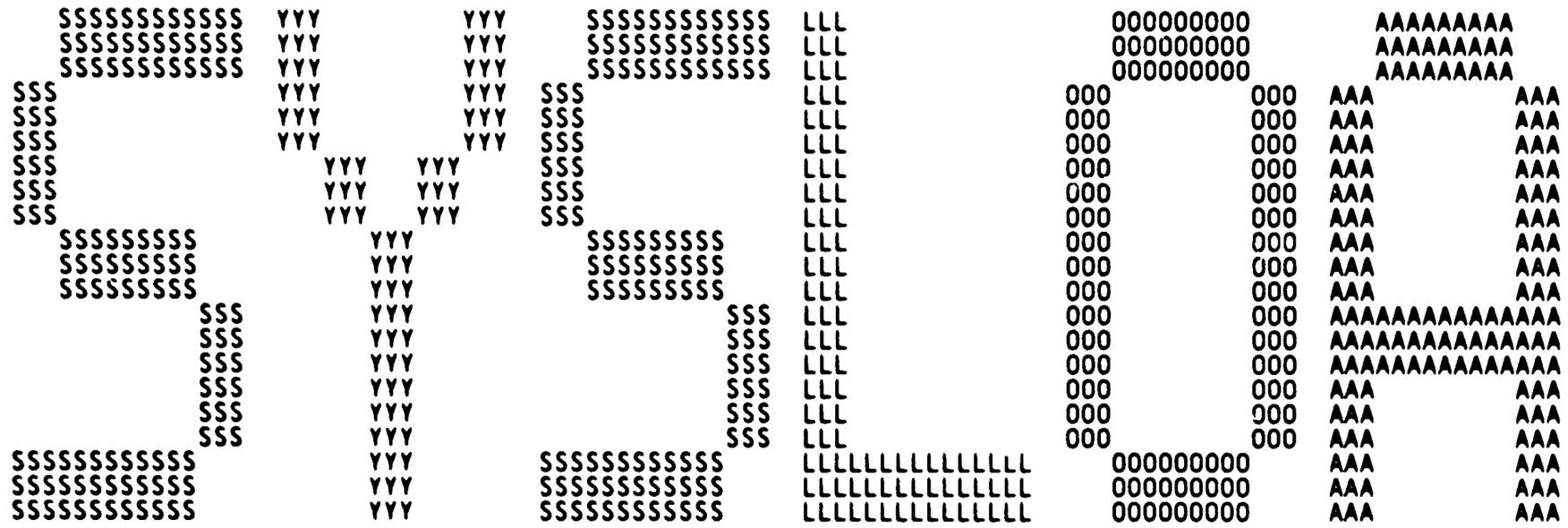
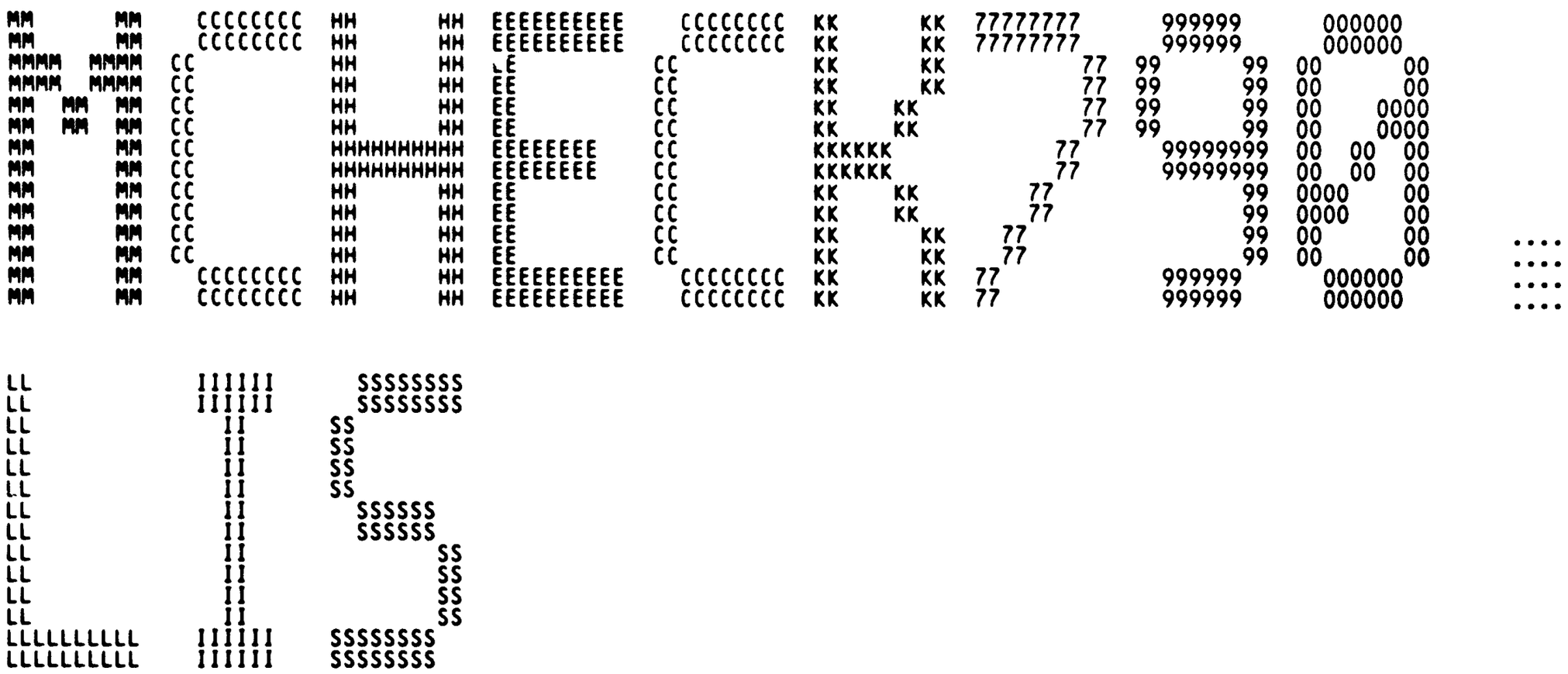

SSSSSSSSSSS YYY YYY SSSSSSSSSSS LLL 000000000 AAAAAAAAA
SSSSSSSSSSS YYY YYY SSSSSSSSSSS LLL 000000000 AAAAAAAAA
SSSSSSSSSSS YYY YYY SSSSSSSSSSS LLL 000000000 AAAAAAAAA
SSS YYY YYY SSS LLL 000 000 AAA AAA
SSS YYY YYY SSS LLL 000 000 AAA AAA
SSS YYY YYY SSS LLL 000 000 AAA AAA
SSS YYY YYY SSS LLL 000 000 AAA AAA
SSS YYY YYY SSS LLL 000 000 AAA AAA
SSS YYY YYY SSS LLL 000 000 AAA AAA
SSSSSSSSS YYY SSSSSSSSS LLL 000 000 AAA AAA
SSSSSSSSS YYY SSSSSSSSS LLL 000 000 AAA AAA
SSSSSSSSS YYY SSSSSSSSS LLL 000 000 AAA AAA
SSS YYY SSS LLL 000 000 AAAAAAAAAAAAAAA
SSS YYY SSS LLL 000 000 AAAAAAAAAAAAAAA
SSS YYY SSS LLL 000 000 AAAAAAAAAAAAAAA
SSS YYY SSS LLL 000 000 AAA AAA
SSS YYY SSS LLL 000 000 AAA AAA
SSS YYY SSS LLL 000 000 AAA AAA
SSSSSSSSSSS YYY SSSSSSSSSSS LLLLLLLLLLLLLLL 000000000 AAA AAA
SSSSSSSSSSS YYY SSSSSSSSSSS LLLLLLLLLLLLLLL 000000000 AAA AAA
SSSSSSSSSSS YYY SSSSSSSSSSS LLLLLLLLLLLLLLL 000000000 AAA AAA


MM MM CCCCCCCC HH HH EEEEEEEEEE CCCCCCCC KK KK 77777777 999999 000000
MM MM CCCCCCCC HH HH EEEEEEEEEE CCCCCCCC KK KK 77777777 999999 000000
MMMM MMMM CC HH HH EE CC KK KK 77 99 99 00 00
MMMM MMMM CC HH HH EE CC KK KK 77 99 99 00 00
MM MM MM CC HH HH EE CC KK KK 77 99 99 00 0000
MM MM MM CC HH HH EE CC KK KK 77 99 99 00 0000
MM MM CC HHHHHHHHHH EEEEEEEE CC KKKKK 77 99999999 00 00 00
MM MM CC HHHHHHHHHH EEEEEEEE CC KKKKK 77 99999999 00 00 00
MM MM CC HH HH EE CC KK KK 77 99 0000 00
MM MM CC HH HH EE CC KK KK 77 99 0000 00
MM MM CC HH HH EE CC KK KK 77 99 00 00
MM MM CC HH HH EE CC KK KK 77 99 00 00
MM MM CCCCCCCC HH HH EEEEEEEEEE CCCCCCCC KK KK 77 999999 000000
MM MM CCCCCCCC HH HH EEEEEEEEEE CCCCCCCC KK KK 77 999999 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
LLLLLLLLLL IIIIII SSSSSSSS
LLLLLLLLLL IIIIII SSSSSSSS

MCH
V04

Table of contents

```
0000     1          .TITLE  MCHECK790 -- VENUS MACHINE CHECK
0000     2          .IDENT  'V04-004'
0000     3
0000     4 ;
0000     5 ;*****************************************************************************
0000     6 ;*                                                                           *
0000     7 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
0000     8 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0000     9 ;*  ALL RIGHTS RESERVED.                                                     *
0000    10 ;*                                                                           *
0000    11 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
0000    12 ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000    13 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
0000    14 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
0000    15 ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
0000    16 ;*  TRANSFERRED.                                                             *
0000    17 ;*                                                                           *
0000    18 ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
0000    19 ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000    20 ;*  CORPORATION.                                                             *
0000    21 ;*                                                                           *
0000    22 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
0000    23 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
0000    24 ;*                                                                           *
0000    25 ;*                                                                           *
0000    26 ;*****************************************************************************
0000    27 ;
0000    28
0000    29 ;++
0000    30 ; FACILITY: SYSLOA790 - loadable CPU-dependent code
0000    31 ;
0000    32 ; ABSTRACT:
0000    33 ;       This module contains routines to handle VAX 11/790 specific
0000    34 ;       machine check errors.
0000    35 ;
0000    36 ; ENVIRONMENT:
0000    37 ;       IPL = 31        Mode = KERNEL
0000    38 ;
0000    39 ; AUTHOR:  Wayne Cardoza        CREATION DATE: 2-Oct-1982
0000    40 ;
0000    41 ; MODIFIED BY:
0000    42 ;
0000    43 ;       V04-004 WMC0020         Wayne Cardoza           13-Sep-1984
0000    44 ;               Make sure we never run with CRD interrupts disabled.
0000    45 ;
0000    46 ;       V04-003 WMC0019         Wayne Cardoza           12-Sep-1984
0000    47 ;               Fix position accounting in CRD log.
0000    48 ;
0000    49 ;       V04-002 TCM0002         Trudy C. Matthews       11-Sep-1984
0000    50 ;               Make CON$KEEPALIVE's reference to ERL$RELEASEMB absolute.
0000    51 ;
0000    52 ;       V04-001 WMC0018         Wayne Cardoza           05-Sep-1984
0000    53 ;               Add MSTAT2 to the CRD log.
0000    54 ;
0000    55 ;       V03-023 WMC0017         Wayne Cardoza           23-Aug-1984
0000    56 ;               Fix a bad index mode.
0000    57 ;
```

```
0000     58 ;       V03-022 WMC0016         Wayne Cardoza           14-Aug-1984
0000     59 ;               CP-IO BUF doesn't always have a cycle type.
0000     60 ;
0000     61 ;       V03-021 WMC0015         Wayne Cardoza           06-Aug-1984
0000     62 ;               And more changes.
0000     63 ;
0000     64 ;       V03-020 TCM0001         Trudy C. Matthews       31-Jul-1984
0000     65 ;               Add routine CON$KEEPALIVE, which is called periodically to
0000     66 ;               determine if the VENUS console software is still functioning.
0000     67 ;               Also, change the CRD interrupt vector from ^X64 in the SCB to
0000     68 ;               ^X54.
0000     69 ;
0000     70 ;       V03-019 WMC0014         Wayne Cardoza           23-Jul-1984
0000     71 ;               Still more spec changes.
0000     72 ;
0000     73 ;       V03-018 WMC0013         Wayne Cardoza           12-Jul-1984
0000     74 ;               Many spec changes.
0000     75 ;
0000     76 ;       V03-017 WMC0012         Wayne Cardoza           15-May-1984
0000     77 ;               Space always reserved for SBIA error word.
0000     78 ;               Add PC-PSL to log message.
0000     79 ;
0000     80 ;       V03-016 WMC0011         Wayne Cardoza           30-Apr-1984
0000     81 ;               Add ADPDEF.
0000     82 ;
0000     83 ;       V03-015 WMC0010         Wayne Cardoza           16-Apr-1984
0000     84 ;               Ignore CP timeouts on BRVVR.
0000     85 ;
0000     86 ;       V03-014 RLRSBICONF      Robert L. Rappaport     22-Mar-1984
0000     87 ;               Test MMG$GL_SBICONF array elements for valid system
0000     88 ;               virtual address (high bit set) before using.  Also
0000     89 ;               correct bug introduced by CONFREGL fix.
0000     90 ;
0000     91 ;       V03-013 WMC0009         Wayne Cardoza           10-Mar-1984
0000     92 ;               Fix SBIA logging bugs.
0000     93 ;
0000     94 ;       V03-012 KPL0100         Peter Lieberwirth       26-Feb-1984
0000     95 ;               Use CONFREGL in lieu of CONFREG.
0000     96 ;
0000     97 ;       V03-011 WMC0008         Wayne Cardoza   29-Jan-1983
0000     98 ;               Clean up stack before bugcheck.
0000     99 ;
0000    100 ;       V03-010 WMC0007         Wayne Cardoza   01-Dec-1983
0000    101 ;               Turn CP-IO-BUF into NXM for EXE$MCHK_BUGCHK checks
0000    102 ;               MBOX doesn't always latch cycle type.
0000    103 ;
0000    104 ;       V03-009 WMC0006         Wayne Cardoza   20-Nov 1983
0000    105 ;               Fix transposed lines of code.
0000    106 ;
0000    107 ;       V03-008 TMK0001         Todd M. Katz            14-Nov-1983
0000    108 ;               Add a .TITLE so that the object file for this module
0000    109 ;               will be inserted into SYSLOA.OLB as MCHECK790 instead of
0000    110 ;               as .MAIN.
0000    111 ;
0000    112 ;       V03-007 WMC0005         Wayne Cardoza   02-Nov-1983
0000    113 ;               Fix IOC$BROADCAST calls.
0000    114 ;               Get adaptor type from ABUS_TYPE.
```

```
0000   115 ;
0000   116 ;       V03-006 KDM0053         Kathleen D. Morse       11-Jul-1983
0000   117 ;               Replace cpu-specific IPR references with the cpu-specific
0000   118 ;               symbols, defined by $PR790DEF.
0000   119 ;
0000   120 ;       V03-005 WMC0004         Wayne Cardoza   06-Apr-1983
0000   121 ;               Changes to bad page handling.
0000   122 ;
0000   123 ;       V03-004 WMC0003         Wayne Cardoza   22-Mar-1983
0000   124 ;               Misc minor fixes.
0000   125 ;
0000   126 ;       V03-003 WMC0002         Wayne Cardoza   22-Feb-1983
0000   127 ;               Resource removed from service bit.
0000   128 ;               Spec changes in MSTAT1, MSTAT2, MDECC.
0000   129 ;
0000   130 ;       V03-002 WMC0001         Wayne Cardoza   10-Feb-1983
0000   131 ;               Add error logging routine for adapters.
0000   132 ;
0000   133 ;--
```

```
                        135          .SBTTL  Declarations
                        136 ;
                        137 ; INCLUDE FILES:
                        138 ;
                        139
                        140 ;
                        141 ; INCLUDED SYSTEM SYMBOL DEFINITIONS
                        142 ;
                        143          $ADPDEF
                        144          $EMBDEF <MC,SE>
                        145          $IPLDEF
                        146          $PCBDEF
                        147          $PFNDEF
                        148          $PHDDEF
                        149          $PRDEF
                        150          $PR790DEF
                        151          $PSLDEF
                        152          $PTEDEF
                        153          $RPBDEF
                        154          $VADEF
                        155          $MCF790DEF
                        156          $MCHKDEF
                        157          $MERGDEF
                        158          $SBIADEF
                        159          $CSWPDEF
                        160          $PAMMDEF
                        161
                        162 ;
                        163 ; OWN STORAGE:
                        164 ;
                        165
                        166          .PSECT  MCHK$DATA,QUAD,WRT
                        167
                        168 EXE$MCHK_ERRCNT::                           ; Used to locate error counters
                        169                                             ; via SYS.MAP.
                        170
                        171 IBOX_OLD1:                                  ; Time of most recent IBOX error
00000000                172          .LONG   0
                        173 IBOX_OLD2:                                  ; Second most recent
00000000                174          .LONG   0
                        175 IBOX_TOTAL:                                 ; Total errors
00000000                176          .LONG   0
0000000A                177 IBOX_THRESHOLD = 10                         ; Minimum time allowed for 3 IBOX errors
                        178                                             ; (10 millisec units)
                        179
                        180 FBOX_OLD1:                                  ; Time of most recent FBOX error
00000000                181          .LONG   0
                        182 FBOX_OLD2:                                  ; Second most recent
00000000                183          .LONG   0
                        184 FBOX_TOTAL:                                 ; Total errors
00000000                185          .LONG   0
0000000A                186 FBOX_THRESHOLD = 10                         ; Minimum time allowed for 3 FBOX errors
                        187                                             ; (10 millisec units)
                        188
                        189 EBOX_OLD1:                                  ; Time of most recent EBOX error
00000000                190          .LONG   0
                        191 EBOX_OLD2:                                  ; Second most recent
```

```
00000000 001C   192          .LONG   0
         0020   193 EBOX_TOTAL:                    ; Total errors
00000000 0020   194          .LONG   0
0000000A 0024   195 EBOX_THRESHOLD = 10            ; Minimum time allowed for 3 EBOX errors
         0024   196                                ; (10 millisec units)
         0024   197
         0024   198 MBOX_FE_OLD:                   ; Time of most recent MBOX fatal error
00000000 0024   199          .LONG   0
         0028   200 MBOX_FE_PC:                    ; PC of most recent MBOX fatal error
00000000 0028   201          .LONG   0
         002C   202 MBOX_FE_TOTAL:                 ; Total errors
00000000 002C   203          .LONG   0
00000002 0030   204 MBOX_FE_THRESHOLD = 2          ; Minimum time allowed for 2 MBOX errors
         0030   205                                ; (10 millisec units)
         0030   206 MBOX_FE_PHY_ADR:               ; Physical address of last MBOX address PE
00000000 0030   207          .LONG   0             ; error
         0034   208
         0034   209 TB_OLD1:                       ; Time of last TB parity error.
00000000 0034   210          .LONG   0
         0038   211 TB_OLD2:                       ; Time of next-to-last TB error.
00000000 0038   212          .LONG   0
         003C   213 TB_TOTAL:                      ; Total errors
00000000 003C   214          .LONG   0
0000000A 0040   215 TB_THRESHOLD = 10              ; Allowable time between TB errors
         004C   216                                ; (in 10 millisecond units).
         0040   217
         0040   218 CSH_A_OLD1:                    ; Time of most recent A cache error
00000000 0040   219          .LONG   0
         0044   220 CSH_A_OLD2:                    ; Second most recent
00000000 0044   221          .LONG   0
         0048   222 CSH_B_OLD1:                    ; Time of most recent B cache error
00000000 0048   223          .LONG   0
         004C   224 CSH_B_OLD2:                    ; Second most recent
00000000 004C   225          .LONG   0
         0050   226 CACHE_TOTAL:                   ; Total errors
00000000 0050   227          .LONG   0
0000000A 0054   228 CACHE_THRESHOLD = 10           ; Minimum time allowed for 3 cache errors
         0054   229                                ; (10 millisec units)
         0054   230
         0054   231 MBOX_1D_OLD1:                  ; Time of most recent MBOX 1D error
00000000 0054   232          .LONG   0
         0058   233 MBOX_1D_OLD2:                  ; Second most recent
00000000 0058   234          .LONG   0
         005C   235 MBOX_1D_TOTAL:                 ; Total errors
00000000 005C   236          .LONG   0
0000000A 0060   237 MBOX_1D_THRESHOLD = 10         ; Minimum time allowed for 3 MBOX 1D errors
         0060   238                                ; (10 millisec units)
         0060   239
         0060   240 CRD_OLD1:                      ; Time of most recent single bit error
00000000 0060   241          .LONG   0
         0064   242 CRD_OLD2:                      ; Second most recent
00000000 0064   243          .LONG   0
         0068   244 CRD_TOTAL:                     ; Total errors
00000000 0068   245          .LONG   0
00000064 006C   246 CRD_THRESHOLD = 100            ; Minimum time allowed for 3 CRD errors
         006C   247                                ; (10 millisec units)
         006C   248
```

```
0000012C  006C    249 CRD_REENAB_TIME = 5 * 60                 ; 5 minute CRD disable time
          006C    250 CRD_TIMER:                              ; Time left to reenable CRD's
0000012C  006C    251         .LONG   CRD_REENAB_TIME
          0070    252
          0070    253 EXE$AB_MEMERR::                         ; Memory error counters for adapter routines
000000B0  0070    254         .BLKB   64                      ; One for each TR on each of 4 SBIA's
          00B0    255
0000005A  00B0    256 KEEPALIVE_TIME = 90                     ; 1 1/2 minute console keepalive check
          00B0    257 KEEPALIVE_TIMER:                        ; Time left to check if console is alive
0000005A  00B0    258         .LONG   KEEPALIVE_TIME
          00B4    259 TODR_VALUE:                             ; Check if console is updating this
00000000  00B4    260         .LONG   0                       ; register
```

```
                00B8   262 ;
                00B8   263 ; This is a table that maps one bit for each opcode in the VAX
                00B8   264 ; instruction set.  If the corresponding bit is set, that opcode
                00B8   265 ; does only one read and may be safely restarted if a CP_IO_BUF error occurs.
                00B8   266 ; The table does not take into account reads done for address calculations.
                00B8   267 ; These are assumed to not be relevant in deciding if I/O space was referenced.
                00B8   268 ; Queue, decimal, and string instructions are not assumed to be safe.
                00B8   269 ;
                00B8   270 $BI_INST:
343F            00B8   271          .WORD   ^B0011010000111111      ;HALT,NOP,REI,RET,PROBER/W
FFFF            00BA   272          .WORD   ^B1111111111111111      ;BRANCHES
0000            00BC   273          .WORD   ^B0000000000000000      ;
D00F            00BE   274          .WORD   ^B1101000000001111      ;BRW,CONVERTS,MOVES
7F55            00C0   275          .WORD   ^B0111111101010101      ;FLOATING OPS
014D            00C2   276          .WORD   ^B0000000101001101      ;
7F55            00C4   277          .WORD   ^B0111111101010101      ;
C14F            00C6   278          .WORD   ^B1100000101001111      ;
D555            00C8   279          .WORD   ^B1101010101010101      ;ARITHMETIC BYTE
DFF5            00CA   280          .WORD   ^B1101111111110101      ;CMPB,TSTB
D555            00CC   281          .WORD   ^B1101010101010101      ;ARITHMETIC WORD
FFF5            00CE   282          .WORD   ^B1111111111110101      ;CMPW,TSTW,BISPSW,BICPSW
D555            00D0   283          .WORD   ^B1101010101010101      ;ARITHMETIC LONG
FFF5            00D2   284          .WORD   ^B1111111111110101      ;CMPL,TSTL
CFFF            00D4   285          .WORD   ^B1100111111111111      ;BBS,BBC,BLBS,BLBC,CMPV,CMPZV
0CFD            00D6   286          .WORD   ^B0000110011111101      ;AOB,CVT
```

```
                        00D8   288 ;
                        00D8   289 ; LOCAL VARIABLES FOR MACHINE CHECK HANDLER
                        00D8   290 ;
                        00D8   291
                        00D8   292 ABUS_CYCLE:                             ; Bit mask for ABUS related cycles
08088000                00D8   293         .LONG   1@MCF790$C_ABUS + 1@MCF790$C_ABUS_WRT + 1@MCF790$C_ABUS_REFL
                        00DC   294
                        00DC   295 ASSUME MCF790$V_ABORTS EQ 1             ; Make sure there is room for software
                        00DC   296                                         ; abort bit
00000001                00DC   297 MCHK_ABORT = 1                          ; Used by machine check handler to abort
                        00DC   298
                        00DC   299 ABORT_BITS:                             ; Local copy of abort bits
00000000                00DC   300         .LONG   0
                        00E0   301
                        00E0   302 SBIA_ERR_SUM:                           ; Temp copy of SBIA summary register
00000000                00E0   303         .LONG   0
                        00E4   304
000000C8                00E4   305 CRD_LOG_SIZE = <2 * 4> + <16 * 3 * 4>   ; Buffer size
                        00E4   306 CRD_BUFFER:                             ; Single bit error logging buffer
000000C4                00E4   307         .LONG   4 + <16*3*4>            ; Size of buffer
                        00E8   308 CRD_FLAGS:                              ; Flags - 1 -> logging turned off
00000000                00E8   309         .LONG   0
                        00EC   310 CRD_FIRST_ENTRY:                        ; Beginning of error entries
000001AC                00EC   311         .BLKL   16 * 3                  ; Room for 16 entries
                        01AC   312 CRD_BUF_END:
                        01AC   313
                        01AC   314 CRD_NEXT:                               ; Next CRD slot to be used
00000000                01AC   315         .LONG   0
                        01B0   316
```

```
                                                01B0    318 ;
                                                01B0    319 ; warning messages
                                                01B0    320 ;
                                                01B0    321 FBOX_MSG:
                                00000034'       01B0    322             .LONG   20$-10$
5F 57 5F 4D 45 54 53 59 53 25 0A 0D             01B4    323 10$:        .ASCII  <13><10>-
4F 42 46 20 2C 46 46 4F 58 4F 42 46             01C0
66 66 6F 20 64 65 6E 72 75 74 20 58             01CC
6F 72 72 65 20 6F 74 20 65 75 64 20             01D8
                        73 72                   01E4
                                                01E6    324             \%SYSTEM_W_FBOXOFF, FBOX turned off due to errors\-
                              0A 0D             01E6    325             <13><10>
                                                01E8    326 20$:
                                                01E8    327 ;
                                                01E8    328 ;
                                                01E8    329 CACHE_MSG:
                                00000038'       01E8    330             .LONG   20$-10$
                                      0D        01EC    331 10$:        .ASCII  <13><10>-
43 5F 57 5F 4D 45 54 53 59 53 25 0A             01ED    332             \%SYSTEM_W_CACHEOFF, half of cache turned off due to errors\-
6C 61 68 20 2C 46 46 4F 45 48 43 41             01F9
74 20 65 68 63 61 63 20 66 6F 20 66             0205
75 74 20 66 66 6F 20 64 65 6E 72 75             0211
   73 72 6F 72 72 65 20 6F 74 20 65 20          021D
                              0A 0D             0228    333             <13><10>
                                                022A    334 20$:
```

```
0000000C   336          .PSECT  MCHK$CODE,QUAD,RD,WRT
     0000  337          .SBTTL  Machine check entry point
     0000  338
     0000  339 ;++
     0000  340 ; MCHECK790 -- VENUS MACHINE CHECK
     0000  341 ;
     0000  342 ; FUNCTIONAL DESCRIPTION:
     0000  343 ;       All machine checks are vectored to this entry point.
     0000  344 ;
     0000  345 ; FORMAT OF VENUS'S MACHINE CHECK LOGOUT STACK:
     0000  346 ;       On entry to this machine check handler, the stack is set up as follows:
     0000  347 ;
     0000  348 ;               00(SP): # bytes in machine check log on stack
     0000  349 ;                 .
     0000  350 ;                 .
     0000  351 ;                machine check frame (see MCF790DEF)
     0000  352 ;                 .
     0000  353 ;                 .
     0000  354 ;
     0000  355 ;
     0000  356 ;       As soon as the machine check handler is invoked, it sets up the stack
     0000  357 ;       as follows:
     0000  358 ;       +-------------------------------+       :(SP)
     0000  359 ;       |                               |
     0000  360 ;       |               .               |
     0000  361 ;       |               .               |
     0000  362 ;       |       saved R0 - R6, AP       |
     0000  363 ;       |               .               |
     0000  364 ;       |               .               |
     0000  365 ;       +-------------------------------+
     0000  366 ;       | pointer to PC/PSL of exception|
     0000  367 ;       +-------------------------------+
     0000  368 ;       |       recovery mask           |
     0000  369 ;       +-------------------------------+
     0000  370 ;       | # bytes pushed on logout stack| :(AP)
     0000  371 ;       +-------------------------------+
     0000  372 ;       |                               |
     0000  373 ;       |                               |
     0000  374 ;       |          machine              |
     0000  375 ;       |           check               |
     0000  376 ;       |           frame               |
     0000  377 ;       |                               |
     0000  378 ;       |                               |
     0000  379 ;       +-------------------------------+
     0000  380 ;
     0000  381 ;       AP will point to the beginning of the machine check log on the stack.
     0000  382 ;       2 longwords are immediately pushed on top of the machine check log, and
     0000  383 ;       are referenced as negative offsets from AP.  These two longwords are
     0000  384 ;       input arguments to EXE$MCHK_BUGCHK, a routine that is called to check
     0000  385 ;       for a user-declared machine check recovery block.  This routine is
     0000  386 ;       called immediately before bugchecking, and expects the mask and the
     0000  387 ;       pointer to the exception PC/PSL to be right on top of the machine check
     0000  388 ;       log on the stack.
     0000  388 ;--
```

```
                                0000   390          .ALIGN  LONG
                                0000   391 EXE$MCHK::                          ; Machine check handler.
                          03    DD     0000   392          PUSHL   #MCHK$M_MCK ! MCHK$M_LOG ; Mask signals machine check.
                        60 AE   DF     0002   393          PUSHAL  MCF790$L_PC+4(SP)        ; Push pointer to exception PC/PSL.
                     107F 8F    BB     0005   394          PUSHR   #^M<R0,R1,R2,R3,R4,R5,R6,AP> ; Working registers.
                5C   5E    28    C1     0009   395          ADDL3   #<10*4>,SP,AP            ; AP points to mchk log frame.
                                000D   396 ;
                                000D   397 ; Do misc initialization
                                000D   398 ;
                                000D   399
                                CB     000D   400          BICL3   #^C<MCF790$M_ABORTS ! - ; Get all the hardware abort bits
                                000E   401                      MCF790$M_AUTO_SHUT>,-
OODC'CF   0C AC   FFFDFFE1 8F           000E   402                  MCF790$L_EBCS(AP),W^ABORT_BITS
                       04 AC   94      0018   403          CLRB    MCF790$B_MCHK_CODE(AP)  ; Just in case
                                001B   404 ;
                                001B   405 ; Look for one of the four basic error types
                                001B   406 ; BUGCHECK if we find two
                                001B   407 ;
                                001B   408 ; MBOX fatal errors
                                001B   409 ;
                04 0C AC   0F   E1     001B   410          BBC     #MCF790$V_MBOX_FE, MCF790$L_EBCS(AP), 10$
                   04 AC   04   90     0020   411          MOVB    #MCF790$C_MBOX_FE, MCF790$B_MCHK_CODE(AP)
                                0024   412 ;
                                0024   413 ; FBOX errors
                                0024   414 ;
                09 04 AC   1C   E1     0024   415 10$:     BBC     #MCF790$V_FBOX, MCF790$L_EHSR(AP), 20$
                      04 AC   95       0029   416          TSTB    MCF790$B_MCHK_CODE(AP)  ; Make sure this is the first match
                         34   12       002C   417          BNEQ    BAD_MCHK
                   04 AC   01   90     002E   418          MOVB    #MCF790$C_FBOX, MCF790$B_MCHK_CODE(AP)
                                0032   419 ;
                                0032   420 ; EBOX errors
                                0032   421 ;
   0C AC   00001E00 8F   D3     0032   422 20$:     BITL    #MCF790$M_EDP_PE ! MCF790$M_USTK_PE !-
                                003A   423                  MCF790$M_ECS_PE ! MCF790$M_EMCR_PE, -
                                003A   424                  MCF790$L_EBCS(AP)
                      09   13     003A   425          BEQL    30$
                   04 AC   95       003C   426          TSTB    MCF790$B_MCHK_CODE(AP)  ; Make sure this is the first match
                      21   12       003F   427          BNEQ    BAD_MCHK
                04 AC   02   90     0041   428          MOVB    #MCF790$C_EBOX, MCF790$B_MCHK_CODE(AP)
                                0045   429 ;
                                0045   430 ; IBOX errors
                                0045   431 ;
             09 0C AC   0D   E1     0045   432 30$:     BBC     #MCF790$V_IBOX_ERR, MCF790$L_EBCS(AP), 40$
                   04 AC   95       004A   433          TSTB    MCF790$B_MCHK_CODE(AP)  ; Make sure this is the first match
                      13   12       004D   434          BNEQ    BAD_MCHK
                04 AC   03   90     004F   435          MOVB    #MCF790$C_IBOX, MCF790$B_MCHK_CODE(AP)
                                0053   436 ;
                                0053   437 ; Call appropriate service routine
                                0053   438 ;
                                0053   439 40$:     CASE    MCF790$B_MCHK_CODE(AP),- ; Use machine check code
                                0053   440                  <CHECK_MBOX_ID,-         ; Nothing so far
                                0053   441                  F_ERR,-                  ; FBOX error
                                0053   442                  E_ERR,-                  ; EBOX error
                                0053   443                  I_ERR,-                  ; IBOX error
                                0053   444                  M_ERR>,-                 ; MBOX fatal error
                                0053   445                  TYPE=B
                                0062   446 ; fall thru to bad code
```

```
                        0062   447 ;
                        0062   448 ; Always bugcheck - may be bug in this handler
                        0062   449 ;
                        0062   450 BAD_MCHK:
                 05D3   30 0062   451         BSBW    LOG_MCHECK                      ; log it
                        0065   452         BUG_CHECK BADMCKCOD,FATAL
                        0069   453 ;
                 00C0   30 0069   454 M_ERR:  BSBW    MBOX_FE_SERV                    ; MBOX_FE service routine
                   0D   11 006C   455         BRB     CHECK_MBOX_1D
                        006E   456 ;
                 0238   30 006E   457 F_ERR:  BSBW    FBOX_SERV                       ; FBOX service routine
                   08   11 0071   458         BRB     CHECK_MBOX_1D
                        0073   459 ;
                 0271   30 0073   460 E_ERR:  BSBW    EBOX_SERV                       ; EBOX service routine
                   03   11 0076   461         BRB     CHECK_MBOX_1D
                        0078   462 ;
                 008F   30 0078   463 I_ERR:  BSBW    IBOX_SERV                       ; IBOX service routine
                        007B   464 ;       BRB     CHECK_MBOX_1D
                        007B   465 ;
                        007B   466 ; fall thru to next stage of checking
                        007B   467 ;
                        007B   468 ; MBOX 1D interrupt indicated by MBOX_INTR without MBOX_FE
                        007B   469 ; TB problems are hard to spot - just check for the bits
                        007B   470 ;
                        007B   471 CHECK_MBOX_1D:
      17 0C AC    0F    E0 007B   472         BBS     #MCF790$V_MBOX_FE, MCF790$L_EBCS(AP), MCHK_EXIT
38 AC  00000F00 8F    D3 0080   473         BITL    #MCF790$M_TB_TAG_PE ! MCF790$M_TB_A_PE ! -
                        0088   474                  MCF790$M_TB_B_PE ! MCF790$M_TB_VAL_PE, -
                        0088   475                 MCF790$L_MSTATT(AP)
                   05   12 0088   476         BNEQ    10$                             ; A TB problem
   08 0C AC    0E    E1 008A   477         BBC     #MCF790$V_MBOX_INT, MCF790$L_EBCS(AP), MCHK_EXIT
   00 04 AC    07    E2 008F   478 10$:     BBSS    #MCF790$V_MBOX_1D, MCF790$B_MCHK_CODE(AP), 20$
            039A   30 0094   479 20$:     BSBW    MBOX_1D_SERV
                        0097   480 ;
                        0097   481 ; fall thru to common exit processing
                        0097   482 ;
                        0097   483 MCHK_EXIT:
         04 AC    95 0097   484         TSTB    MCF790$B_MCHK_CODE(AP)
                   C6   13 009A   485         BEQL    BAD_MCHK                        ; We couldn't figure it out
                 0599   30 009C   486         BSBW    LOG_MCHECK                      ; Log it
                        009F   487 ;
                        009F   488 ; If any abort bits are set, we cannot resume the instruction
                        00C F  489 ;
         00DC'CF    D5 00?F   490         TSTL    W^ABORT_BITS
                   12   12 00A?   491         BNEQ    REFLECT_MCHK
         107F 8F    BA 00A>   492         POPR    #^M<R0,R1,R2,R3,R4,R5,R6,AP>
         5E   08    C0 00A9   493         ADDL    #<2*4>,SP                       ; Get rid of recovery block check data
         5E   8E    C0 00AC   494         ADDL    (SP)+,SP                        ; Get rid of the machine chack frame
0000004A 8F   00    DA 00AF   495         MTPR    #0,#PR790$_EHSR                 ; Clear the VMS entered flag
                   02   00B6   496         REI                                     ; At last - continue
```

```
                        00B7   498          .SBTTL  Reflect machine check to user
                        00B7   499 ;++
                        00B7   500 ;       REFLECT MACHINE CHECK TO USER
                        00B7   501 ;
                        00B7   502 ; This code is entered if the machine check was fatal.  It determines
                        00B7   503 ; if it was just fatal to the process which caused it (current process
                        00B7   504 ; is in USER or SUPER mode), or if it was fatal to the entire system
                        00B7   505 ; (current process is in EXEC or KERNEL mode).
                        00B7   506 ;
                        00B7   507 ; If current process is in USER or SUPER mode,
                        00B7   508 ;       set up an exception on user's stack and REI to it
                        00B7   509 ; If current process is in EXEC or KERNEL mode,
                        00B7   510 ;       issue a fatal bugcheck.
                        00B7   511 ;
                        00B7   512 ; CALLING SEQUENCE:
                        00B7   513 ;       BRB/W -- NOTHING EXTRA CAN BE ON THE STACK!!
                        00B7   514 ;
                        00B7   515 ; STACK CONTENTS:
                        00B7   516 ;       00(SP): saved R0,R1,R2,R3,R4,R5,R6,AP
                        00B7   517 ;       20(SP): 2 longword inputs for recovery block check
                        00B7   518 ;       28(SP): (also AP) machine check log -- 1st longword is a byte count.
                        00B7   519 ;--
                        00B7   520 REFLECT_MCHK:                                ; Reflect exception according
                        00B7   521                                              ; to current access mode.
03 60 AC    19    E0    00B7   522          BBS     #PSL$V_CURMOD+1, -         ; Branch if USER or SUPER.
                        00BC   523                  MCF790$L_PSL(AP),10$       ;
          0036    31    00BC   524          BRW     BUGCHECK_NOLOG             ; EXEC or KERNEL; bugcheck.
                        00BF   525
                        00BF   526 10$:                                         ; SUPER or USER; exception.
            50    00    DB    00BF   527          MFPR    #PR$_KSP,R0           ; Get kernel stack pointer.
        70   5C AC    7D    00C2   528          MOVQ    MCF790$L_PC(AP),-(R0) ; Push PC,PSL on kernel stack.
            00    50    DA    00C6   529          MTPR    R0,#PR$_KSP           ; Replace new kernel stack ptr.
            107F  8F    BA    00C9   530          POPR    #^M<R0,R1,R2,R3,R4,R5,R6,AP> ; Restore registers.
            5E    08    C0    00CD   531          ADDL    #<2*4>,SP             ; Pop inputs for recovery block check.
            5E    8E    C0    00D0   532          ADDL    (SP)+,SP              ; Pop mchk log from stack.
                        00D3   533 ;
                        00D3   534 ; Set up an exception stack for current process.
                        00D3   535 ; The faulting PC,PSL pair are still on the interrupt stack.  Alter
                        00D3   536 ; them to look as if an exception has occurred.
                        00D3   537 ;
      6E    00000000'GF   9E    00D3   538          MOVAB   G^EXE$MCHECK,(SP)     ; Replace exception PC.
04 AE   04 AE   02    18    EF    00DA   539          EXTZV   #PSL$V_CURMOD, -      ; Zero exception PSL, except
                        00E1   540                  #PSL$S_CURMOD, -              ; for current access mode.
                        00E1   541                  4(SP),4(SP)
    04 AE   04 AE   16    9C    00E1   542          ROTL    #PSL$V_PRVMOD, -      ; Create a PSL of current mode
                        00E7   543                  4(SP),4(SP)                   ; kernel, correct previous
                        00E7   544                                                ; mode, and IPL 0.
     0000004A 8F    00    DA    00E7   545          MTPR    #0,#PR790$_EHSR       ; Clear "VMS entered" flag.
                        02    00EE   546          REI                           ; Go to exception handler.
```

```
                        00EF    548          .SBTTL  Bugcheck
                        00EF    549 ;++
                        00EF    550 ; If user has declared a recovery block, transfer control to it.
                        00EF    551 ; Else issue a fatal bugcheck.
                        00EF    552 ;
                        00EF    553 ; CALLING SEQUENCE:
                        00EF    554 ;       BRB/W -- NOTHING EXTRA CAN BE ON THE STACK!!!
                        00EF    555 ;
                        00EF    556 ; STACK CONTENTS ON ENTRY:
                        00EF    557 ;       00(SP): saved R0,R1,R2,R3,R4,R5,R6,AP
                        00EF    558 ;       20(SP): 2 longword inputs for recovery block check
                        00EF    559 ;       28(SP): (also AP) machine check log
                        00EF    560 ;--
                        00EF    561 BUGCHECK_POP:
              5E   04   C0  00EF    562          ADDL    #4,SP                   ; Clean off garbage on stack
                        00F2    563 BUGCHECK:
                 0543   30  00F2    564          BSBW    LOG_MCHECK              ; Log it
                        00F5    565 BUGCHECK_NOLOG:                             ; Entry point if already logged
              107F 8F   BA  00F5    566          POPR    #^M<R0,R1,R2,R3,R4,R5,R6,AP> ; Restore registers.
                        00F9    567 ;
                        00F9    568 ; A fatal bugcheck is now inevitable unless a user has declared a machine
                        00F9    569 ; check recovery block.
                        00F9    570 ;
                        00F9    571
      0000004A 8F   00  DA  00F9    572          MTPR    #0,#PR790$_EHSR         ; clear "VMS entered" flag
           00000000'GF  16  0100    573          JSB     G^EXE$MCHK_BUGCHK       ; If return, no recovery block.
                        0106    574          BUG_CHECK -                     ; Issue fatal bugcheck.
                        0106    575                  MACHINECHK,FATAL
```

```
                                010A   577          .SBTTL  IBOX errors
                                010A   578 ;++
                                010A   579 ; An IBOX error has been detected.
                                010A   580 ; These are all treated alike.
                                010A   581 ; This is cause for a bugcheck if they are too frequent.
                                010A   582 ;--
                                010A   583
                                010A   584 IBOX_SERV:
                0008'CF   D6    010A   585          INCL    W^IBOX_TOTAL            ; Count the error
                50   1B   DB    010E   586          MFPR    #PR790$_TODR,R0         ; Current time - 10 millisec units
        52      50   0004'CF   C3    0111   587          SUBL3   W^IBOX_OLD2,R0,R2       ; Time for three errors
                     0A   52   D1    0117   588          CMPL    R2,#IBOX_THRESHOLD      ; Too quick
                          03   1A    011A   589          BGTRU   10$                     ; No
                             FFD0   31    011C   590          BRW     BUGCHECK_POP            ; Clean off return address before BUGCHECK
0004'CF         0000'CF   D0    011F   591 10$:     MOVL    W^IBOX_OLD1, W^IBOX_OLD2 ; Save new times
     0000'CF      50   D0    0126   592          MOVL    R0,W^IBOX_OLD1
                       05    012B   593          RSB
```

```
                                012C    595          .SBTTL  MBOX fatal errors
                                012C    596 ;++
                                012C    597 ; A MBOX fatal error (FE set) has been detected.
                                012C    598 ; Error handling depends on the exact error which occurred.
                                012C    599 ;--
                                012C    600
                                012C    601 MBOX_FE_SERV:
                                012C    602          .ENABL  LSB
                                012C    603
                 002C'CF    D6  012C    604          INCL    W^MBOX_FE_TOTAL         ; Count the error
                                0130    605 ;
                                0130    606 ; First make sure we have a single error, otherwise give up
                                0130    607 ;
          3B 3C AC    07    E0  0130    608          BBS     #MCF790$V_MUL_ERR, MCF790$L_MSTAT2(AP),35$
                                0135    609 ;
                                0135    610 ; Check for cache tag parity errors with the written bit set
                                0135    611 ;
          05 3C AC    04    E1  0135    612          BBC     #MCF790$V_CSH_W, -      ; cache not written
                                013A    613                  MCF790$L_MSTAT2(AP),30$
          31 3C AC    06    E0  013A    614          BBS     #MCF790$V_CSH_TAG_PE, - ; parity error
                                013F    615                  MCF790$L_MSTAT2(AP),35$
                                013F    616 ;
                                013F    617 ; Was it a write to nonexistent memory.
                                013F    618 ; A repeated error is fatal.
                                013F    619 ;
                                013F    620 ; R3 must remain cycle type from this point on
                                013F    621 ;
    53    38 AC    04    1A    EF  013F    622 30$:     EXTZV   #MCF790$V_CYCLE_TYP, -
                                0145    623                  #MCF790$S_CYCLE_TYP, -
                                0145    624                  MCF790$L_MSTAT1(AP),R3          ; get cycle type
            29 3C AC    03    E1  0145    625          BBC     #MCF790$V_NXM, -        ; not NXM
                                014A    626                  MCF790$L_MSTAT2(AP),40$
               FC AC    04    C8  014A    627          BISL    #MCHK$M_NEXM,-4(AP)     ; Add NXM to recovery mask
         0028'CF    5C AC    D1  014E    628          CMPL    MCF790$L_PC(AP), W^MBOX_FE_PC   ; same PC for two errors
                       1A    13  0154    629          BEQL    35$                     ; yes - bugcheck
                  50   1B    DB  0156    630          MFPR    #PR790$_TODR,R0         ; Current time - 10 millisec units
       52    50    0024'CF    C3  0159    631          SUBL3   W^MBOX_FE_OLD,R0,R2     ; Time since last error
                  02    52    D1  015F    632          CMPL    R2, #MBOX_FE_THRESHOLD
                        0C    1F  0162    633          BLSSU   35$                     ; Too quick - bugcheck
            0024'CF    50    D0  0164    634          MOVL    R0,W^MBOX_FE_OLD        ; Save time of this error
         0028'CF    5C AC    D0  0169    635          MOVL    MCF790$L_PC(AP), W^MBOX_FE_PC   ; and the PC
                        05    05  016F    636          RSB
                              0170    637
                     FF7C    31  0170    638 35$:     BRW     BUGCHECK_POP            ; Clean off return address
                              0173    639 ;
                              0173    640 ; Was it an error on instruction access to I/O space
                              0173    641 ; We can try to recover if it never made it out to the SBI
                              0173    642 ; This code will need work if SBI memory is ever supported.  It will not have
                              0173    643 ; physical addresses in the same range as its SBI.
                              0173    644 ;
                              0173    645 ; R3 = CYCLE TYPE
                              0173    646 ;
          03 3C AC    02    E0  0173    647 40$:     BBS     #MCF790$V_CP_IO_BUF, -  ; Check for the error bit
                              0178    648                  MCF790$L_MSTAT2(AP),45$
                        010D    31  0178    649          BRW     130$
    50    38 AC    02    1E    EF  017B    650 45$:     EXTZV   #MCF790$V_DEST_CP, -    ; code for port being serviced
                              0181    651                  #MCF790$S_DEST_CP, -
```

```
                     0181   652                      MCF790$L_MSTAT1(AP), R0
        51   24 AC   D0  0181   653          MOVL    MCF790$L_IVASAV(AP),R1          ; assume it will be op-port
           01   50   91  0185   654          CMPB    R0,#MCF790$C_OP_PORT
                0D   13  0188   655          BEQL    46$                             ; op-port  (most common case)
        51   2C AC   D0  018A   656          MOVL    MCF790$L_ESASAV(AP),R1          ; assume it will be EBOX
           02   50   91  018E   657          CMPB    R0,#MCF790$C_EBOX_PORT
                04   13  0191   658          BEQL    46$                             ; EBOX
        51   28 AC   D0  0193   659          MOVL    MCF790$L_VIBASAV(AP),R1         ; IBUF (very unlikely)
 50   00000000'GF   D0  0197   660 46$:      MOVL    G^MMG$GL_SPTBASE,R0             ; SPT base
        0A 51    1F  E4  019E   661          BBSC    #31,R1,47$                      ; clear the system space bit
           50    08  DB  01A2   662          MFPR    #PR$_P0BR,R0                    ; assume P0 space
        03 51    1E  E1  01A5   663          BBC     #30,R1,47$                      ; it was P0
           50    0A  DB  01A9   664          MFPR    #PR$_P1BR,R0                    ; get P1
  51    51    F7 8F  78  01AC   665 47$:      ASHL    #-9,R1,R1                       ; page number in right space
           50  6041  D0  01B1   666          MOVL    (R0)[R1],R0                     ; PTE
50   50    02   10  EF  01B5   667          EXTZV   #25-9,#2,R0,R0                  ; IOA number (25-26 in PA)
   01    0000'CF40  91  01BA   668          CMPB    W^ABUS_TYPE[R0],#1
                AE  12  01C0   669          BNEQ    35$                             ; not SBIA
           10   50  F0  01C2   670          INSV    R0, #MCF790$V_AB_ADPT,-         ; relative IOA number
        38 AC   02      01C5   671                  #MCF790$S_AB_ADPT, -            ; needed for error logging
                        01C8   672                  MCF790$L_MSTAT1(AP)
  51    0000'CF40  D0  01C8   673          MOVL    W^ABUS_VA[R0],R1                ; base address of SBIA
   9D 08 A1   17  E1  01CE   674          BBC     #SBIA$V_BEL, -                  ; error if error lock not set
                   01D3   675                  SBIA$L_SUMRY(R1),35$
   4E 08 A1   13  E0  01D3   676          BBS     #SBIA$V_CAE, -                  ; CAE - didn't make it to SBI
                   01D8   677                  SBIA$L_SUMRY(R1),70$
           0D   53  91  01D8   678          CMPB    R3, #MCF790$C_CP_WRT            ; all other reads bad
                3F  13  01DB   679          BEQL    60$                             ; go check for OK write errors
                    01DD   680 ;
                    01DD   681 ; Check for read of BRRVR register in UBA.  If this machine check
                    01DD   682 ; is the result of a BRRVR read, then just REI.  Someone will either loose
                    01DD   683 ; a character from a terminal, or a device timeout will result.  This is
                    01DD   684 ; better than a system crash.
                    01DD   685 ;
     38 34 A1   0C  E1  01DD   686          BBC     #SBIA$V_CTO,SBIA$L_SBIERR(R1),55$ ; not a timeout
 50   00000000'GF   D0  01E2   687          MOVL    G^IOC$GL_ADPLIST,R0             ; point to ADP list
                2F  13  01E9   688 50$:     BEQL    55$                             ; done if nothing left on list
  0000'8F    0E A0  B1  01EB   689          CMPW    ADP$W_ADPTYPE(R0),#AT$_UBA ; is this ADP for a UBA?
                21  12  01F1   690          BNEQ    53$                             ; no, look at rest of list
           54   03  D0  01F3   691          MOVL    #3,R4                           ; look at VA's of all 4 BRRVR registers
52   44 A044   09  C1  01F6   692 51$:     ADDL3   #9,ADP$L_UBASCB(R0)[R4],R2 ; calculate address of BRRVR from
                    01FC   693                                                      ; the SCB vector address saved in ADP
                    01FC   694 ; **** NOTE ****  The previous instruction assumes the currently used coding
                    01FC   695 ; sequence for dispatching UBA interrupts in the module INIADP.MAR.  Any
                    01FC   696 ; changes to that code my affect this routine.  The assumptions are that the
                    01FC   697 ; virtual address of the UBA BRRVR register is at an offset of 10. bytes past
                    01FC   698 ; the interrupt vector address (9 is added to the SCB vector value because
                    01FC   699 ; the vector has bit 0 set to indicate handling the interrupt on the interrupt
                    01FC   700 ; stack), that the PC of the instruction accessing BRRVR is 3 bytes past the
                    01FC   701 ; interrupt vector entry, and that R4 and R5 have been pushed onto the stack.
                    01FC   702
        24 AC   62  D1  01FC   703          CMPL    (R2),MCF790$L_IVASAV(AP) ; same va as in machine check stack?
                0F  12  0200   704          BNEQ    52$                             ; branch if not BRRVR reference
           52   06  C2  0202   705          SUBL    #<9-3>,R2                       ; else back up to PC of instruction
                    0205   706                                                      ; in UB interrupt service that reads BRRVR
        5C AC   52  D1  0205   707          CMPL    R2,MCF790$L_PC(AP)              ; does it match PC in MCHECK stack?
                06  12  0209   708          BNEQ    52$                             ; if so, came from UB interrupt service.
```

```
                0028'CF    D4  020B   709          CLRL    W^MBOX_FE_PC                ; matching PC doesn't justify bugcheck
                      15   11  020F   710          BRB     70$                         ; go finish up
                   E2 54   F4  0211   711 52$:     SOBGEQ  R4,51$                      ; loop through all 4
              50   04 A0   D0  0214   712 53$:     MOVL    ADP$L_LINK(R0),R0           ; follow ADP list to end
                      CF   11  0218   713          BRB     50$
                      63   11  021A   714 55$:     BRB     120$                                ; full log & bugcheck (abort first)
                           021C   715 ;
08 A1    00600000 8F    D3  021C   716 60$:     BITL    #SBIA$M_ADP ! SBIA$M_RCP, -         ; ADP or RCP also didn't reach SBI
                           0224   717                  SBIA$L_SUMRY(R1)
                      59   13  0224   718          BEQL    120$
          04 AC    40 8F   88  0226   719 70$:     BISB    #MCF790$M_SBIA_ERR, -               ; add SBIA error summary
                           022B   720                  MCF790$B_MCHK_CODE(AP)
      00E0'CF    08 A1   D0  022B   721          MOVL    SBIA$L_SUMRY(R1),W^SBIA_ERR_SUM ; save the error summary
        08 A1    08 A1   C8  0231   722          BISL    SBIA$L_SUMRY(R1),SBIA$L_SUMRY(R1) ; clear all the error locks
34 A1    00001000 8F    C8  0236   723          BISL    #SBIA$M_CTO,SBIA$L_SBIERR(R1)       ; and the CPU timeout bit
           52    5C BC   9A  023E   724          MOVZBL  @MCF790$L_PC(AP),R2                 ; get the guilty instruction
      01 00B8'CF    52   E0  0242   725          BBS     R2,W^SBI_INST,90$                   ; does instruction only do one read
                           05  0248   726          RSB                                 ; no - eventually abort
                   0E   53   91  0249   727 90$:     CMPB    R3,#MCF790$C_CP_READ        ; read cycle?
                      07   12  024C   728          BNEQ    100$
          00DC'CF    02   CA  024E   729          BICL    #MCF790$M_IO_RD,W^ABORT_BITS     ; clear IO read abort
                      05   11  0253   730          BRB     110$
          00DC'CF    04   CA  0255   731 100$:    BICL    #MCF790$M_MEM_WRT, W^ABORT_BITS  ; clear write abort
                           025A   732 ;
                           025A   733 ; now make sure errors are not repeated
                           025A   734 ;
      0028'CF    5C AC   D1  025A   735 110$:    CMPL    MCF790$L_PC(AP),W^MBOX_FE_PC    ; same PC for two errors
                      1A   13  0260   736          BEQL    115$                        ; yes - bugcheck
                   50   1B   DB  0262   737          MFPR    #PR790$_TODR,R0             ; Current time - 10 millisec units
   52   50    0024'CF    C3  0265   738          SUBL3   W^MBOX_FE_OLD,R0,R2         ; Time since last error
                      02   52  D1  026B   739          CMPL    R2, #MBOX_FE_THRESHOLD
                      0C   1F  026E   740          BLSSU   115$                        ; Too quick - bugcheck
          0024'CF    50   D0  0270   741          MOVL    R0,W^MBOX_FE_OLD            ; Save time of this error
      0028'CF    5C AC   D0  0275   742          MOVL    MCF790$L_PC(AP), W^MBOX_FE_PC   ; and the PC
                           05  027B   743          RSB
                           027C   744 ;
                   FE70    31  027C   745 115$:    BRW     BUGCHECK_POP                ; Clean off return address
                           027F   746 ;
          FC AC    04   C8  027F   747 120$:    BISL    #MCHK$M_NEXM,-4(AP)         ; Add NXM to recovery mask
          04 AC    20   88  0283   748          BISB    #MCF790$M_SBIA, -           ; log full SBIA
                           0287   749                  MCF790$B_MCHK_CODE(AP)      ; eventually abort (bit set)
                           05  0287   750          RSB                             ; and bugcheck (assume kernel mode)
                           0288   751 ;
                           0288   752 ; Give up if there is no cycle type
                           0288   753 ;
                00    53   91  0288   754 130$:    CMPB    R3, #MCF790$C_NOP
                      EF   13  028B   755          BEQL    115$
                           028D   756 ;
                           028D   757 ; if a cycle parameter RAM error, give up
                           028D   758 ;
      00C00000 8F    D3  028D   759          BITL    #MCF790$M_CPR_PE_A ! MCF790$M_CPR_PE_B,-
                38 AC      0293   760                  MCF790$L_MSTAT1(AP)
                      E5   12  0295   761          BNEQ    115$
                           0297   762 ;
                           0297   763 ; Look for an error writing an MBOX register
                           0297   764 ; MBOX confused, so no recovery
                           0297   765 ; R3 = CYCLE TYPE
```

```
                                0297   766 ;
                     02   53   91 0297   767 150$:   CMPB    R3, #MCF790$C_WRITE_REG
                          E0   13 029A   768         BEQL    115$                                    ; yes - bugcheck
                               029C   769
                               029C   770 ;
                               029C   771 ; Is it ABUS PE of some sort
                               029C   772 ;
                               D3 029C   773         BITL    #MCF790$M_AB_DAT_PE ! -
                               029D   774                  MCF790$M_AB_CM_PE,-
                 38 AC 00300000 8F     029D   775                  MCF790$L_MSTAT1(AP)
                          D6   12 02A4   776         BNEQ    115$                                    ; ABUS so go bugcheck
                               02A6   777 ;
                               02A6   778 ; This is an unknown error type
                               02A6   779 ; Either the hardware is very sick or this handler has a bug
                               02A6   780 ;
                        FDB9   31 02A6   781 160$:   BRW     BAD_MCHK
                               02A9   782
                               02A9   783         .DSABL  LSB
```

```
                                02A9    785          .SBTTL  FBOX errors
                                02A9    786 ;++
                                02A9    787 ; An FBOX error has been detected
                                02A9    788 ; These are all treated alike
                                02A9    789 ; This is cause for turning off the FBOX if they are too frequent
                                02A9    790 ; A console message is printed if FBOX is turned off
                                02A9    791 ;--
                                02A9    792 FBOX_SERV:
                0014'CF   D6    02A9    793          INCL    W^FBOX_TOTAL                ; Count the error
                50    1B  DB    02AD    794          MFPR    #PR790$_TODR,R0             ; Current time - 10 millisec units
        52  50  0010'CF   C3    02B0    795          SUBL3   W^FBOX_OLD2,R0,R2           ; Time for three errors
                0A    52  D1    02B6    796          CMPL    R2,#FBOX_THRESHOLD          ; Too quick
                      1F  1A    02B9    797          BGTRU   10$                         ; No
                28    00  DA    02BB    798          MTPR    #0,#PR790$_ACCS             ; Turn FBOX off
                      10  88    02BE    799          BISB    #MCF790$M_RSRC_REM,-        ; Indicate resource removed
                  04  AC        02C0    800                  MCF790$B_MCHK_CODE(AP)
            51  01B0'CF   D0    02C2    801          MOVL    W^FBOX_MSG,R1               ; Mesage length
            52  01B4'CF   9E    02C7    802          MOVAB   W^FBOX_MSG+4,R2             ; Message address
         55  00000000'GF   9E    02CC    803          MOVAB   G^OPA$CBO,R5                ; Send it to the console terminal
             00000000'GF   16    02D3    804          JSB     G^IOC$BROADCAST
                           05    02D9    805          RSB
    0010'CF  000C'CF   D0    02DA    806 10$:     MOVL    W^FBOX_OLD1, W^FBOX_OLD2 ; Save new times
             000C'CF   50    D0    02E1    807          MOVL    R0,W^FBOX_OLD1
                           05    02E6    808          RSB
```

```
                        02E7   810          .SBTTL  EBOX errors
                        02E7   811 ;++
                        02E7   812 ; An EBOX error has been detected
                        02E7   813 ; These are all treated alike.
                        02E7   814 ; This is cause for a bugcheck if they are too frequent.
                        02E7   815 ;--
                        02E7   816
                        02E7   817 EBOX_SERV:
                        02E7   818 ;
                        02E7   819 ; First see if the error is a result of an MBOX problem
                        02E7   820 ;
    05 0C AC    09  E1  02E7   821          BBC     #MCF790$V_EDP_PE, MCF790$L_EBCS(AP), 10$
                0E  E0  02EC   822          BBS     #MCF790$V_MBOX_INT,-    ; It was the MBOX - don't blame EBOX
          21 0C AC          02EE   823                  MCF790$L_EBCS(AP), 30$
           0020'CF  D6  02F1   824 10$:    INCL    W^EBOX_TOTAL            ; Count the error
           50   1B  DB  02F5   825          MFPR    #PR790$_TODR,R0         ; Current time - 10 millisec units
52  50   001C'CF  C3  02F8   826          SUBL3   W^EBOX_OLD2,R0,R2       ; Time for three errors
           0A   52  D1  02FE   827          CMPL    R2,#EBOX_THRESHOLD      ; Too quick
                03  1A  0301   828          BGTRU   20$                     ; No
              FDE9  31  0303   829          BRW     BUGCHECK_POP            ; Clean off return address before BUGCHECK
001C'CF   0018'CF   D0  0306   830 20$:    MOVL    W^EBOX_OLD1,W^EBOX_OLD2 ; Save new times
     0018'CF   50   D0  030D   831          MOVL    R0,W^EBOX_OLD1
                    05  0312   832 3C$:    RSB
```

```
                        0313   834          .SBTTL  Bad Data or Double Bit error
                        0313   835 ;++
                        0313   836 ; If possible a new copy of the page is paged in.
                        0313   837 ; A double bit error will cause the physical page to be put on the bad page list
                        0313   838 ; while a bad data error simply gets a new copy since it is not the array that
                        0313   839 ; is at fault.
                        0313   840 ;--
                        0313   841 BAD_MEM:
        03 60 AC    1A  E1  0313   842          BBC     #PSL$V_IS, MCF790$L_PSL(AP),1$
                  FDD4  31  0318   843          BRW     BUGCHECK_POP            ; Interrupt stack - must give up
02   60 AC    05    10  EC  031B   844 1$:       CMPV    #PSL$V_IPL, #PSL$S_IPL, -
                        0321   845                  MCF790$L_PSL(AP), #IPL$_ASTDEL ; Are we at a non-pageable priority?
                     4E 14  0321   846          BGTR    12$                     ; Abort - recovery is useless
  50    50 AC    F7  8F  78  0323   847          ASHL    #-9, MCF790$L_MEAR(AP),R0 ; Get physical PFN of error
    00000000'GF    50  D1  0329   848          CMPL    R0,G^MMG$GL_MAXPFN      ; Is there PFN data base for page?
                     3F 1A  0330   849          BGTRU   12$                     ; Br if no PFN data base for page
    7E  00000000'GF    D0  0332   850          MOVL    G^PFN$AL_PTE,-(SP)
          53    9E40   D0  0339   851          MOVL    @(SP)+[R0],R3           ; address of PTE
                     04 13  033D   852          BEQL    3$                      ; none - give up
                     63 D5  033F   853          TSTL    (R3)
                     03 19  0341   854          BLSS    5$                      ; Branch if page valid
                  00D0  31  0343   855 3$:       BRW     RDSNONRES               ; else fatal error
           27 63    15  E0  0346   856 5$:       BBS     #PTE$V_WINDOW,(R3),12$  ; BR if page is PFN-mapped
                     14 E1  034A   857          BBC     #MCF790$V_DBL_BIT,-     ; Is it a double bit error
             0B 40 AC       034C   858                  MCF790$L_MDECC(AP),7$   ; If not, let page go to free list
    7E  00000000'GF    D0  034F   859          MOVL    G^PFN$AB_TYPE,-(SP)     ; PFN type array address
          9E40   20  88  0356   860          BISB    #PFN$M_BADPAG,@(SP)+[R0] ; Mark page bad
                     51 D4  035A   861 7$:       CLRL    R1                      ; Clear modify bit propagator
           04 63    1A  E5  035C   862          BBCC    #PTE$V_MODIFY,(R3),10$  ; Test (& clear) modify bit in PTE
           51    80 8F  9A  0360   863          MOVZBL  #PFN$M_MODIFY,R1        ; Set modify propagator
    7E  00000000'GF    D0  0364   864 10$:      MOVL    G^PFN$AB_STATE,-(SP)    ; Address of PFN state array
          9E40   51  88  036B   865          BISB    R1,@(SP)+[R0]           ; Propagate modify bit to PFN database
                        036F   866
                        036F   867          ASSUME  PFN$M_MODIFY EQ 128
                        036F   868
                     06 14  036F   869          BGTR    15$                     ; Page not modified - he's OK
      00DC'CF    01  88  0371   870 12$:       BISB    #MCHK_ABORT,W^ABORT_BITS ; Force an abort
                     05 0376   871          RSB
  7E  00000000'GF    D0  0377   872 15$:       MOVL    G^PFN$AW_REFCNT,-(SP)   ; Address of PFN refcnt array
        01    9E40   B1  037E   873          CMPW    @(SP)+[R0],#1           ; Check for I/O in progress, etc.
                     ED 1A  0382   874          BGTRU   12$                     ; If so, don't try anything fancy
  7E  00000000'GF    D0  0384   875          MOVL    G^PFN$AB_TYPE,-(SP)     ; Address of PFN type array
                        038B   876
                        038B   877 ;        In the future we may recover from hard ecc errors on global pages
                        038B   878 ;        as well, but for now we abort the image.
                        038B   879
                        038B   880          ASSUME  PFN$C_SYSTEM EQ 1       ; Check type of page
                        038B   881          ASSUME  PFN$C_PROCESS EQ 0
                        038B   882
  9E40    03    00  EC  038B   883          CMPV    #PFN$V_PAGTYP, #PFN$S_PAGTYP, -
                     01 0390   884                  @(SP)+[R0],#PFN$C_SYSTEM ; Check for system or global page
                     DE 1A  0391   885          BGTRU   12$                     ; Branch if page table or global page
                     0D 12  0393   886          BNEQ    30$                     ; Branch if page is process private
54  00000000'GF    9E  0395   887          MOVAB   G^MMG$AL_SYSPCB,R4      ; System pages are kept track of in
      55    6C A4  D0  039C   888          MOVL    PCB$L_PHD(R4),R5        ; a working set list in the system pcb
                     2A 11  03A0   889          BRB     40$
                        03A2   890                                          ; Find the PHD and PCB for process page
```

```
55   53  00000000'GF  C3  03A2  30$:    SUBL3   G^SWP$GL_BALBASE,R3,R5    ; Bytes past first balance set page
     55  00000000'GF  C6  03AA          DIVL    G^SWP$GL_BSLOTSZ,R5       ; Process header index
      55   55   F7 8F  78  03B1          ASHL    #-9,R5,R5                 ; Divide by page size
     55  00000000'GF  C4  03B6          MULL    G^SWP$GL_BSLOTSZ,R5       ; Convert process index
        55   55   09  9C  03BD          ROTL    #9,R5,R5                  ; to process header address
     55  00000000'GF  C0  03C1          ADDL    G^SWP$GL_BALBASE,R5
          54   78 A5  D0  03C8          MOVL    PHD$L_PCB(R5),R4          ; Get PCB
        00 63     1F  E5  03CC  40$:    BBCC    #PTE$V_VALID,(R3),50$     ; Clear valid bit from PTE
             39   00  DA  03D0  50$:    MTPR    #0,#PR$_TBIA              ; Invalidate translation buffer
     7E  00000000'GF  D0  03D3          MOVL    G^PFN$AQ_REFCNT,-(SP)     ; Address of PFN refcnt array
                9E40  B7  03DA          DECW    @(SP)+[R0]                ; Reduce reference count to 0
                  06  18  03DD          BGEQ    60$                       ;
         00000000'GF  16  03DF          JSB     G^MMG$REFCNTNEG
     7E  00000000'GF  D0  03E5  60$:    MOVL    G^PFN$Ax_WSLX,-(SP)       ; Address of PFN WSLX array
                          03EC                  PFN_REFERENCE   -
                          03EC          MOVZWL  <@(SP)+[R0],R1>,-         ; Get working-set list index for page
                          03EC                  LONG_OPCODE=MOVL,-
                          03EC                  IMAGE=SYSLOA790.EXE
         00000000'GF  16  03FE          JSB     G^MMG$DELWSLEX            ; Delete it from the working set its in
     7E  00000000'GF  D0  0404          MOVL    G^PFN$AB_STATE,-(SP)      ; PFN type array address
           9E40   10  88  040B          BISB    #PFN$M_DELCON,@(SP)+[R0]  ; Delete contents
         00000000'GF  16  040F          JSB     G^MMG$RELPFN              ; Put page on appropriate list
                          0415
                          0415  ; At this point, the PTE for the bad page contains its mass storage
                          0415  ; address. This will cause a fresh copy of the page to be fetched when
                          0415  ; the process is resumed.
                          0415
                      05  0415          RSB                               ; Log machine check and resume process
                          0416
                          0416  RDSNONRES:                                ;
                021F  30  0416          BSBW    LOG_MCHECK                ; Go log it
     0000004A 8F  00  DA  0419          MTPR    #0,#PR790$_EHSR           ; Clear the VMS entered flag
             5E   04  C0  0420          ADDL    #4,SP                     ; Clean off return address
             107F 8F  BA  0423          POPR    #^M<R0,R1,R2,R3,R4,R5,R6,AP>
         00000000'GF  16  0427          JSB     G^EXE$MCHK_BUGCHK         ; Recovery block in effect?
                          042D          BUG_CHECK RDSNONRES,FATAL         ; Read data substitute page nonresident
```

```
                                0431    928          .SBTTL MBOX 1D errors
                                0431    929 ;++
                                0431    930 ; An MBOX error was reported when IPL dropped below 1D
                                0431    931 ; these are not directly related to instruction execution
                                0431    932 ; From here we will dispatch to one of two subcategories
                                0431    933 ;       asynchronous CPU errors
                                0431    934 ;       DMA errors
                                0431    935 ; TB errors do not latch the cycle type, so a separate check is required
                                0431    936 ; for them
                                0431    937 ;
                                0431    938 ; Cycle parameter RAM errors are occasionally reported here instead of as
                                0431    939 ; MBOX FE
                                0431    940 ;--
                                0431    941 MBOX_1D_SERV:
                                0431    942 ;
                                0431    943 ; First make sure we have a single error, otherwise give up
                                0431    944 ;
          0A 3C AC    07    E0  0431    945          BBS     #MCF790$V_MUL_ERR, MCF790$L_MSTAT2(AP),5$
                                0436    946 ;
                                0436    947 ; Now look for that errant CPR parity error
                                0436    948 ;
        00C00000 8F     D3      0436    949          BITL    #MCF790$M_CPR_PE_A ! MCF790$M_CPR_PE_B,-
              38 AC                     043C    950                  MCF790$L_MSTAT1(AP)
                 03     13      043E    951          BEQL    10$
               FCAC     31      0440    952 5$:      BRW     BUGCHECK_POP            ; Disaster - give up
                                0443    953 ;
                                0443    954 ; Finally, do the normal error processing
                                0443    955 ;
    38 AC   00000F00 8F     D3  0443    956 10$:     BITL    #MCF790$M_TB_TAG_PE ! -
                                044B    957                  MCF790$M_TB_A_PE ! -
                                044B    958                  MCF790$M_TB_B_PE ! -
                                044B    959                  MCF790$M_TB_VAL_PE, -
                                044B    960                  MCF790$L_MSTAT1(AP)
                    11  12      044B    961          BNEQ    MBOX_1D_CPU             ; A TB problem
53  38 AC    04    1A  EF      044D    962          EXTZV   #MCF790$V_CYCLE_TYP, -
                                0453    963                  #MCF790$S_CYCLE_TYP, -
                                0453    964                  MCF790$L_MSTAT1(AP),R3  ; get cycle type
      02 00D8'CF    53  E0      0453    965          BBS     R3,W^ABUS_CYCLE,20$     ; is it ABUS related
                    03  11      0459    966          BRB     MBOX_1D_CPU             ; a CPU error
                  0117  31      045B    967 20$:     BRW     MBOX_1D_DMA             ; a DMA error
```

```
                                045E   969          .SBTTL  MBOX 1D asynchronous CPU error
                                045E   970 ;++
                                045E   971 ; Handle an MBOX 1D error triggered by a CPU reference
                                045E   972 ; In general, these are non-fatal.  We bugcheck only if the error rate is
                                045E   973 ; too high.
                                045E   974 ; Whenever bad data is written, we ignore the error until someone reads it.
                                045E   975 ; When bad data is consumed, the assumption is that it will be used (not
                                045E   976 ; absolutely true, but not worth the effort for the slight chance that it won't
                                045E   977 ; be used) and we treat this like the MBOX FE case.
                                045E   978 ;--
                                045E   979 MBOX_1D_CPU:
                                045E   980          .ENABL  LSB
                                045E   981 ;
                                045E   982 ; First check for translation buffer problems
                                045E   983 ;
      38 AC    00000F00 8F    D3  045E   984          BITL    #MCF790$M_TB_TAG_PE ! -
                                0466   985                   MCF790$M_TB_A_PE ! -
                                0466   986                   MCF790$M_TB_B_PE ! -
                                0466   987                   MCF790$M_TB_VAL_PE, -
                                0466   988                   MCF790$L_MSTAT1(AP)
                          22    13  0466   989          BEQL    20$                     ; Not a TB problem
                   003C'CF      D6  0468   990          INCL    W^TB_TOTAL              ; Count the error
                   50    1B     DB  046C   991          MFPR    #PR790$_TODR,R0         ; Current time - 10 millisec units
   52   50     0038'CF      C3  046F   992          SUBL3   W^TB_OLD2,R0,R2         ; Time for three errors
                   0A    52     D1  0475   993          CMPL    R2,#TB_THRESHOLD        ; Too quick
                         03     1A  0478   994          BGTRU   10$                     ; No
                      FC72      31  047A   995          BRW     BUGCHECK_POP            ; Clean off return address before BUGCHECK
   0038'CF      0034'CF      D0  047D   996 10$:     MOVL    W^TB_OLDT, W^TB_OLD2    ; Save new times
         0034'CF    50     D0  0484   997          MOVL    R0,W^TB_OLD1
                           05  0489   998          RSB
                               048A   999 ;
                               048A  1000 ; Next cache errors.
                               048A  1001 ; Cache errors are not counted if the bad data flag is set.  This is assumed
                               048A  1002 ; to be the fault of some other hardware.  Some previous error reported this.
                               048A  1003 ;
   3C AC    00000060 8F    D3  048A  1004 20$:     BITL    #MCF790$M_CSH_TAG_W ! -
                               0492  1005                   MCF790$M_CSH_TAG_PE, -
                               0492  1006                   MCF790$L_MSTAT2(AP)     ; Is error in tag or written bit
                         0D    12  0492  1007          BNEQ    CACHE_ERR               ; Yes - process cache error
                         E1    E1  0494  1008          BBC     #MCF790$V_CSH_DAT_BW ! - ; Is it cache error
                               0495  1009                   MCF790$V_CSH_DAT_NBW,-
      44 38 AC     03          0495  1010                   MCF790$L_MSTAT1(AP),70$ ; No
                   16    E1    0499  1011 30$:     BBC     #MCF790$V_BAD_DATA,-    ; If BAD DATA error - not fault of cache
         03 40 AC            049B  1012                   MCF790$L_MDECC(AP), -   ; or it has already been accounted for
                               049E  1013                   CACHE_ERR
                               049E  1014 ;
                               049E  1015 ; Cache has bad data.  It either got it from memory or it was written into
                               049E  1016 ; cache by CPU or IO.  In either case, the memory page is now useless.
                               049E  1017 ;
                      FE72      31  049E  1018          BRW     BAD_MEM                 ; Go try to recover
                               04A1  1019 ;
                               04A1  1020 ; Check for too many errors and turn off cache if necessary
                               04A1  1021 ;
                               04A1  1022 CACHE_ERR:                              ; Entry point used for FE and others
                   0050'CF     D6  04A1  1023          INCL    W^CACHE_TOTAL           ; Count the error
                   50    1B    DB  04A5  1024          MFPR    #PR790$_TODR,R0         ; Current time - 10 millisec units
                         02    E0  04A8  1025          BBS     #MCF790$V_CSH_ERR,-     ; It was cache B
```

```
                18 38 AC        04AA  1026                    MCF790$L_MSTAT1(AP), 50$
      52    50   0044'CF   C3   04AD  1027          SUBL3     W^CSH_A_OLD2,R0,R2      ; Time for three errors
                   0A   52   D1   04B3  1028          CMPL      R2,#CACHE_THRESHOLD     ; Too quick
                        16   1F   04B6  1029          BLSSU     60$
0044'CF    0040'CF   D0   04B8  1030          MOVL      W^CSH_A_OLD1, W^CSH_A_OLD2 ; Save new times
     0040'CF    50   D0   04BF  1031          MOVL      R0, W^CSH_A_OLD1
                     05   04C4  1032          RSB
                          04C5  1033
      52    50   004C'CF   C3   04C5  1034 50$:     SUBL3     W^CSH_B_OLD2, R0,R2     ; Time for three errors
                   0A   52   D1   04CB  1035          CMPL      R2,#CACHE_THRESHOLD     ; Too quick
                        6E   1F   04CE  1036 60$:     BLSSU     CACHE_OFF
004C'CF    0048'CF   D0   04D0  1037          MOVL      W^CSH_B_OLD1, W^CSH_B_OLD2 ; Save new times
     0048'CF    50   D0   04D7  1038          MOVL      R0, W^CSH_B_OLD1
                     05   04DC  1039          RSB
                          04DD  1040 ;
                          04DD  1041 ; data address parity error
                          04DD  1042 ;
    28 40 AC    13   E1   04DD  1043 70$:     BBC       #MCF790$V_ADR_PE, -             ; Something else
                          04E2  1044                    MCF790$L_MDECC(AP),80$
                50 AC   D1   04E2  1045          CMPL      MCF790$L_MEAR(AP),-             ; same physical address
                   0030'CF        04E5  1046                    W^MBOX_FE_PHY_ADR
                        32   13   04E8  1047          BEQL      95$                     ; yes - bugcheck
                   50    1B   DB   04EA  1048          MFPR      #PR790$_TODR,R0         ; Current time - 10 millisec units
      52    50   0024'CF   C3   04ED  1049          SUBL3     W^MBOX_FE_OLD,R0,R2     ; Time since last error
                   02   52   D1   04F3  1050          CMPL      R2, #MBOX_FE_THRESHOLD
                        24   1F   04F6  1051          BLSSU     95$                     ; Too quick - bugcheck
     0024'CF    50   D0   04F8  1052          MOVL      R0,W^MBOX_FE_OLD        ; Save time of this error
  0028'CF    5C AC   D0   04FD  1053          MOVL      MCF790$L_PC(AP), W^MBOX_FE_PC   ; Save PC
  0030'CF    50 AC   D0   0503  1054          MOVL      MCF790$L_MEAR(AP), W^MBOX_FE_PHY_ADR ; Save physical address
                     05   0509  1055          RSB
                          050A  1056 ;
                          050A  1057 ; Double bit or bad data errors from memory.
                          050A  1058 ; These errors are either in the context of the correct process or the BAD_MEM
                          050A  1059 ; routine will spot it
                          050A  1060 ;
 40 AC    00500000 8F   D3   050A  1061 80$:     BITL      #MCF790$M_BAD_DATA ! -  ; check error bits
                          0512  1062                     MCF790$M_DBL_BIT, -
                          0512  1063                    MCF790$L_MDECC(AP)
                     03   13   0512  1064          BEQL      90$                     ; Nope - something else
                   FDFC   31   0514  1065          BRW       BAD_MEM                 ; Go try to recover
                          0517  1066 ;
                          0517  1067 ; ABUS bad data flag
                          0517  1068 ;
                     0E   E0   0517  1069 90$:     BBS       #MCF790$V_AB_BAD_DAT,-  ; Go count misc errors
                  3C AC        0519  1070                    MCF790$L_MSTAT2(AP),-
                     00        051B  1071                    MISC_MBOX_1D
                          051C  1072 95$:                                   ; Fall through to misc
                          051C  1073 ;
                          051C  1074 ; misc MBOX errors
                          051C  1075 ;
                          051C  1076 MISC_MBOX_1D:
                005C'CF   D6   051C  1077          INCL      W^MBOX_1D_TOTAL         ; Count the error
                50    1B   DB   0520  1078          MFPR      #PR790$_TODR,R0         ; Current time - 10 millisec units
  52    50    0058'CF   C3   0523  1079          SUBL3     W^MBOX_1D_OLD2,R0,R2    ; Time for three errors
                0A   52   D1   0529  1080          CMPL      R2, #MBOX_1D_THRESHOLD  ; Too quick
                     03   1A   052C  1081          BGTRU     100$                    ; No
                   FBBE   31   052E  1082          BRW       BUGCHECK_POP            ; Clean off return address before BUGCHECK
```

```
                0058'CF   0054'CF   D0  0531  1083 100$:   MOVL    W^MBOX_1D_OLD1, W^MBOX_1D_OLD2 ; Save new times
                     0054'CF   50   D0  0538  1084         MOVL    R0, W^MBOX_1D_OLD1
                                    05  053D  1085         RSB
                                        053E  1086         .DSABL  LSB
                                        053E  1087 ;
                                        053E  1088 ; Handle turning off the correct half of cache
                                        053E  1089 ;
                                        053E  1090 CACHE_OFF:
                50   00000042 8F   DB  053E  1091         MFPR    #PR790$_CSWP,R0         ; Get current cache state
                                02  E0  0545  1092         BBS     #MCF790$V_CSH_ERR,-     ; Which half is at fault
                          05 38 AC          0547  1093                 MCF790$L_MSTAT1(AP),10$
                           50   06  88  054A  1094         BISB    #CSWP$M_C1ENA ! CSWP$M_VAL,R0   ; Cache 0 - leave 1 on
                                03  11  054D  1095         BRB     20$
                           50   05  88  054F  1096 10$:     BISB    #CSWP$M_C0ENA ! CSWP$M_VAL,R0   ; Cache 1 - leave 0 on
                00000042 8F   50   DA  0552  1097 20$:     MTPR    R0,#PR790$_CSWP         ; Disable it
                              10   88  0559  1098         BISB    #MCF790$M_RSRC_REM,-    ; Indicate resource removed
                                  04 AC         055B  1099                 MCF790$B_MCHK_CODE(AP)
                     51   01E8'CF   D0  055D  1100         MOVL    W^CACHE_MSG,R1          ; Mesage length
                     52   01EC'CF   9E  0562  1101         MOVAB   W^CACHE_MSG+4,R2        ; Message address
                55    00000000'GF   9E  0567  1102         MOVAB   G^OPA$UCB0,R5           ; Send it to the console terminal
                      00000000'GF   16  056E  1103         JSB     G^IOC$BROADCAST
                                    05  0574  1104         RSB
```

```
                                0575  1106          .SBTTL  MBOX 1D ABUS (DMA) errors
                                0575  1107 ;++
                                0575  1108 ; Handle an MBOX 1D error triggered by an ABUS reference.
                                0575  1109 ; In general, these are non-fatal.  We bugcheck only if the error rate is
                                0575  1110 ; too high.
                                0575  1111 ; Whenever bad data is written, the error is ignored until someone reads it.
                                0575  1112 ; Errors on reading are handled by device drivers.
                                0575  1113 ;--
                                0575  1114 MBOX_1D_DMA:
                                0575  1115 ;
                                0575  1116 ;       Cycle type is in R3  -  This must not be destroyed
                                0575  1117 ;
                                0575  1118 ; All errors require logging the ABUS error summary register and unlocking the
                                0575  1119 ; adaptor.
                                0575  1120 ;
                      10   EF   0575  1121          EXTZV   #MCF790$V_AB_ADPT,-     ; relative IOA number
            50   38 AC   02        0577  1122                  #MCF790$S_AB_ADPT, -
                                057B  1123                  MCF790$L_MSTAT1(AP), R0
             01   0000'CF40   91   057B  1124          CMPB    W^ABUS_TYPE[R0],#1
                           6F   12   0581  1125          BNEQ    50$                     ; not SBIA
             51   0000'CF40   D0   0583  1126          MOVL    W^ABUS_VA[R0],R1        ; base address of SBIA
              04 AC   40 8F   88   0589  1127          BISB    #MCF790$M_SBIA_ERR, -   ; add SBIA error summary
                                058E  1128                  MCF790$B_MCHK_CODE(AP)
                     08 A1   D0   058E  1129          MOVL    SBIA$L_SUMRY(R1),-      ; Save the error summary
                  00E0'CF            0591  1130                  W^SBIA_ERR_SUM
              08 A1   08 A1   C8   0594  1131          BISL    SBIA$L_SUMRY(R1), -     ; Clear all the error locks
                                0599  1132                  SBIA$L_SUMRY(R1)
        34 A0   00001000 8F   C8   0599  1133          BISL    #SBIA$M_CTO, -          ; and the CPU timeout bit
                                05A1  1134                  SBIA$L_SBIERR(R0)
                                05A1  1135 ;
                                05A1  1136 ; Check for cache errors.
                                05A1  1137 ; Cache errors are not counted if the bad data flag is set.  This is assumed
                                05A1  1138 ; to be the fault of some other hardware.  Some previous error reported this.
                                05A1  1139 ;
     3C AC   00000060 8F   D3   05A1  1140 20$:    BITL    #MCF790$M_CSH_TAG_W ! -
                                05A9  1141                    MCF790$M_CSH_TAG_PE, -
                                05A9  1142                  MCF790$L_MSTAT2(AP)     ; Is error in tag or written bit
                           0B   12   05A9  1143          BNEQ    35$                     ; Yes - process cache error
                           E1   05AB  1144          BBC     #MCF790$V_CSH_DAT_BW ! - ; Is it cache error
                                05AC  1145                  MCF790$V_CSH_DAT_NBW,-
            13 38 AC   03        05AC  1146                  MCF790$L_MSTAT1(AP),40$ ; No
                           16   E1   05B0  1147 30$:    BBC     #MCF790$V_BAD_DATA,-    ; If BAD DATA error - not fault of cache
                 0B 40 AC        05B2  1148                  MCF790$L_MDECC(AP),38$  ; or it has already been accounted for
                           05   05B5  1149          RSB
            05 3C AC   04   E1   05B6  1150 35$:    BBC     #MCF790$V_CSH_W, -      ; cache not written
                                05BB  1151                  MCF790$L_MSTAT2(AP),38$
            32 3C AC   06   E0   05BB  1152          BBS     #MCF790$V_CSH_TAG_PE, - ; parity error
                                05C0  1153                  MCF790$L_MSTAT2(AP),50$ ; tag error with W bit set - bugcheck
                      FEDE   31   05C0  1154 38$:    BRW     CACHE_ERR
                                05C3  1155 ;
                                05C3  1156 ; Check for MDECC errors.
                                05C3  1157 ; Simply log them except for a bugcheck on a data address parity error.  Here
                                05C3  1158 ; data probably went to a wrong location so crash fast.
                                05C3  1159 ; In the case of double bit errors in memory, the page will eventually go to
                                05C3  1160 ; the bad page list.
                                05C3  1161 ; The device will also get an error on reads.
                                05C3  1162 ;
```

```
                    13   E0  05C3  1163 40$:    BBS     #MCF790$V_ADR_PE,-      ; Address parity error - bugcheck
                 2A 40 AC        05C5  1164              MCF790$L_MDECC(AP),50$
          40 AC  00500000 8F   D3  05C8  1165          BITL    #MCF790$M_BAD_DATA ! -
                                   05D0  1166               MCF790$M_DBL_BIT, -
                                   05D0  1167               MCF790$L_MDECC(AP)
                    23   13      05D0  1168          BEQL    60$                     ; Try something else
           50   50 AC    F7 8F   78  05D2  1169          ASHL    #-9, MCF790$L_MEAR(AP),R0 ; Get physical PFN of error
             00000000'GF   50   D1  05D8  1170          CMPL    R0,G^MMG$GL_MAXPFN      ; Is there PFN data base for page?
                    10   1A      05DF  1171          BGTRU   45$                     ; Br if no PFN data base for page
                    14   E1      05E1  1172          BBC     #MCF790$V_DBL_BIT,-     ; Is it a double bit error
                 0B 40 AC        05E3  1173              MCF790$L_MDECC(AP),45$  ; If not, let page go to free list
              7E  00000000'GF   D0  05E6  1174          MOVL    G^PFN$AB_TYPE,-(SP)     ; PFN type array address
                   9E40   20   88  05ED  1175          BISB    #PFN$M_BADPAG,a(SP)+[R0] ; Mark page bad
                          05   05F1  1176 45$:    RSB                             ; Log it
                               05F2  1177 ;
                        FAFA   31  05F2  1178 50$:    BRW     BUGCHECK_POP            ; Clean off return address before BUGCHECK
                               05F5  1179 ;
                               05F5  1180 ; Check for NXM
                               05F5  1181 ; Recovery from a write to memory is impossible and it doesn't seem to be
                               05F5  1182 ; worthwhile to try to distiguish it from a read since something is very
                               05F5  1183 ; wrong in any case.
                               05F5  1184 ;
                 F8 3C AC   03   E0  05F5  1185 60$:    BBS     #MCF790$V_NXM, -        ; NXM
                               05FA  1186              MCF790$L_MSTAT2(AP),50$
                               05FA  1187 ;
                               05FA  1188 ; The only other error to really worry about is ABUS address PE or control PE
                               05FA  1189 ; All other errors return a device error.
                               05FA  1190 ;
          38 AC  00180000 8F   D3  05FA  1191 90$:    BITL    #MCF790$M_AB_ADR_PE ! -
                               0602  1192               MCF790$M_AB_CM_PE, -
                               0602  1193               MCF790$L_MSTAT1(AP)
                        EE   12  0602  1194          BNEQ    50$
                               05  0604  1195          RSB                             ; Not address or control PE
```

```
                     0605  1197          .SBTTL  SBI vectors & error handling
                     0605  1198 ;++
                     0605  1199 ; SBI alert, fault a i error are handled here
                     0605  1200 ; All interrupts cause a full SBIA log
                     0605  1201 ; SBI fail is treated like power fail and is handled elsewhere.
                     0605  1202 ;
                     0605  1203 ; Stack on entry:
                     0605  1204 ;       pointer to SBIA base address
                     0605  1205 ;       PC,PSL pair
                     0605  1206 ;--
                     0605  1207          .ALIGN  LONG
                     0608  1208 EXE$INT58::                             ; SBI ALERT
                     0608  1209 EXE$INT5C::                             ; SBI FAULT
     107F 8F    BB   0608  1210          PUSHR   #^M<R0,R1,R2,R3,R4,R5,R6,AP>
  53   20 BE    D0   060C  1211          MOVL    @<8*4>(SP),R3           ; SBIA base address
51   5E   24    C1   0610  1212          ADDL3   #<9*4>,SP,R1            ; Point to PC,PSL
                     0614  1213 SBI_RECOV:
        009E    30   0614  1214          BSBW    LOGSBI                  ; Go log it
     107F 8F    BA   0617  1215          POPR    #^M<R0,R1,R2,R3,R4,R5,R6,AP>
     5E   04    C0   061B  1216          ADDL    #4,SP                   ; Get rid of the base address
                02   061E  1217          REI
                     061F  1218 ;
                     061F  1219 ;
                     061F  1220          .ALIGN  LONG
                     0620  1221 EXE$INT60::                             ; SBIA ERROR
     107F 8F    BB   0620  1222          PUSHR   #^M<R0,R1,R2,R3,R4,R5,R6,AP>
  53   20 BE    D0   0624  1223          MOVL    @<8*4>(SP),R3           ; SBIA base address
51   5E   24    C1   0628  1224          ADDL3   #<9*4>,SP,R1            ; Point to PC,PSL
        12      E1   062C  1225          BBC     #SBIA$V_IME,-           ; Is it an unrecoverable internal error
     E3 08 A3        062E  1226                  SBIA$L_SUMRY(R3),SBI_RECOV ; No
        0081    30   0631  1227          BSBW    LOGSBI                  ; Go log it
                     0634  1228          BUG_CHECK SBIAERROR,FATAL
```

```
                        0638  1230          .SBTTL  Logging routines for machine checks
                        0638  1231 ;++
                        0638  1232 ; LOG_MCHECK -- format inputs to LOGGER
                        0638  1233 ;
                        0638  1234 ;
                        0638  1235 ; IMPLICIT INPUTS:
                        0638  1236 ;       (AP): points to machine check log on stack
                        0638  1237 ;
                        0638  1238 ; OUTPUTS:
                        0638  1239 ;       Error is formatted and logged in system error log.
                        0638  1240 ;       If no error log buffer, return with error status in R0.
                        0638  1241 ;       R0-R6 destroyed.
                        0638  1242 ;--
                        0638  1243
                        0638  1244 LOG_MCHECK:
                        0638  1245 ;
                        0638  1246 ; Test if a machine check recovery block that specifies no error
                        0638  1247 ; logging is in effect.
                        0638  1248 ;
         51    F8 AC    7D  0638  1249          MOVQ    -8(AP),R1               ; Get PC-PSL pointer and mask in R1,R2
   00000000'GF    16    63C  1250          JSB     G^EXE$MCHK_TEST         ; Logging inhibited?
         36 50    E8  0642  1251          BLBS    R0,20$                  ; Branch if YES - but go clear SBIA
                        0645  1252                                          ; Set up inputs to LOGGER.
   00000000'GF    D6  0645  1253          INCL    G^EXE$GL_MCHKERRS       ; Bump machine check error count
51 00000074 8F    D0  064B  1254          MOVL    #MCF790$L_PSL -         ; Size of machine check frame
                        0652  1255                  + EMB$B_MC_SUMCOD -     ; Add space for log header.
                        0652  1256                  + 4 ,R1                 ; SBIA summary
   00000000'GF    16  0652  1257          JSB     G^ERL$ALLOCEMB          ; Get error logging buffer.
         20 50    E9  0658  1258          BLBC    R0, 20$                 ; Br if failed to get buffer.
            52    DD  065B  1259          PUSHL   R2                      ; Save buffer addr on stack.
      04 A2    02    B0  065D  1260          MOVW    #EMB$K_MC, EMB$W_MC_ENTRY(R2) ; Set entry type.
         0060 8F    28  0661  1261          MOVC3   #MCF790$L_PSL,-         ; transfer info to log
            04 AC        0665  1262                  MCF790$L_EHSR(AP),-     ; do not include the size word
            10 A2        0667  1263                  EMB$B_MC_SUMCOD(R2)
               06    E1  0669  1264          BBC     #MCF790$V_SBIA_ERR,-    ; Should the SBIA error summary be added
      05 04 AC           066B  1265                  MCF790$B_MCHK_CODE(AP),15$
63    00E0'CF    D0  066E  1266          MOVL    W^SBIA_ERR_SUM,(R3)     ; Add summary at end of MOVC3 transfer
               04    BA  0673  1267 15$:    POPR    #^M<R2>                 ; Retreive buffer address.
   00000000'GF    16  0675  1268          JSB     G^ERL$RELEASEMB         ; Give buffer to logger.
                        067B  1269 ;
                        067B  1270 ; Now see if we need to log the SBIA data
                        067B  1271 ;
               05    E1  067B  1272 20$:    BBC     #MCF790$V_SBIA,-        ; Is the logging flag set
         12 04 AC           067D  1273                  MCF790$B_MCHK_CODE(AP),40$
               10    EF  0680  1274          EXTZV   #MCF790$V_AB_ADPT,-     ; relative IOA number
50    38 AC    02        0682  1275                  #MCF790$S_AB_ADPT, -
                        0686  1276                  MCF790$L_MSTAT1(AP), R0
53    0000'CF40    D0  0686  1277          MOVL    W^ABUS_VA[R0],R3        ; base address of SBIA
      51    F8 AC    7D  068C  1278          MOVQ    -8(AP),R1               ; Get PC-PSL pointer and mask in R1,R2
               23    11  0690  1279          BRB     LOGSBI                  ; Go to the SBIA logger
                        0692  1280
               05    0692  1281 40$:    RSB
```

```
                     0693  1283          .SBTTL  Logging routines for use by adaptor error routines
                     0693  1284 ;++
                     0693  1285 ; MCHK$GL_LOG -- format inputs to LOGGER
                     0693  1286 ;
                     0693  1287 ; INPUTS:
                     0693  1288 ;       R3: error type code
                     0693  1289 ;       R4: length of frame
                     0693  1290 ;       R5: address of frame
                     0693  1291 ;
                     0693  1292 ;
                     0693  1293 ; OUTPUTS:
                     0693  1294 ;       Error is formatted and logged in system error log.
                     0693  1295 ;       If no error log buffer, return with error status in R0.
                     0693  1296 ;       R0-R5 destroyed.
                     0693  1297 ;--
                     0693  1298
                     0693  1299 MCHK$GL_LOG::
     51    54    10  C1  0693  1300          ADDL3   #EMB$B_MC_SUMCOD,R4,R1  ; Space for log header
      00000000'GF    16  0697  1301          JSB     G^ERL$ALLOCEMB          ; Get a buffer
            14  50   E9  069D  1302          BLBC    R0,10$                  ; No space - give up
                52   DD  06A0  1303          PUSHL   R2                      ; Save address of buffer
        04  A2  53   B0  06A2  1304          MOVW    R3,EMB$W_MC_ENTRY(R2)   ; Rason code
  10    A2  65  54   28  06A6  1305          MOVC3   R4,(R5),EMB$B_MC_SUMCOD(R2) ; Move the frame
            52  8E   D0  06AB  1306          MOVL    (SP)+,R2                ; Get back the address
      00000000'GF    16  06AE  1307          JSB     G^ERL$RELEASEMB         ; Give buffer to logger
                     05  06B4  1308 10$:     RSB
```

```
                                06B5  1310          .SBTTL  SBIA logging
                                06B5  1311 ;++
                                06B5  1312 ; This routine is used to log SBIA (IOA) and SBI registers and related
                                06B5  1313 ; information.
                                06B5  1314 ;
                                06B5  1315 ; INPUTS:
                                06B5  1316 ;       R1 -> Address of PC, PSL
                                06B5  1317 ;       R3 -> Base adress of IOA registers
                                06B5  1318 ;--
                    52   01    D0  06B5  1319 LOGSBI: MOVL    #MCHK$M_LOG,R2          ; Mask to test if logging is enabled
          00000000'GF    16    06B8  1320          JSB     G^EXE$MCHK_TEST         ; Enabled?
                03 50    E9    06BE  1321          BLBC    R0,10$                  ; yes
                  00C9   31    06C1  1322 5$:      BRW     60$                     ; go clear errors
                55   51    D0  06C4  1323 10$:     MOVL    R1,R5                   ; Save address of PC, PSL
                                06C7  1324 ;                    adapters   silo     registers     overhead
                       000000E4 06C7  1325          LOG_SIZE =  <16*4> + <16*4>  +  <21*4>  +  EMB$B_MC_SUMCOD
          51    000000E4 8F    D0  06C7  1326          MOVL    #LOG_SIZE,R1            ; Need this much for error log buffer
                00000000'GF    16  06CE  1327          JSB     G^ERL$ALLOCEMB          ; Request it
                      EA 50    E9  06D4  1328          BLBC    R0,5$                   ; Can't get it - give up quietly
                 04 A2   0D    B0  06D7  1329          MOVW    #EMB$K_SBIA, EMB$W_MC_ENTRY(R2) ; Entry type code
     54   52    000000E4 8F    C1  06DB  1330          ADDL3    #LOG_SIZE,R2,R4        ; Address of end of error log buffer
                   74   65    7D  06E3  1331          MOVQ    (R5),-(R4)              ; PC, PSL
          56    00000000'GF    D0  06E6  1332          MOVL    G^EXE$GL_CONFREGL,R6    ; Array of NEXUS device type codes
          55    00000000'GF    D0  06ED  1333          MOVL    G^MMG$GL_SBICONF,R5     ; Array of adapter VA's
                        50    D4  06F4  1334          CLRL    R0                      ; Start with the first SBIA
     00000000'EF40      53    D1  06F6  1335 15$:     CMPL    R3,ABUS_VA[R0]          ; Is it this one?
                        08    13  06FE  1336          BEQL    20$
                  F2 50   04    F2  0700  1337          AOBLSS  #4,R0,15$
                                0704  1338          BUG_CHECK SBIAERROR,FATAL       ; No match
     50   00000000'EF40   9A    0708  1339 20$:     MOVZBL  ABUS_INDEX[R0],R0       ; Index in arrays for this SBIA
           50   50   02    78  0710  1340          ASHL    #2,R0,R0                ; Get longword offset
                56   50    C0  0714  1341          ADDL    R0,R6                   ; EXE$GL_CONFREGL
                55   50    C0  0717  1342          ADDL    R0,R5                   ; MMG$GL_SBICONF
                50   0F    D0  071A  1343          MOVL    #15,R0                  ; Index of last possible item on SBI
                        74    D4  071D  1344 30$:     CLRL    -(R4)                   ; Assume no adaptor here
                 51   6540    D0  071F  1345          MOVL    (R5)[R0],R1             ; Get VA of controller/adapter
                        08    18  0723  1346          BGEQ    40$                     ; GEQ implies no valid system VA.
                      6640    D5  0725  1347          TSTL    (R6)[R0]                ; Test adapter type
                        03    13  0728  1348          BEQL    40$                     ; If eql, no adaptor here
                   64   61    D0  072A  1349          MOVL    (R1),(R4)               ; Store adaptor CSR0 on stack
                      ED 50    F4  072D  1350 40$:     SOBGEQ  R0,30$                  ; Loop thru all possible 16
                50   0F    D0  0730  1351          MOVL    #15,R0                  ; Set up count of times to read silo
              74   30 A3    D0  0733  1352 50$:     MOVL    SBIA$L_SBISILO(R3),-(R4) ; Save information for error logger
                   F9 50    F4  0737  1353          SOBGEQ  R0,50$                  ; Loop thru all 16
              74   38 A3    D0  073A  1354          MOVL    SBIA$L_TMOADDRS(R3),-(R4) ; SBI timeout address
              74   34 A3    D0  073E  1355          MOVL    SBIA$L_SBIERR(R3),-(R4) ; SBI error register
              74   44 A3    D0  0742  1356          MOVL    SBIA$L_MAINT(R3),-(R4)  ; SBI maintenance register
              74   40 A3    D0  0746  1357          MOVL    SBIA$L_SILOCMP(R3),-(R4); SBI silo comparator
              74   3C A3    D0  074A  1358          MOVL    SBIA$L_SBISTS(R3),-(R4) ; SBI fault/status register
                 74   63    D0  074E  1359          MOVL    SBIA$L_CR(R3),-(R4)     ; IOA_CF (configuration register)
              74   04 A3    D0  0751  1360          MOVL    SBIA$L_CSR(R3),-(R4)    ; IOA_CS (control/status register)
              74   08 A3    D0  0755  1361          MOVL    SBIA$L_SUMRY(R3),-(R4)  ; IOA_ES (error summary register)
              74   0C A3    D0  0759  1362          MOVL    SBIA$L_DIAGNOS(R3),-(R4); IOA_DC (diagnostic control register)
              74   10 A3    D0  075D  1363          MOVL    SBIA$L_DMAICA(R3),-(R4) ; DMAI cmd/address register
              74   14 A3    D0  0761  1364          MOVL    SBIA$L_DMAIID(R3),-(R4) ; DMAI ID register
              74   18 A3    D0  0765  1365          MOVL    SBIA$L_DMAACA(R3),-(R4) ; DMAA cmd/address register
              74   1C A3    D0  0769  1366          MOVL    SBIA$L_DMAAID(R3),-(R4) ; DMAA ID register
```

```
                74    20 A3    D0  076D  1367          MOVL    SBIA$L_DMABCA(R3),-(R4) ; DMAB cmd/address register
                74    24 A3    D0  0771  1368          MOVL    SBIA$L_DMABID(R3),-(R4) ; DMAB ID register
                74    28 A3    D0  0775  1369          MOVL    SBIA$L_DMACCA(R3),-(R4) ; DMAC cmd/address register
                74    2C A3    D0  0779  1370          MOVL    SBIA$L_DMACID(R3),-(R4) ; DMAC ID register
                   74    53    D0  077D  1371          MOVL    R3,-(R4)                ; IOA base address
          74    000000D4 8F    D0  0780  1372          MOVL    #LOG_SIZE-EMB$B_MC_SUMCOD,-(R4) ; Frame size
                00000000'GF    16  0787  1373          JSB     G^ER[$RELEASEMB         ; Give it to the error logger
             08 A3    08 A3    C8  078D  1374 60$:     BISL    SBIA$L_SUMRY(R3),SBIA$L_SUMRY(R3) ; clear all the error locks
       34 A3    00001000 8F    C8  0792  1375          BISL    #SBIA$M_CTO,SBIA$L_SBIERR(R3)   ; and the CPU timeout bit
       3C A3    00080000 8F    C8  079A  1376          BISL    #SBIA$M_FLTLA,SBIA$L_SBISTS(R3) ; and the fault latch
                               05  07A2  1377 70$:     RSB
```

```
                        07A3  1379          .SBTTL  Single Bit (CRD) error handling
                        07A3  1380 ;++
                        07A3  1381 ; This routine logs single bit errors.
                        07A3  1382 ; 16 errors are accumulated before an error log entry is made.
                        07A3  1383 ; 3 errors in 1 second causes reporting to be turned off for 5 minutes
                        07A3  1384 ;
                        07A3  1385 ; Due to a hardware bug, we must never run with CRD interrupts turned off.
                        07A3  1386 ; Doing so can cause us to miss double-bit errors.  Instead, we will only
                        07A3  1387 ; disable logging of the errors.
                        07A3  1388 ;--
                        07A3  1389          .ALIGN  LONG
                        07A4  1390 EXE$INT54::
        70 00E8'CF   E8 07A4  1391          BLBS    W^CRD_FLAGS,20$         ; Is logging turned off
              07   BB 07A9  1392          PUSHR   #^M<R0,R1,R2>
         0068'CF   D6 07AB  1393          INCL    W^CRD_TOTAL             ; Count the error
     50  01AC'CF   D0 07AF  1394          MOVL    W^CRD_NEXT,R0           ; Next slot in CRD log buffer
              0A   12 07B4  1395          BNEQ    10$                     ; Is it initialized
     50  00EC'CF   9E 07B6  1396          MOVAB   W^CRD_FIRST_ENTRY,R0    ; First slot in CRD log buffer
     01AC'CF   50   D0 07BB  1397          MOVL    R0,W^CRD_NEXT           ; Initialize it
 0000004E 8F   27   DA 07C0  1398 10$:     MTPR    #^X27,#PR790$_LSPA      ; MDECC code to scratchpad address
 80   0000004F 8F   DB 07C7  1399          MFPR    #PR790$_RSPD,(R0)+      ; Save MDECC
 0000004E 8F   2A   DA 07CE  1400          MTPR    #^X2A,#PR790$_LSPA      ; MEAR code to scratchpad address
 80   0000004F 8F   DB 07D5  1401          MFPR    #PR790$_RSPD,(R0)+      ; Save MEAR
 0000004E 8F   26   DA 07DC  1402          MTPR    #^X26,#PR790$_LSPA      ; MSTAT2 code to scratchpad address
 80   0000004F 8F   DB 07E3  1403          MFPR    #PR790$_RSPD,(R0)+      ; Save MSTAT2
              51   1B DB 07EA  1404          MFPR    #PR790$_TODR,R1         ; Current time - 10 millisecond units
 52   51      0064'CF   C3 07ED  1405          SUBL3   W^CRD_OLD2,R1,R2        ; Time for three errors
  00000064 8F   52   D1 07F3  1406          CMPL    R2,#CRD_THRESHOLD       ; Too quick
              6B   1F 07FA  1407          BLSSU   CRD_OFF                 ; Yes
0064'CF   0060'CF   D0 07FC  1408          MOVL    W^CRD_OLD1,W^CRD_OLD2   ; Save new times
     0060'CF   51   D0 0803  1409          MOVL    R1,W^CRD_OLD1
     51  01AC'CF   9E 0808  1410          MOVAB   W^CRD_BUF_END,R1        ; End of buffer
         51   50   D1 080D  1411          CMPL    R0,R1                   ; Is buffer full?
              08   1E 0810  1412          BGEQU   CRD_LOG                 ; Yes
     01AC'CF   50   D0 0812  1413          MOVL    R0,W^CRD_NEXT           ; Next buffer slot
              07   BA 0817  1414          POPR    #^M<R0,R1,R2>
              02   0819  1415 20$:     REI
                   081A  1416 ;
                   081A  1417 ; Log the current buffer and clear it for reuse.
                   081A  1418 ;
                   081A  1419 CRD_LOG:
              38   BB 081A  1420          PUSHR   #^M<R3,R4,R5>
         52   01   D0 081C  1421          MOVL    #MCHK$M_LOG,R2          ; Mask to test if logging is enabled
      00000000'GF   16 081F  1422          JSB     G^EXE$MCHK_TEST         ; Enabled?
            28   50 E8 0825  1423          BLBS    R0,10$                  ; No - just go clear buffer
 51   000000D8 8F   D0 0828  1424          MOVL    #CRD_LOG_SIZE + EMB$B_MC_SUMCOD,R1 ; Error log buffer size
      00000000'GF   16 082F  1425          JSB     G^ERL$ALLOCEMB          ; Request it
            18   50 E9 0835  1426          BLBC    R0,10$                  ; Can't get it - just clear the buffer
 00000004'EF   0E   B0 0838  1427          MOVW    #EMB$K_CRD, EMB$W_MC_ENTRY ; Entry type code
              52   DD 083F  1428          PUSHL   R2
     00E4'CF   51   28 0841  1429          MOVC3   R1,W^CRD_BUFFER,-       ; Move log to buffer
            10 A2      0846  1430                  EMB$B_MC_SUMCOD(R2)
              04   BA 0848  1431          POPR    #^M<R2>
      00000000'GF   16 084A  1432          JSB     G^ERL$RELEASEMB
     50  00EC'CF   9E 0850  1433 10$:     MOVAB   W^CRD_FIRST_ENTRY,R0    ; First slot in CRD log buffer
     01AC'CF   50   D0 0855  1434          MOVL    R0,W^CRD_NEXT           ; Start here next time
     00   60   00   2C 085A  1435          MOVC5   #0,(R0),#0,-            ; Clear the buffer
```

```
                                                H 15
00C0'8F          085E  1436                    #CRD_BUF_END-CRD_FIRST_ENTRY,-
      60         0861  1437                    (R0)
      38    BA   0862  1438          POPR      #^M<R3,R4,R5>
      07    BA   0864  1439          POPR      #^M<R0,R1,R2>
      02         0866  1440          REI
                 0867  1441 ;
                 0867  1442 ; Turn off CRD error reporting for 5 minutes
                 0867  1443 ; Due to hardware bug, we will not turn off interrupts, only logging.
                 0867  1444 ;
                 0867  1445 CRD_OFF:
                 0867  1446 ;         MFPR      #PR790$_MERG,R0          ; Get current error reporting state
                 0867  1447 ;         BBSS      #MERG$V_INHCRD,R0,10$    ; Turn off CRD reporting
                 0867  1448 ;10$:     MTPR      R0,#PR790$_MERG
00E8'CF    01    D0    0867  1449               MOVL      #1,W^CRD_FLAGS           ; Indicate CRD reporting turned off
      AC    11    086C  1450               BRB       CRD_LOG                  ; Go log it
```

```
                                086E  1452          .SBTTL  CRD reenable timer routine
                                086E  1453 ;++
                                086E  1454 ; Routine called by system clock routine.
                                086E  1455 ; If CRD reporting is disabled and the timer has expired, reporting is
                                086E  1456 ; reenabled if allowed by the SYSGEN parameter.
                                086E  1457 ;--
                                086E  1458 ECC$REENABLE::
                          50    DD  086E  1459          PUSHL   R0
                    1F 00E8'CF  E9  0870  1460          BLBC    W^CRD_FLAGS,20$         ; Return if reporting is not disabled
                       006C'CF  D7  0875  1461          DECL    W^CRD_TIMER
                            19  14  0879  1462          BGTR    20$                     ; Timer not expired yet
                0000012C 8F     D0  087B  1463          MOVL    #CRD_REENAB_TIME,-      ; Reset the timer
                       006C'CF      0881  1464                  W^CRD_TIMER
                00000000'8F     E1  0884  1465          BBC     #EXE$V_CRDENABL,-       ; Check SYSGEN parameter
             04 00000000'GF         088A  1466                  G^EXE$GL_FLAGS, 20$     ; Br if CRD's not wanted
                                0890  1467 ;
                                0890  1468 ; Due to hardware bug, we did not turn off interrupts, only logging.
                                0890  1469 ;
                                0890  1470 ;       MFPR    #PR790$_MERG,R0         ; Get current error reporting state
                                0890  1471 ;       BBCC    #MERG$V_INHCRD,R0,10$   ; Turn on CRD reporting
                                0890  1472 ;10$:   MTPR    R0,#PR790$_MERG
                       00E8'CF  D4  0890  1473          CLRL    W^CRD_FLAGS             ; Indicate CRD reporting turned on
                            01  BA  0894  1474 20$:     POPR    #^M<R0>
                       00B0'CF  D7  0896  1475          DECL    W^KEEPALIVE_TIMER       ; Decrement console keepalive timer.
                            01  15  089A  1476          BLEQ    30$                     ; Branch if it has expired.
                                05  089C  1477          RSB                             ; Else we're done.
                                089D  1478 30$:                                         ; Fall into CON$KEEPALIVE...
```

```
                                089D  1480          .SBTTL  CON$KEEPALIVE
                                089D  1481 ;++
                                089D  1482 ; CON$KEEPALIVE - DETERMINE IF VENUS CONSOLE SOFTWARE IS STILL FUNCTIONING
                                089D  1483 ;
                                089D  1484 ; FUNCTIONAL DESCRIPTION:
                                089D  1485 ;
                                089D  1486 ;       THIS ROUTINE IS CALLED PERIODICALLY TO DETERMINE IF THE VENUS CONSOLE
                                089D  1487 ;       SOFTWARE IS STILL FUNCTIONING.  IT READS THE TIME-OF-DAY PROCESSOR
                                089D  1488 ;       REGISTER, WHICH IS MAINTAINED BY THE CONSOLE SOFTWARE.  IF IT IS
                                089D  1489 ;       NOT BEING UPDATED, THEN THE CONSOLE IS REBOOTED.
                                089D  1490 ;
                                089D  1491 ; INPUTS:
                                089D  1492 ;       R3      - ADDRESS OF CRB
                                089D  1493 ;
                                089D  1494 ; IMPLICIT INPUTS:
                                089D  1495 ;       THIS ROUTINE IS EXECUTED ONCE EVERY 'KEEPALIVE_TIME' SECONDS.
                                089D  1496 ;
                                089D  1497 ; IMPLICIT OUTPUTS:
                                089D  1498 ;       THE VENUS CONSOLE MAY BE RE-BOOTED.
                                089D  1499 ;
                                089D  1500 ;--
                                089D  1501
                                089D  1502 CON$KEEPALIVE:
                    50    DD    089D  1503          PUSHL   R0                      ; Save a register.
                  50    1B    DB  089F  1504          MFPR    #PR790$_TODR,R0         ; Read the TODR register.
          00B4'CF   50    D1    08A2  1505          CMPL    R0,W^TODR_VALUE         ; Has the value changed?
                    12    13    08A7  1506          BEQL    20$                     ; Branch if not.
          00B4'CF   50    D0    08A9  1507          MOVL    R0,W^TODR_VALUE         ; Save most recent value.
00B0'CF   0000005A  8F    D0    08AE  1508 10$:      MOVL    #KEEPALIVE_TIME, -      ; Re-set the timer so we are called
                                08B7  1509                  W^KEEPALIVE_TIMER       ; again.
                    50  8ED0    08B7  1510          POPL    R0                      ; Restore the register.
                        05      08BA  1511          RSB
                                08BB  1512 20$:
                                08BB  1513 ;
                                08BB  1514 ; Reboot the console, and log that we did it.
                                08BB  1515 ;
00000048 8F     01      DA      08BB  1516          MTPR    #1,#PR790$_CRBT         ; Reboot the console.
                06      BB      08C2  1517          PUSHR   #^M<R1,R2>              ; Save some registers.
              51    11      D0  08C4  1518          MOVL    #EMB$C_HD_LENGTH+1,R1   ; Allocate a header plus one byte.
        00000000'GF     16      08C7  1519          JSB     G^ERL$ALLOCEMB          ; Allocate space in the errorlog buffer.
             EB 50      E9      08CD  1520          BLBC    R0,20$                  ; Branch if unable to allocate.
          04 A2     11      B0  08D0  1521          MOVW    #EMB$C_CRBT, -          ; Set entry type = console reboot.
                                08D4  1522                  EMB$W_RD_ENTRY(R2)      ;
              05 A2     94      08D4  1523          CLRB    EMB$W_HD_ENTRY+1(R2)    ; Set flag = reboot attempted.
        00000000'GF     16      08D7  1524          JSB     G^ER $RELEASEMB         ; Release the errorlog data.
                06      BA      08DD  1525          POPR    #^M<R1,R2>              ; Restore the registers.
                CD      11      08DF  1526          BRB     10$                     ; Join common exit code.
                                08E1  1527
                                08E1  1528          .END
```

```
ABORT_BITS                      000000DC R      02
ABUS_CYCLE                      000000D8 R      02
ABUS_INDEX                      ********   X    03
ABUS_TYPE                       ********   X    03
ABUS_VA                         ********   X    03
ADP$C_LINK                  =   00000004
ADP$L_UBASCB                =   00000044
ADP$W_ADPTYPE               =   0000000E
ATS_UBA                         ********   X    03
BAD_MCHK                        00000062 R      03
BAD_MEM                         00000313 R      03
BUG$_BADMCKCOD                  ********   X    03
BUG$_MACHINECHK                 ********   X    03
BUG$_RDSNONRES                  ********   X    03
BUG$_SBIAERROR                  ********   X    03
BUGCHECK                        000000F2 R      03
BUGCHECK_NOLOG                  000000F5 R      03
BUGCHECK_POP                    000000EF R      03
CACHE_ERR                       000004A1 R      03
CACHE_MSG                       000001E8 R      02
CACHE_OFF                       0000053E R      03
CACHE_THRESHOLD             =   0000000A
CACHE_TOTAL                     00000050 R      02
CHECK_MBOX_1D                   0000007B R      03
CONSKEEPALIVE                   0000089D R      03
CRD_BUFFER                      000000E4 R      02
CRD_BUF_END                     000001AC R      02
CRD_FIRST_ENTRY                 000000EC R      02
CRD_FLAGS                       000000E8 R      02
CRD_LOG                         0000081A R      03
CRD_LOG_SIZE                =   000000C8
CRD_NEXT                        000001AC R      02
CRD_OFF                         00000867 R      03
CRD_OLD1                        00000060 R      02
CRD_OLD2                        00000064 R      02
CRD_REENAB_TIME             =   0000012C
CRD_THRESHOLD               =   00000064
CRD_TIMER                       0000006C R      02
CRD_TOTAL                       00000068 R      02
CSH_A_OLD1                      00000040 R      02
CSH_A_OLD2                      00000044 R      02
CSH_B_OLD1                      00000048 R      02
CSH_B_OLD2                      0000004C R      02
CSWPSM_COENA                =   00000001
CSWPSM_C1ENA                =   00000002
CSWPSM_VAL                  =   00000004
EBOX_OLD1                       00000018 R      02
EBOX_OLD2                       0000001C R      02
EBOX_SERV                       000002E7 R      03
EBOX_THRESHOLD              =   0000000A
EBOX_TOTAL                      00000020 R      02
ECC$REENABLE                    0000086E RG     03
EMB$B_MC_SUMCOD             =   00000010
EMB$C_CRBT                  =   00000011
EMB$C_HD_LENGTH             =   00000010
EMB$K_CRD                   =   0000000E
EMB$K_MC                    =   00000002
EMB$K_SBIA                  =   0000000D
EMB$W_HD_ENTRY              =   00000004
EMB$W_MC_ENTRY              =   00000004
ERL$ALLOCEMB                    ********   X    03
ERL$RELEASEMB                   ********   X    03
EXE$AB_MEMERR                   00000070 RG     02
EXE$GL_CONFREGL                 ********   X    03
EXE$GL_FLAGS                    ********   X    03
EXE$GL_MCHKERRS                 ********   X    03
EXE$INT54                       000007A4 RG     03
EXE$INT58                       00000608 RG     03
EXE$INT5C                       00000608 RG     03
EXE$INT60                       00000620 RG     03
EXE$MCHECK                      ********   X    03
EXE$MCHK                        00000000 RG     03
EXE$MCHK_BUGCHK                 ********   X    03
EXE$MCHK_ERRCNT                 00000000 RG     02
EXE$MCHK_TEST                   ********   X    03
EXE$V_CRDENABL                  ********   X    03
E_ERR                           00000073 R      03
FBOX_MSG                        000001B0 R      02
FBOX_OLD1                       0000000C R      02
FBOX_OLD2                       00000010 R      02
FBOX_SERV                       000002A9 R      03
FBOX_THRESHOLD              =   0000000A
FBOX_TOTAL                      00000014 R      02
F_ERR                           0000006E R      03
IBOX_OLD1                       00000000 R      02
IBOX_OLD2                       00000004 R      02
IBOX_SERV                       0000010A R      03
IBOX_THRESHOLD              =   0000000A
IBOX_TOTAL                      00000008 R      02
IOC$BROADCAST                   ********   X    03
IOC$GL_ADPLIST                  ********   X    03
IPL$_ASTDEL                 =   00000002
I_ERR                           00000078 R      03
KEEPALIVE_TIME              =   0000005A
KEEPALIVE_TIMER                 000000B0 R      02
LOGSBI                          000006B5 R      03
LOG_MCHECK                      00000638 R      03
LOG_SIZE                    =   000000E4
MBOX_1D_CPU                     0000045E R      03
MBOX_1D_DMA                     00000575 R      03
MBOX_1D_OLD1                    00000054 R      02
MBOX_1D_OLD2                    00000058 R      02
MBOX_1D_SERV                    00000431 R      03
MBOX_1D_THRESHOLD           =   0000000A
MBOX_1D_TOTAL                   0000005C R      02
MBOX_FE_OLD                     00000024 R      02
MBOX_FE_PC                      00000028 R      02
MBOX_FE_PHY_ADR                 00000030 R      02
MBOX_FE_SERV                    0000012C R      03
MBOX_FE_THRESHOLD           =   00000002
MBOX_FE_TOTAL                   0000002C R      02
MCF790$B_MCHK_CODE          =   00000004
MCF790$C_ABUS               =   00000008
MCF790$C_ABUS_REFL          =   0000000F
```

```
MCF790$C_ABUS_WRT               = 00000004
MCF790$C_CP_READ                = 0000000E
MCF790$C_CP_WRT                 = 0000000D
MCF790$C_EBOX                   = 00000002
MCF790$C_EBOX_PORT              = 00000002
MCF790$C_FBOX                   = 00000001
MCF790$C_IBOX                   = 00000003
MCF790$C_MBOX_FE                = 00000004
MCF790$C_NOP                    = 00000000
MCF790$C_OP_PORT                = 00000001
MCF790$C_WRITE_REG              = 00000002
MCF790$L_EBCS                   = 0000000C
MCF790$L_EHSR                   = 00000004
MCF790$L_ESASAV                 = 0000002C
MCF790$L_IVASAV                 = 00000024
MCF790$L_MDECC                  = 0000004C
MCF790$L_MEAR                   = 00000050
MCF790$L_MSTAT1                 = 00000038
MCF790$L_MSTAT2                 = 0000003C
MCF790$L_PC                     = 0000005C
MCF790$L_PSL                    = 00000060
MCF790$L_VIBASAV                = 00000028
MCF790$M_ABORTS                 = 0000001E
MCF790$M_AB_ADR_PE              = 00080000
MCF790$M_AB_CM_PE               = 00100000
MCF790$M_AB_DAT_PE              = 00200000
MCF790$M_AUTO_SHUT              = 00020000
MCF790$M_BAD_DATA               = 00400000
MCF790$M_CPR_PE_A               = 00400000
MCF790$M_CPR_PE_B               = 00800000
MCF790$M_CSH_TAG_PE             = 00000040
MCF790$M_CSH_TAG_W              = 00000020
MCF790$M_DBL_BIT                = 00100000
MCF790$M_ECS_PE                 = 00000800
MCF790$M_EDP_PE                 = 00000200
MCF790$M_EMCR_PE                = 00001000
MCF790$M_IO_RD                  = 00000002
MCF790$M_MEM_WRT                = 00000004
MCF790$M_RSRC_REM               = 00000010
MCF790$M_SBIA                   = 00000020
MCF790$M_SBIA_ERR               = 00000040
MCF790$M_TB_A_PE                = 00000200
MCF790$M_TB_B_PE                = 00000400
MCF790$M_TB_TAG_PE              = 00000100
MCF790$M_TB_VAL_PE              = 00000800
MCF790$M_USTK_PE                = 00000400
MCF790$S_AB_ADPT                = 00000002
MCF790$S_CYCLE_TYP              = 00000004
MCF790$S_DEST_CP                = 00000002
MCF790$V_ABORTS                 = 00000001
MCF790$V_AB_ADPT                = 00000010
MCF790$V_AB_BAD_DAT             = 0000000E
MCF790$V_ADR_PE                 = 00000013
MCF790$V_BAD_DATA               = 00000016
MCF790$V_CP_TO_BUF              = 00000002
MCF790$V_CSH_DAT_BW             = 00000000
MCF790$V_CSH_DAT_NBW            = 00000003
MCF790$V_CSH_ERR                = 00000002
MCF790$V_CSH_TAG_PE             = 00000006
MCF790$V_CSH_W                  = 00000004
MCF790$V_CYCLE_TYP              = 0000001A
MCF790$V_DBL_BIT                = 00000014
MCF790$V_DEST_CP                = 0000001E
MCF790$V_EDP_PE                 = 00000009
MCF790$V_FBOX                   = 0000001C
MCF790$V_IBOX_ERR               = 0000000D
MCF790$V_MBOX_1D                = 00000007
MCF790$V_MBOX_FE                = 0000000F
MCF790$V_MBOX_INT               = 0000000E
MCF790$V_MUL_ERR                = 00000007
MCF790$V_NXM                    = 00000003
MCF790$V_SBIA                   = 00000005
MCF790$V_SBIA_ERR               = 00000006
MCHK$GL_LOG                       00000693 RG     03
MCHK$M_LOG                      = 00000001
MCHK$M_MCK                      = 00000002
MCHK$M_NEXM                     = 00000004
MCHK_ABORT                      = 00000001
MCHK_EXIT                         00000097 R      03
MISC_MBOX_1D                      0000051C R      03
MMG$AL_SYSPCB                     ********    X   03
MMG$DELWSLEX                      ********    X   03
MMG$GL_MAXPFN                     ********    X   03
MMG$GL_SBICONF                    ********    X   03
MMG$GL_SPTBASE                    ********    X   03
MMG$GW_BIGPFN                     ********    X   03
MMG$REFCNTNEG                     ********    X   03
MMG$RELPFN                        ********    X   03
M_ERR                             00000069 R      03
OPA$UCBO                          ********    X   03
PCB$L_PHD                       = 0000006C
PFN$AB_STATE                      ********    X   03
PFN$AB_TYPE                       ********    X   03
PFN$AL_PTE                        ********    X   03
PFN$AW_REFCNT                     ********    X   03
PFN$AX_WSLX                       ********    X   03
PFN$C_PROCESS                   = 00000000
PFN$C_SYSTEM                    = 00000001
PFN$M_BADPAG                    = 00000020
PFN$M_DELCON                    = 00000010
PFN$M_MODIFY                    = 00000080
PFN$S_PAGTYP                    = 00000003
PFN$V_PAGTYP                    = 00000000
PHD$L_PCB                       = 00000078
PR$_KSP                         = 00000000
PR$_P0BR                        = 00000008
PR$_P1BR                        = 0000000A
PR$_TBIA                        = 00000039
PR790$_ACCS                     = 00000028
PR790$_CRBT                     = 00000048
PR790$_CSWP                     = 00000042
PR790$_EHSR                     = 0000004A
PR790$_LSPA                     = 0000004E
PR790$_RSPD                     = 0000004F
```

```
PR790$_TODR                     =  0000001B
PSL$S_CURMOD                    =  00000002
PSL$S_IPL                       =  00000005
PSL$V_CURMOD                    =  00000018
PSL$V_IPL                       =  00000010
PSL$V_IS                        =  0000001A
PSL$V_PRVMOD                    =  00000016
PTE$V_MODIFY                    =  0000001A
PTE$V_VALID                     =  0000001F
PTE$V_WINDOW                    =  00000015
RDSNONRES                          00000416 R      03
REFLECT_MCHK                       000000B7 R      03
SBIA$L_CR                          0000C000
SBIA$L_CSR                         00000004
SBIA$L_DIAGNOS                     0000000C
SBIA$L_DMAACA                      00000018
SBIA$L_DMAAID                      0000001C
SBIA$L_DMABCA                      00000020
SBIA$L_DMABID                      00000024
SBIA$L_DMACCA                      00000028
SBIA$L_DMACID                      0000002C
SBIA$L_DMAICA                      00000010
SBIA$L_DMAIID                      00000014
SBIA$L_MAINT                       00000044
SBIA$L_SBIERR                      00000034
SBIA$L_SBIQC                       0000004C
SBIA$L_SBISILO                     00000030
SBIA$L_SBISTS                      0000003C
SBIA$L_SILOCMP                     00000040
SBIA$L_SUMRY                       00000008
SBIA$L_TMOADDRS                    00000038
SBIA$L_UNJAM                       00000048
SBIA$M_ADP                      =  00400000
SBIA$M_CTO                      =  00001000
SBIA$M_FLTLA                    =  00080000
SBIA$M_RCP                      =  00200000
SBIA$V_BEL                      =  00000017
SBIA$V_CAE                      =  00000013
SBIA$V_CTO                      =  0000000C
SBIA$V_IME                      =  00000012
SBIA_ERR_SUM                       000000E0 R      02
SBI_INST                           000000B8 R      02
SBI_RECOV                          00000614 R      03
SWP$GL_BALBASE                     ******** X      03
SWP$GL_BSLOTSZ                     ******** X      03
TB_OLDT                            00000034 R      02
TB_OLD2                            00000038 R      02
TB_THRESHOLD                    =  0000000A
TB_TOTAL                           0000003C R      02
TODR_VALUE                         000000B4 R      02
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
ABS	00000050 (80.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
MCHK$DATA	0000022A (554.)	02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC QUAD
MCHK$CODE	000008E1 (2273.)	03 (3.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC QUAD

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	29	00:00:00.05	00:00:00.44
Command processing	111	00:00:00.45	00:00:04.35
Pass 1	431	00:00:11.17	00:00:42.48
Symbol table sort	0	00:00:01.62	00:00:07.22
Pass 2	280	00:00:02.99	00:00:11.83
Symbol table output	34	00:00:00.17	00:00:00.38
Psect synopsis output	1	00:00:00.01	00:00:00.01
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	888	00:00:16.47	00:01:06.72

The working set limit was 2100 pages.
93941 bytes (184 pages) of virtual memory were used to buffer the intermediate code.
There were 80 pages of symbol table space allocated to hold 1453 non-local and 96 local symbols.
1528 source lines were read in Pass 1, producing 23 object records in Pass 2.
34 pages of virtual memory were used to define 32 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[SYSLOA.OBJ]790DEF.MLB;1	5
_255DUA28:[SYS.OBJ]LIB.MLB;1	17
_255DUA28:[SYSLIB]STARLET.MLB;2	7
TOTALS (all libraries)	29

1542 GETS were required to define 29 macros.

There were no errors, warnings or information messages.

```
MACRO/LIS=LIS$:MCHECK790/OBJ=OBJ$:MCHECK790 MSRC$:MCHECK790/UPDATE=(ENH$:MCHECK790)+EXECML$/LIB+LIB$:790DEF/LIB
```

LIOSUB750
LIS

MCHECKUV1
LIS

MCHECK730
LIS

MCHECK750
LIS

LIOSUB730
LIS

LIOSUB790
LIS

MCHECK780
LIS

MCF790
LIS

MCHECK790
LIS

LIOSUB780
LIS

LIOSUBUV1
LIS