```
   SSSSSSSSSSSS  YYY         YYY     SSSSSSSSSSSS  LLL                 OOOOOOOOO        AAAAAAAAA
   SSSSSSSSSSSS  YYY         YYY     SSSSSSSSSSSS  LLL                 OOOOOOOOO        AAAAAAAAA
   SSSSSSSSSSSS  YYY         YYY     SSSSSSSSSSSS  LLL                 OOOOOOOOO        AAAAAAAAA
SSS              YYY         YYY  SSS              LLL              OOO         OOO  AAA         AAA
SSS              YYY         YYY  SSS              LLL              OOO         OOO  AAA         AAA
SSS              YYY         YYY  SSS              LLL              OOO         OOO  AAA         AAA
SSS                 YYY   YYY     SSS              LLL              OOO         OOO  AAA         AAA
SSS                 YYY   YYY     SSS              LLL              OOO         OOO  AAA         AAA
SSS                 YYY   YYY     SSS              LLL              OOO         OOO  AAA         AAA
   SSSSSSSSS           YYY           SSSSSSSSS     LLL              OOO         OOO  AAA         AAA
   SSSSSSSSS           YYY           SSSSSSSSS     LLL              OOO         OOO  AAA         AAA
   SSSSSSSSS           YYY           SSSSSSSSS     LLL              OOO         OOO  AAA         AAA
            SSS        YYY                    SSS  LLL              OOO         OOO  AAAAAAAAAAAAAAA
            SSS        YYY                    SSS  LLL              OOO         OOO  AAAAAAAAAAAAAAA
            SSS        YYY                    SSS  LLL              OOO         OOO  AAAAAAAAAAAAAAA
            SSS        YYY                    SSS  LLL              OOO         OOO  AAA         AAA
            SSS        YYY                    SSS  LLL              OOO         OOO  AAA         AAA
            SSS        YYY                    SSS  LLL              OOO         OOO  AAA         AAA
SSSSSSSSSSSS           YYY        SSSSSSSSSSSS     LLLLLLLLLLLLLLL     OOOOOOOOO     AAA         AAA
SSSSSSSSSSSS           YYY        SSSSSSSSSSSS     LLLLLLLLLLLLLLL     OOOOOOOOO     AAA         AAA
SSSSSSSSSSSS           YYY        SSSSSSSSSSSS     LLLLLLLLLLLLLLL     OOOOOOOOO     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  MM CC         HH        HH  EE         CC         KK    KK            77    99      99  00    0000
MM  MM  MM  MM CC         HH        HH  EE         CC         KK    KK            77    99      99  00    0000
MM    MM    MM CC         HHHHHHHHHH  EEEEEEEE     CC         KKKKKK            77      99999999  00  00  00
MM    MM    MM CC         HHHHHHHHHH  EEEEEEEE     CC         KKKKKK            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

MCH
V04

```
0000        .TITLE  MCHECK790 -- VENUS MACHINE CHECK
0000        .IDENT  'V04-004'
0000
0000  ;
0000  ;****************************************************************************
0000  ;*                                                                          *
0000  ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0000  ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
0000  ;*  ALL RIGHTS RESERVED.                                                    *
0000  ;*                                                                          *
0000  ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
0000  ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000  ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
0000  ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
0000  ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
0000  ;*  TRANSFERRED.                                                            *
0000  ;*                                                                          *
0000  ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
0000  ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000  ;*  CORPORATION.                                                            *
0000  ;*                                                                          *
0000  ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
0000  ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0000  ;*                                                                          *
0000  ;*                                                                          *
0000  ;****************************************************************************
0000  ;
0000
0000  ;++
0000  ; FACILITY: SYSLOA790 - loadable CPU-dependent code
0000  ;
0000  ; ABSTRACT:
0000  ;       This module contains routines to handle VAX 11/790 specific
0000  ;       machine check errors.
0000  ;
0000  ; ENVIRONMENT:
0000  ;       IPL = 31        Mode = KERNEL
0000  ;
0000  ; AUTHOR:  Wayne Cardoza        CREATION DATE: 2-Oct-1982
0000  ;
0000  ; MODIFIED BY:
0000  ;
0000  ;       V04-004 WMC0020         Wayne Cardoza           13-Sep-1984
0000  ;               Make sure we never run with CRD interrupts disabled.
0000  ;
0000  ;       V04-003 WMC0019         Wayne Cardoza           12-Sep-1984
0000  ;               Fix position accounting in CRD log.
0000  ;
0000  ;       V04-002 TCM0002         Trudy C. Matthews       11-Sep-1984
0000  ;               Make CON$KEEPALIVE's reference to ERL$RELEASEMB absolute.
0000  ;
0000  ;       V04-001 WMC0018         Wayne Cardoza           05-Sep-1984
0000  ;               Add MSTAT2 to the CRD log.
0000  ;
0000  ;       V03-023 WMC0017         Wayne Cardoza           23-Aug-1984
0000  ;               Fix a bad index mode.
0000  ;
```

```
0000  ;     V03-022 WMC0016         Wayne Cardoza           14-Aug-1984
0000  ;             CP-IO BUF doesn't always have a cycle type.
0000  ;
0000  ;     V03-021 WMC0015         Wayne Cardoza           06-Aug-1984
0000  ;             And more changes.
0000  ;
0000  ;     V03-020 TCM0001         Trudy C. Matthews       31-Jul-1984
0000  ;             Add routine CON$KEEPALIVE, which is called periodically to
0000  ;             determine if the VENUS console software is still functioning.
0000  ;             Also, change the CRD interrupt vector from ^X64 in the SCB to
0000  ;             ^X54.
0000  ;
0000  ;     V03-019 WMC0014         Wayne Cardoza           23-Jul-1984
0000  ;             Still more spec changes.
0000  ;
0000  ;     V03-018 WMC0013         Wayne Cardoza           12-Jul-1984
0000  ;             Many spec changes.
0000  ;
0000  ;     V03-017 WMC0012         Wayne Cardoza           15-May-1984
0000  ;             Space always reserved for SBIA error word.
0000  ;             Add PC-PSL to log message.
0000  ;
0000  ;     V03-016 WMC0011         Wayne Cardoza           30-Apr-1984
0000  ;             Add ADPDEF.
0000  ;
0000  ;     V03-015 WMC0010         Wayne Cardoza           16-Apr-1984
0000  ;             Ignore CP timeouts on BRVVR.
0000  ;
0000  ;     V03-014 RLRSBICONF      Robert L. Rappaport     22-Mar-1984
0000  ;             Test MMG$GL_SBICONF array elements for valid system
0000  ;             virtual address (high bit set) before using.  Also
0000  ;             correct bug introduced by CONFREGL fix.
0000  ;
0000  ;     V03-013 WMC0009         Wayne Cardoza           10-Mar-1984
0000  ;             Fix SBIA logging bugs.
0000  ;
0000  ;     V03-012 KPL0100         Peter Lieberwirth       26-Feb-1984
0000  ;             Use CONFREGL in lieu of CONFREG.
0000  ;
0000  ;     V03-011 WMC0008         Wayne Cardoza   29-Jan-1983
0000  ;             Clean up stack before bugcheck.
0000  ;
0000  ;     V03-010 WMC0007         Wayne Cardoza   01-Dec-1983
0000  ;             Turn CP-IO-BUF into NXM for EXE$MCHK_BUGCHK checks
0000  ;             MBOX doesn't always latch cycle type.
0000  ;
0000  ;     V03-009 WMC0006         Wayne Cardoza   20-Nov 1983
0000  ;             Fix transposed lines of code.
0000  ;
0000  ;     V03-008 TMK0001         Todd M. Katz            14-Nov-1983
0000  ;             Add a .TITLE so that the object file for this module
0000  ;             will be inserted into SYSLOA.OLB as MCHECK790 instead of
0000  ;             as .MAIN.
0000  ;
0000  ;     V03-007 WMC0005         Wayne Cardoza   02-Nov-1983
0000  ;             Fix IOC$BROADCAST calls.
0000  ;             Get adaptor type from ABUS_TYPE.
```

```
0000   ;
0000   ;       V03-006 KDM0053         Kathleen D. Morse       11-Jul-1983
0000   ;               Replace cpu-specific IPR references with the cpu-specific
0000   ;               symbols, defined by $PR790DEF.
0000   ;
0000   ;       V03-005 WMC0004         Wayne Cardoza   06-Apr-1983
0000   ;               Changes to bad page handling.
0000   ;
0000   ;       V03-004 WMC0003         Wayne Cardoza   22-Mar-1983
0000   ;               Misc minor fixes.
0000   ;
0000   ;       V03-003 WMC0002         Wayne Cardoza   22-Feb-1983
0000   ;               Resource removed from service bit.
0000   ;               Spec changes in MSTAT1, MSTAT2, MDECC.
0000   ;
0000   ;       V03-002 WMC0001         Wayne Cardoza   10-Feb-1983
0000   ;               Add error logging routine for adapters.
0000   ;
0000   ;--
```

```
              0000          .SBTTL  Declarations
              0000  ;
              0000  ; INCLUDE FILES:
              0000  ;
              0000
              0000  ;
              0000  ; INCLUDED SYSTEM SYMBOL DEFINITIONS
              0000  ;
              0000          $ADPDEF
              0000          $EMBDEF <MC,SE>
              0000          $IPLDEF
              0000          $PCBDEF
              0000          $PFNDEF
              0000          $PHDDEF
              0000          $PRDEF
              0000          $PR790DEF
              0000          $PSLDEF
              0000          $PTEDEF
              0000          $RPBDEF
              0000          $VADEF
              0000          $MCF790DEF
              0000          $MCHKDEF
              0000          $MERGDEF
              0000          $SBIADEF
              0000          $CSWPDEF
              0000          $PAMMDEF
              0000
              0000  ;
              0000  ; OWN STORAGE:
              0000  ;
              0000
          00000000          .PSECT  MCHK$DATA,QUAD,WRT
              0000
              0000  EXE$MCHK_ERRCNT::                       ; Used to locate error counters
              0000                                          ; via SYS.MAP.
              0000
              0000  IBOX_OLD1:                              ; Time of most recent IBOX error
00000000      0000          .LONG   0
              0004  IBOX_OLD2:                              ; Second most recent
00000000      0004          .LONG   0
              0008  IBOX_TOTAL:                             ; Total errors
00000000      0008          .LONG   0
0000000A      000C  IBOX_THRESHOLD = 10                     ; Minimum time allowed for 3 IBOX errors
              000C                                          ; (10 millisec units)
              000C
              000C  FBOX_OLD1:                              ; Time of most recent FBOX error
00000000      000C          .LONG   0
              0010  FBOX_OLD2:                              ; Second most recent
00000000      0010          .LONG   0
              0014  FBOX_TOTAL:                             ; Total errors
00000000      0014          .LONG   0
0000000A      0018  FBOX_THRESHOLD = 10                     ; Minimum time allowed for 3 FBOX errors
              0018                                          ; (10 millisec units)
              0018
              0018  EBOX_OLD1:                              ; Time of most recent EBOX error
00000000      0018          .LONG   0
              001C  EBOX_OLD2:                              ; Second most recent
```

```
00000000  001C              .LONG   0
          0020      EBOX_TOTAL:                         ; Total errors
00000000  0020              .LONG   0
0000000A  0024      EBOX_THRESHOLD = 10                 ; Minimum time allowed for 3 EBOX errors
          0024                                          ; (10 millisec units)
          0024
          0024      MBOX_FE_OLD:                        ; Time of most recent MBOX fatal error
00000000  0024              .LONG   0
          0028      MBOX_FE_PC:                         ; PC of most recent MBOX fatal error
00000000  0028              .LONG   0
          002C      MBOX_FE_TOTAL:                      ; Total errors
00000000  002C              .LONG   0
00000002  0030      MBOX_FE_THRESHOLD = 2               ; Minimum time allowed for 2 MBOX errors
          0030                                          ; (10 millisec units)
          0030      MBOX_FE_PHY_ADR:                    ; Physical address of last MBOX address PE
00000000  0030              .LONG   0                   ; error
          0034
          0034      TB_OLD1:                            ; Time of last TB parity error.
00000000  0034              .LONG   0
          0038      TB_OLD2:                            ; Time of next-to-last TB error.
00000000  0038              .LONG   0
          003C      TB_TOTAL:                           ; Total errors
00000000  003C              .LONG   0
0000000A  0040      TB_THRESHOLD = 10                   ; Allowable time between TB errors
          0040                                          ; (in 10 millisecond units).
          0040
          0040      CSH_A_OLD1:                         ; Time of most recent A cache error
00000000  0040              .LONG   0
          0044      CSH_A_OLD2:                         ; Second most recent
00000000  0044              .LONG   0
          0048      CSH_B_OLD1:                         ; Time of most recent B cache error
00000000  0048              .LONG   0
          004C      CSH_B_OLD2:                         ; Second most recent
00000000  004C              .LONG   0
          0050      CACHE_TOTAL:                        ; Total errors
00000000  0050              .LONG   0
0000000A  0054      CACHE_THRESHOLD = 10                ; Minimum time allowed for 3 cache errors
          0054                                          ; (10 millisec units)
          0054
          0054      MBOX_1D_OLD1:                       ; Time of most recent MBOX 1D error
00000000  0054              .LONG   0
          0058      MBOX_1D_OLD2:                       ; Second most recent
00000000  0058              .LONG   0
          005C      MBOX_1D_TOTAL:                      ; Total errors
00000000  005C              .LONG   0
0000000A  0060      MBOX_1D_THRESHOLD = 10              ; Minimum time allowed for 3 MBOX 1D errors
          0060                                          ; (10 millisec units)
          0060
          0060      CRD_OLD1:                           ; Time of most recent single bit error
00000000  0060              .LONG   0
          0064      CRD_OLD2:                           ; Second most recent
00000000  0064              .LONG   0
          0068      CRD_TOTAL:                          ; Total errors
00000000  0068              .LONG   0
00000064  006C      CRD_THRESHOLD = 100                 ; Minimum time allowed for 3 CRD errors
          006C                                          ; (10 millisec units)
          006C
```

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

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

```
          00D8   ;
          00D8   ; LOCAL VARIABLES FOR MACHINE CHECK HANDLER
          00D8   ;
          00D8
          00D8   ABUS_CYCLE:                                       ; Bit mask for ABUS related cycles
08088000  00D8           .LONG   1@MCF790$C_ABUS + 1@MCF790$C_ABUS_WRT + 1@MCF790$C_ABUS_REFL
          00DC
          00DC   ASSUME MCF790$V_ABORTS EQ 1                       ; Make sure there is room for software
          00DC                                                     ; abort bit
00000001  00DC   MCHK_ABORT = 1                                    ; Used by machine check handler to abort
          00DC
          00DC   ABORT_BITS:                                       ; Local copy of abort bits
00000000  00DC           .LONG   0
          00E0
          00E0   SBIA_ERR_SUM:                                     ; Temp copy of SBIA summary register
00000000  00E0           .LONG   0
          00E4
000000C8  00E4   CRD_LOG_SIZE = <2 * 4> + <16 * 3 * 4>             ; Buffer size
          00E4   CRD_BUFFER:                                       ; Single bit error logging buffer
000000C4  00E4           .LONG   4 + <16*3*4>                      ; Size of buffer
          00E8   CRD_FLAGS:                                        ; Flags - 1 -> logging turned off
00000000  00E8           .LONG   0
          00EC   CRD_FIRST_ENTRY:                                  ; Beginning of error entries
000001AC  00EC           .BLKL   16 * 3                            ; Room for 16 entries
          01AC   CRD_BUF_END:
          01AC
          01AC   CRD_NEXT:                                         ; Next CRD slot to be used
00000000  01AC           .LONG   0
          01B0
```

```
                                    01B0      ;
                                    01B0      ; warning messages
                                    01B0      ;
                                    01B0      FBOX_MSG:
                          00000034' 01B0              .LONG   20$-10$
5F 57 5F 4D 45 54 53 59 53 25 0A 0D 01B4      10$:    .ASCII  <13><10>-
4F 42 46 20 2C 46 46 4F 58 4F 42 46 01C0
66 66 6F 20 64 65 6E 72 75 74 20 58 01CC
6F 72 72 65 20 6F 74 20 65 75 64 20 01D8
                              73 72 01E4
                                    01E6              \%SYSTEM_W_FBOXOFF, FBOX turned off due to errors\-
                              0A 0D 01E6              <13><10>
                                    01E8      20$:
                                    01E8      ;
                                    01E8      ;
                                    01E8      CACHE_MSG:
                          0000003E' 01E8              .LONG   20$-10$
                                 0D 01EC      10$:    .ASCII  <13><10>-
43 5F 57 5F 4D 45 54 53 59 53 25 0A 01ED              \%SYSTEM_W_CACHEOFF, half of cache turned off due to errors\-
6C 61 68 20 2C 46 46 4F 45 48 43 41 01F9
74 20 65 68 63 61 63 20 66 6F 20 66 0205
75 64 20 66 66 6F 20 64 65 6E 72 75 0211
   73 72 6F 72 72 65 20 6F 74 20 65 021D
                              0A 0D 0228              <13><10>
                                    022A      20$:
```

```
00000000        .PSECT  MCHK$CODE,QUAD,RD,WRT
    0000        .SBTTL  Machine check entry point
    0000
    0000  ;++
    0000  ; MCHECK790 -- VENUS MACHINE CHECK
    0000  ;
    0000  ; FUNCTIONAL DESCRIPTION:
    0000  ;         All machine checks are vectored to this entry point.
    0000  ;
    0000  ; FORMAT OF VENUS'S MACHINE CHECK LOGOUT STACK:
    0000  ;         On entry to this machine check handler, the stack is set up as follows:
    0000  ;
    0000  ;                 00(SP): # bytes in machine check log on stack
    0000  ;                   .
    0000  ;                   .
    0000  ;                   machine check frame (see MCF790DEF)
    0000  ;                   .
    0000  ;                   .
    0000  ;
    0000  ;
    0000  ;         As soon as the machine check handler is invoked, it sets up the stack
    0000  ;         as follows:
    0000  ;         +--------------------------------+              :(SP)
    0000  ;         !               .                !
    0000  ;         !               .                !
    0000  ;         !       saved R0 - R6, AP        !
    0000  ;         !               .                !
    0000  ;         !               .                !
    0000  ;         +--------------------------------+
    0000  ;         ! pointer to PC/PSL of exception!
    0000  ;         +--------------------------------+
    0000  ;         !       recovery mask            !
    0000  ;         +--------------------------------+
    0000  ;         ! # bytes pushed on logout stack! :(AP)
    0000  ;         +--------------------------------+
    0000  ;         !                                !
    0000  ;         !                                !
    0000  ;         !            machine             !
    0000  ;         !             check              !
    0000  ;         !             frame              !
    0000  ;         !                                !
    0000  ;         !                                !
    0000  ;         !                                !
    0000  ;         +--------------------------------+
    0000  ;
    0000  ;         AP will point to the beginning of the machine check log on the stack.
    0000  ;         2 longwords are immediately pushed on top of the machine check log, and
    0000  ;         are referenced as negative offsets from AP.  These two longwords are
    0000  ;         input arguments to EXE$MCHK_BUGCHK, a routine that is called to check
    0000  ;         for a user-declared machine check recovery block.  This routine is
    0000  ;         called immediately before bugchecking, and expects the mask and the
    0000  ;         pointer to the exception PC/PSL to be right on top of the machine check
    0000  ;         log on the stack.
    0000  ;--
```

```
                                 0000        .ALIGN  LONG
                                 0000 EXE$MCHK::                                    ; Machine check handler.
                       03    DD  0000        PUSHL   #MCHK$M_MCK ! MCHK$M_LOG  ; Mask signals machine check.
                    60 AE    DF  0002        PUSHAL  MCF790$L_PC+4(SP)           ; Push pointer to exception PC/PSL.
                  107F 8F    BB  0005        PUSHR   #^M<R0,R1,R2,R3,R4,R5,R6,AP> ; Working registers.
             5C     5E 28    C1  0009        ADDL3   #<10*4>,SP,AP               ; AP points to mchk log frame.
                                 000D ;
                                 000D ; Do misc initialization
                                 000D ;
                                 000D ;
                             CB  000D        BICL3   #^C<MCF790$M_ABORTS ! - ; Get all the hardware abort bits
                                 000E                MCF790$M_AUTO_SHUT>,-
00DC'CF    0C AC  FFFDFFE1 8F    000E                MCF790$L_EBCS(AP),W^ABORT_BITS
                 04 AC       94  0018        CLRB    MCF790$B_MCHK_CODE(AP)    ; Just in case
                                 001B ;
                                 001B ; Look for one of the four basic error types
                                 001B ; BUGCHECK if we find two
                                 001B ;
                                 001B ; MBOX fatal errors
                                 001B ;
          04 0C AC     0F    E1  001B        BBC     #MCF790$V_MBOX_FE, MCF790$L_EBCS(AP), 10$
             04 AC     04    90  0020        MOVB    #MCF790$C_MBOX_FE, MCF790$B_MCHK_CODE(AP)
                                 0024 ;
                                 0024 ; FBOX errors
                                 0024 ;
          09 04 AC     1C    E1  0024 10$:   BBC     #MCF790$V_FBOX, MCF790$L_EHSR(AP), 20$
                04 AC        95  0029        TSTB    MCF790$B_MCHK_CODE(AP)   ; Make sure this is the first match
                   34        12  002C        BNEQ    BAD_MCHK
             04 AC     01    90  002E        MOVB    #MCF790$C_FBOX, MCF790$B_MCHK_CODE(AP)
                                 0032 ;
                                 0032 ; EBOX errors
                                 0032 ;
     0C AC    00001E00 8F    D3  0032 20$:   BITL    #MCF790$M_EDP_PE ! MCF790$M_USTK_PE !-
                                 003A                MCF790$M_ECS_PE ! MCF790$M_EMCR_PE, -
                                 003A                MCF790$L_EBCS(AP)
                       09    13  003A        BEQL    30$
                04 AC        95  003C        TSTB    MCF790$B_MCHK_CODE(AP)   ; Make sure this is the first match
                   21        12  003F        BNEQ    BAD_MCHK
             04 AC     02    90  0041        MOVB    #MCF790$C_EBOX, MCF790$B_MCHK_CODE(AP)
                                 0045 ;
                                 0045 ; IBOX errors
                                 0045 ;
          09 0C AC     0D    E1  0045 30$:   BBC     #MCF790$V_IBOX_ERR, MCF790$L_EBCS(AP), 40$
                04 AC        95  004A        TSTB    MCF790$B_MCHK_CODE(AP)   ; Make sure this is the first match
                   13        12  004D        BNEQ    BAD_MCHK
             04 AC     03    90  004F        MOVB    #MCF790$C_IBOX, MCF790$B_MCHK_CODE(AP)
                                 0053 ;
                                 0053 ; Call appropriate service routine
                                 0053 ;
                                 0053 40$:   CASE    MCF790$B_MCHK_CODE(AP),- ; Use machine check code
                                 0053                <CHECK_MBOX_ID,-         ; Nothing so far
                                 0053                F_ERR,-                  ; FBOX error
                                 0053                E_ERR,-                  ; EBOX error
                                 0053                I_ERR,-                  ; IBOX error
                                 0053                M_ERR>,-                 ; MBOX fatal error
                                 0053                TYPE=B
                                 0062 ; fall thru to bad code
```

```
                               0062   ;
                               0062   ; Always bugcheck - may be bug in this handler
                               0062   ;
                               0062   BAD_MCHK:
                 05D3   30     0062            BSBW    LOG_MCHECK                      ; log it
                               0065            BUG_CHECK BADMCKCOD,FATAL
                               0069   ;
                 00C0   30     0069   M_ERR:   BSBW    MBOX_FE_SERV                    ; MBOX_FE service routine
                   0D   11     006C            BRB     CHECK_MBOX_1D
                               006E   ;
                 0238   30     006E   F_ERR:   BSBW    FBOX_SERV                       ; FBOX service routine
                   08   11     0071            BRB     CHECK_MBOX_1D
                               0073   ;
                 0271   30     0073   E_ERR:   BSBW    EBOX_SERV                       ; EBOX service routine
                   03   11     0076            BRB     CHECK_MBOX_1D
                               0078   ;
                 008F   30     0078   I_ERR:   BSBW    IBOX_SERV                       ; IBOX service routine
                               007B   ;        BRB     CHECK_MBOX_1D
                               007B   ;
                               007B   ; fall thru to next stage of checking
                               007B   ;
                               007B   ; MBOX 1D interrupt indicated by MBOX_INTR without MBOX_FE
                               007B   ; TB problems are hard to spot - just check for the bits
                               007B   ;
                               007B   CHECK_MBOX_1D:
         17 0C AC   0F  E0     007B            BBS     #MCF790$V_MBOX_FE, MCF790$L_EBCS(AP), MCHK_EXIT
  38 AC  00000F00 8F    D3     0080            BITL    #MCF790$M_TB_TAG_PE ! MCF790$M_TB_A_PE ! -
                               0088             MCF790$M_TB_B_PE ! MCF790$M_TB_VAL_PE, -
                               0088            MCF790$L_MSTATT(AP)
                   05   12     0088            BNEQ    10$                             ; A TB problem
         08 0C AC   0E  E1     008A            BBC     #MCF790$V_MBOX_INT, MCF790$L_EBCS(AP), MCHK_EXIT
         00 04 AC   07  E2     008F   10$:     BBSS    #MCF790$V_MBOX_1D, MCF790$B_MCHK_CODE(AP), 20$
                 039A   30     0094   20$:     BSBW    MBOX_1D_SERV
                               0097   ;
                               0097   ; fall thru to common exit processing
                               0097   ;
                               0097   MCHK_EXIT:
                04 AC   95     0097            TSTB    MCF790$B_MCHK_CODE(AP)
                   C6   13     009A            BEQL    BAD_MCHK                        ; We couldn't figure it out
                 0599   30     009C            BSBW    LOG_MCHECK                      ; Log it
                               009F   ;
                               009F   ; If any abort bits are set, we cannot resume the instruction
                               009F   ;
              00DC'CF   D5     009F            TSTL    W^ABORT_BITS
                   12   12     00A3            BNEQ    REFLECT_MCHK
              107F 8F   BA     00A5            POPR    #^M<R0,R1,R2,R3,R4,R5,R6,AP>
                5E 08   C0     00A9            ADDL    #<2*4>,SP                       ; Get rid of recovery block check data
                5E 8E   C0     00AC            ADDL    (SP)+,SP                        ; Get rid of the machine chack frame
      0000004A 8F  00   DA     00AF            MTPR    #0,#PR790$_EHSR                 ; Clear the VMS entered flag
                   02          00B6            REI                                     ; At last - continue
```

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

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

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

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

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

```
              0028'CF  D4 020B          CLRL    W^MBOX_FE_PC                    ; matching PC doesn't justify bugcheck
                   15  11 020F          BRB     70$                             ; go finish up
                E2 54  F4 0211  52$:    SOBGEQ  R4,51$                          ; loop through all 4
             50 04 A0  D0 0214  53$:    MOVL    ADP$L_LINK(R0),R0               ; follow ADP list to end
                   CF  11 0218          BRB     50$
                   63  11 021A  55$:    BRB     120$                                    ; full log & bugcheck (abort first)
                          021C  ;
  08 A1  00600000 8F   D3 021C  60$:    BITL    #SBIA$M_ADP ! SBIA$M_RCP, -             ; ADP or RCP also didn't reach SBI
                          0224                  SBIA$L_SUMRY(R1)
                   59  13 0224          BEQL    120$
          04 AC  40 8F 88 0226  70$:    BISB    #MCF790$M_SBIA_ERR, -                   ; add SBIA error summary
                          022B                  MCF790$B_MCHK_CODE(AP)
        00E0'CF 08 A1  D0 022B          MOVL    SBIA$L_SUMRY(R1),W^SBIA_ERR_SUM ; save the error summary
          08 A1 08 A1  C8 0231          BISL    SBIA$L_SUMRY(R1),SBIA$L_SUMRY(R1) ; clear all the error locks
  34 A1  00001000 8F   C8 0236          BISL    #SBIA$M_CTO,SBIA$L_SBIERR(R1)   ; and the CPU timeout bit
             52 5C BC  9A 023E          MOVZBL  @MCF790$L_PC(AP),R2                     ; get the guilty instruction
     01 00B8'CF    52  E0 0242          BBS     R2,W^SBI_INST,90$                       ; does instruction only do one read
                   05     0248          RSB                                             ; no - eventually abort
             0E    53  91 0249  90$:    CMPB    R3,#MCF790$C_CP_READ                    ; read cycle?
                   07  12 024C          BNEQ    100$
        00DC'CF    02  CA 024E          BICL    #MCF790$M_IO_RD,W^ABORT_BITS            ; clear IO read abort
                   05  11 0253          BRB     110$
        00DC'CF    04  CA 0255  100$:   BICL    #MCF790$M_MEM_WRT, W^ABORT_BITS ; clear write abort
                          025A  ;
                          025A  ; now make sure errors are not repeated
                          025A  ;
        0028'CF 5C AC  D1 025A  110$:   CMPL    MCF790$L_PC(AP),W^MBOX_FE_PC    ; same PC for two errors
                   1A  13 0260          BEQL    115$                            ; yes - bugcheck
             50    1B  DB 0262          MFPR    #PR790$_TODR,R0                 ; Current time - 10 millisec units
     52   50  0024'CF  C3 0265          SUBL3   W^MBOX_FE_OLD,R0,R2             ; Time since last error
             02    52  D1 026B          CMPL    R2, #MBOX_FE_THRESHOLD
                   0C  1F 026E          BLSSU   115$                            ; Too quick - bugcheck
           0024'CF 50  D0 0270          MOVL    R0,W^MBOX_FE_OLD                ; Save time of this error
        0028'CF 5C AC  D0 0275          MOVL    MCF790$L_PC(AP), W^MBOX_FE_PC   ; and the PC
                   05     027B          RSB
                          027C  ;
                 FE70  31 027C  115$:   BRW     BUGCHECK_POP                            ; Clean off return address
                          027F  ;
             FC AC 04  C8 027F  120$:   BISL    #MCHK$M_NEXM,-4(AP)                     ; Add NXM to recovery mask
             04 AC 20  88 0283          BISB    #MCF790$M_SBIA, -                       ; log full SBIA
                          0287                  MCF790$B_MCHK_CODE(AP)                  ; eventually abort (bit set)
                   05     0287          RSB                                             ; and bugcheck (assume kernel mode)
                          0288  ;
                          0288  ; Give up if there is no cycle type
                          0288  ;
             00    53  91 0288  130$:   CMPB    R3, #MCF790$C_NOP
                   EF  13 028B          BEQL    115$
                          028D  ;
                          028D  ; if a cycle parameter RAM error, give up
                          028D  ;
        00C00000 8F    D3 028D          BITL    #MCF790$M_CPR_PE_A ! MCF790$M_CPR_PE_B,-
                38 AC     0293                  MCF790$L_MSTAT1(AP)
                   E5  12 0295          BNEQ    115$
                          0297  ;
                          0297  ; Look for an error writing an MBOX register
                          0297  ; MBOX confused, so no recovery
                          0297  ; 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          .SBTTL  FBOX errors
                         02A9     ;++
                         02A9     ; An FBOX error has been detected
                         02A9     ; These are all treated alike
                         02A9     ; This is cause for turning off the FBOX if they are too frequent
                         02A9     ; A console message is printed if FBOX is turned off
                         02A9     ;--
                         02A9     FBOX_SERV:
         0014'CF   D6    02A9          INCL    W^FBOX_TOTAL                 ; Count the error
         50   1B   DB    02AD          MFPR    #PR790$_TODR,R0              ; Current time - 10 millisec units
52   50  0010'CF   C3    02B0          SUBL3   W^FBOX_OLD2,R0,R2            ; Time for three errors
         0A   52   D1    02B6          CMPL    R2,#FBOX_THRESHOLD           ; Too quick
              1F   1A    02B9          BGTRU   10$                          ; No
         28   00   DA    02BB          MTPR    #0,#PR790$_ACCS              ; Turn FBOX off
              10   88    02BE          BISB    #MCF790$M_RSRC_REM,-         ; Indicate resource removed
           04 AC         02C0                  MCF790$B_MCHK_CODE(AP)
      51 01B0'CF   D0    02C2          MOVL    W^FBOX_MSG,R1                ; Mesage length
      52 01B4'CF   9E    02C7          MOVAB   W^FBOX_MSG+4,R2              ; Message address
 55  00000000'GF   9E    02CC          MOVAB   G^OPA$UCB0,R5                ; Send it to the console terminal
     00000000'GF   16    02D3          JSB     G^IOC$BROADCAST
              05         02D9          RSB
0010'CF  000C'CF   D0    02DA     10$: MOVL    W^FBOX_OLD1, W^FBOX_OLD2 ; Save new times
   000C'CF    50   D0    02E1          MOVL    R0,W^FBOX_OLD1
              05         02E6          RSB
```

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

```
                             0313          .SBTTL  Bad Data or Double Bit error
                             0313  ;++
                             0313  ; If possible a new copy of the page is paged in.
                             0313  ; A double bit error will cause the physical page to be put on the bad page list
                             0313  ; while a bad data error simply gets a new copy since it is not the array that
                             0313  ; is at fault.
                             0313  ;--
                             0313  BAD_MEM:
        03 60 AC    1A   E1  0313          BBC     #PSL$V_IS, MCF790$L_PSL(AP),1$
                  FDD4   31  0318          BRW     BUGCHECK_POP              ; Interrupt stack - must give up
02   60 AC    05    10   EC  031B  1$:     CMPV    #PSL$V_IPL, #PSL$S_IPL, -
                             0321          MCF790$L_PSL(AP), #IPL$_ASTDEL ; Are we at a non-pageable priority?
                    4E   14  0321          BGTR    12$                       ; Abort - recovery is useless
  50    50 AC    F7 8F   78  0323          ASHL    #-9, MCF790$L_MEAR(AP),R0 ; Get physical PFN of error
     00000000'GF    50   D1  0329          CMPL    R0,G^MMG$GL_MAXPFN        ; Is there PFN data base for page?
                    3F   1A  0330          BGTRU   12$                       ; Br if no PFN data base for page
      7E   00000000'GF   D0  0332          MOVL    G^PFN$AL_PTE,-(SP)
            53    9E40   D0  0339          MOVL    @(SP)+[R0],R3             ; address of PTE
                    04   13  033D          BEQL    3$                        ; none - give up
                    63   D5  033F          TSTL    (R3)
                    03   19  0341          BLSS    5$                        ; Branch if page valid
                  00D0   31  0343  3$:     BRW     RDSNONRES                 ; else fatal error
            27 63   15   E0  0346  5$:     BBS     #PTE$V_WINDOW,(R3),12$    ; BR if page is PFN-mapped
                    14   E1  034A          BBC     #MCF790$V_DBL_BIT,-       ; Is it a double bit error
              0B 40 AC       034C          MCF790$L_MDECC(AP),7$     ; If not, let page go to free list
      7E   00000000'GF   D0  034F          MOVL    G^PFN$AB_TYPE,-(SP)       ; PFN type array address
            9E40    20   88  0356          BISB    #PFN$M_BADPAG,@(SP)+[R0]  ; Mark page bad
                    51   D4  035A  7$:     CLRL    R1                        ; Clear modify bit propagator
            04 63   1A   E5  035C          BBCC    #PTE$V_MODIFY,(R3),10$    ; Test (& clear) modify bit in PTE
          51    80 8F    9A  0360          MOVZBL  #PFN$M_MODIFY,R1          ; Set modify propagator
      7E   00000000'GF   D0  0364  10$:    MOVL    G^PFN$AB_STATE,-(SP)      ; Address of PFN state array
            9E40    51   88  036B          BISB    R1,@(SP)+[R0]             ; Propagate modify bit to PFN database
                             036F
                             036F          ASSUME  PFN$M_MODIFY EQ 128
                             036F
                    06   14  036F          BGTR    15$                       ; Page not modified - he's OK
         00DC'CF    01   88  0371  12$:    BISB    #MCHK_ABORT,W^ABORT_BITS  ; Force an abort
                    05       0376          RSB
      7E   00000000'GF   D0  0377  15$:    MOVL    G^PFN$AW_REFCNT,-(SP)     ; Address of PFN refcnt array
            01    9E40   B1  037E          CMPW    @(SP)+[R0],#1             ; Check for I/O in progress, etc.
                    ED   1A  0382          BGTRU   12$                       ; If so, don't try anything fancy
      7E   00000000'GF   D0  0384          MOVL    G^PFN$AB_TYPE,-(SP)       ; Address of PFN type array
                             038B
                             038B  ;       In the future we may recover from hard ecc errors on global pages
                             038B  ;       as well, but for now we abort the image.
                             038B
                             038B          ASSUME  PFN$C_SYSTEM EQ 1         ; Check type of page
                             038B          ASSUME  PFN$C_PROCESS EQ 0
                             038B
        9E40    03    00     EC  038B          CMPV    #PFN$V_PAGTYP, #PFN$S_PAGTYP, -
                      01         0390          @(SP)+[R0],#PFN$C_SYSTEM ; Check for system or global page
                    DE   1A  0391          BGTRU   12$                       ; Branch if page table or global page
                    0D   12  0393          BNEQ    30$                       ; Branch if page is process private
      54   00000000'GF   9E  0395          MOVAB   G^MMG$AL_SYSPCB,R4        ; System pages are kept track of in
          55    6C A4    D0  039C          MOVL    PCB$L_PHD(R4),R5          ; a working set list in the system pcb
                    2A   11  03A0          BRB     40$
                             03A2                                            ; 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          .SBTTL MBOX 1D errors
                              0431   ;++
                              0431   ; An MBOX error was reported when IPL dropped below 1D
                              0431   ; these are not directly related to instruction execution
                              0431   ; From here we will dispatch to one of two subcategories
                              0431   ;          asynchronous CPU errors
                              0431   ;          DMA errors
                              0431   ; TB errors do not latch the cycle type, so a separate check is required
                              0431   ; for them
                              0431   ;
                              0431   ; Cycle parameter RAM errors are occasionally reported here instead of as
                              0431   ; MBOX FE
                              0431   ;--
                              0431   MBOX_1D_SERV:
                              0431   ;
                              0431   ; First make sure we have a single error, otherwise give up
                              0431   ;
        0A 3C AC    07    E0  0431          BBS     #MCF790$V_MUL_ERR, MCF790$L_MSTAT2(AP),5$
                              0436   ;
                              0436   ; Now look for that errant CPR parity error
                              0436   ;
         00C00000 8F    D3    0436          BITL    #MCF790$M_CPR_PE_A ! MCF790$M_CPR_PE_B,-
                  38 AC       043C                  MCF790$L_MSTAT1(AP)
                     03  13   043E          BEQL    10$
                   FCAC  31   0440   5$:    BRW     BUGCHECK_POP             ; Disaster - give up
                              0443   ;
                              0443   ; Finally, do the normal error processing
                              0443   ;
   38 AC    00000F00 8F  D3   0443   10$:   BITL    #MCF790$M_TB_TAG_PE ! -
                              044B                   MCF790$M_TB_A_PE ! -
                              044B                   MCF790$M_TB_B_PE ! -
                              044B                   MCF790$M_TB_VAL_PE, -
                              044B                  MCF790$L_MSTAT1(AP)
                     11  12   044B          BNEQ    MBOX_1D_CPU              ; A TB problem
  53   38 AC    04   1A  EF   044D          EXTZV   #MCF790$V_CYCLE_TYP, -
                              0453                  #MCF790$S_CYCLE_TYP, -
                              0453                  MCF790$L_MSTAT1(AP),R3   ; get cycle type
        02 00D8'CF   53  E0   0453          BBS     R3,W^ABUS_CYCLE,20$      ; is it ABUS related
                     03  11   0459          BRB     MBOX_1D_CPU              ; a CPU error
                   0117  31   045B   20$:   BRW     MBOX_1D_DMA              ; a DMA error
```

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

```
           18 38 AC       04AA              MCF790$L_MSTAT1(AP), 50$
  52    50   0044'CF  C3  04AD        SUBL3  W^CSH_A_OLD2,R0,R2        ; Time for three errors
        0A      52    D1  04B3        CMPL   R2,#CACHE_THRESHOLD       ; Too quick
                16    1F  04B6        BLSSU  60$
 0044'CF   0040'CF    D0  04B8        MOVL   W^CSH_A_OLD1, W^CSH_A_OLD2 ; Save new times
     0040'CF    50    D0  04BF        MOVL   R0, W^CSH_A_OLD1
                      05  04C4        RSB
                          04C5
  52    50   004C'CF  C3  04C5  50$:  SUBL3  W^CSH_B_OLD2, R0,R2       ; Time for three errors
        0A      52    D1  04CB        CMPL   R2,#CACHE_THRESHOLD       ; Too quick
                6E    1F  04CE  60$:  BLSSU  CACHE_OFF
 004C'CF   0048'CF    D0  04D0        MOVL   W^CSH_B_OLD1, W^CSH_B_OLD2 ; Save new times
     0048'CF    50    D0  04D7        MOVL   R0, W^CSH_B_OLD1
                      05  04DC        RSB
                          04DD  ;
                          04DD  ; data address parity error
                          04DD  ;
     28 40 AC   13    E1  04DD  70$:  BBC    #MCF790$V_ADR_PE, -                       ; Something else
                          04E2               MCF790$L_MDECC(AP),80$
            50 AC     D1  04E2        CMPL   MCF790$L_MEAR(AP),-                       ; same physical address
           0030'CF        04E5               W^MBOX_FE_PHY_ADR
                32    13  04E8        BEQL   95$                                       ; yes - bugcheck
           50   1B    DB  04EA        MFPR   #PR790$_TODR,R0                           ; Current time - 10 millisec units
  52    50   0024'CF  C3  04ED        SUBL3  W^MBOX_FE_OLD,R0,R2                       ; Time since last error
        02      52    D1  04F3        CMPL   R2, #MBOX_FE_THRESHOLD
                24    1F  04F6        BLSSU  95$                                       ; Too quick - bugcheck
     0024'CF    50    D0  04F8        MOVL   R0,W^MBOX_FE_OLD                          ; Save time of this error
   0028'CF   5C AC    D0  04FD        MOVL   MCF790$L_PC(AP), W^MBOX_FE_PC             ; Save PC
   0030'CF   50 AC    D0  0503        MOVL   MCF790$L_MEAR(AP), W^MBOX_FE_PHY_ADR ; Save physical address
                      05  0509        RSB
                          050A  ;
                          050A  ; Double bit or bad data errors from memory.
                          050A  ; These errors are either in the context of the correct process or the BAD_MEM
                          050A  ; routine will spot it
                          050A  ;
 40 AC    00500000 8F     D3  050A  80$:  BITL   #MCF790$M_BAD_DATA ! -  ; check error bits
                          0512                MCF790$M_DBL_BIT, -
                          0512               MCF790$L_MDECC(AP)
                03    13  0512        BEQL   90$                                       ; Nope - something else
              FDFC    31  0514        BRW    BAD_MEM                                   ; Go try to recover
                          0517  ;
                          0517  ; ABUS bad data flag
                          0517  ;
                0E    E0  0517  90$:  BBS    #MCF790$V_AB_BAD_DAT,-  ; Go count misc errors
             3C AC        0519               MCF790$L_MSTAT2(AP),-
                00        051B               MISC_MBOX_1D
                          051C  95$:                                                   ; Fall through to misc
                          051C  ;
                          051C  ; misc MBOX errors
                          051C  ;
                          051C  MISC_MBOX_1D:
           005C'CF    D6  051C        INCL   W^MBOX_1D_TOTAL                ; Count the error
           50   1B    DB  0520        MFPR   #PR790$_TODR,R0                ; Current time - 10 millisec units
  52    50   0058'CF  C3  0523        SUBL3  W^MBOX_1D_OLD2,R0,R2           ; Time for three errors
        0A      52    D1  0529        CMPL   R2, #MBOX_1D_THRESHOLD         ; Too quick
                03    1A  052C        BGTRU  100$                           ; No
              FBBE    31  052E        BRW    BUGCHECK_POP                   ; Clean off return address before BUGCHECK
```

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

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

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

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

```
                         0638         .SBTTL  Logging routines for machine checks
                         0638   ;++
                         0638   ; LOG_MCHECK -- format inputs to LOGGER
                         0638   ;
                         0638   ;
                         0638   ; IMPLICIT INPUTS:
                         0638   ;       (AP): points to machine check log on stack
                         0638   ;
                         0638   ; OUTPUTS:
                         0638   ;       Error is formatted and logged in system error log.
                         0638   ;       If no error log buffer, return with error status in R0.
                         0638   ;       R0-R6 destroyed.
                         0638   ;--
                         0638
                         0638   LOG_MCHECK:
                         0638   ;
                         0638   ; Test if a machine check recovery block that specifies no error
                         0638   ; logging is in effect.
                         0638   ;
      51   F8 AC    7D   0638         MOVQ    -8(AP),R1              ; Get PC-PSL pointer and mask in R1,R2
       00000000'GF  16   063C         JSB     G^EXE$MCHK_TEST        ; Logging inhibited?
             36 50  E8   0642         BLBS    R0,20$                 ; Branch if YES - but go clear SBIA
                         0645                                        ; Set up inputs to LOGGER.
       00000000'GF  D6   0645         INCL    G^EXE$GL_MCHKERRS      ; Bump machine check error count
51     00000074 8F  D0   064B         MOVL    #MCF790$L_PSL -        ; Size of machine check frame
                         0652                 + EMB$B_MC_SUMCOD -    ; Add space for log header.
                         0652                 + 4 ,R1                ; SBIA summary
       00000000'GF  16   0652         JSB     G^ERL$ALLOCEMB         ; Get error logging buffer.
             20 50  E9   0658         BLBC    R0, 20$                ; Br if failed to get buffer.
                52  DD   065B         PUSHL   R2                     ; Save buffer addr on stack.
      04 A2     02  B0   065D         MOVW    #EMB$K_MC, EMB$W_MC_ENTRY(R2) ; Set entry type.
         0060 8F    28   0661         MOVC3   #MCF790$L_PSL,-        ; transfer info to log
             04 AC       0665                 MCF790$L_EHSR(AP),-    ; do not include the size word
             10 A2       0667                 EMB$B_MC_SUMCOD(R2)
                06  E1   0669         BBC     #MCF790$V_SBIA_ERR,-   ; Should the SBIA error summary be added
          05 04 AC       066B                 MCF790$B_MCHK_CODE(AP),15$
   63     00E0'CF   D0   066E         MOVL    W^SBIA_ERR_SUM,(R3)    ; Add summary at end of MOVC3 transfer
                04  BA   0673   15$:  POPR    #^M<R2>                ; Retreive buffer address.
       00000000'GF  16   0675         JSB     G^ERL$RELEASEMB        ; Give buffer to logger.
                         067B   ;
                         067B   ; Now see if we need to log the SBIA data
                         067B   ;
                05  E1   067B   20$:  BBC     #MCF790$V_SBIA,-       ; Is the logging flag set
          12 04 AC       067D                 MCF790$B_MCHK_CODE(AP),40$
                10  EF   0680         EXTZV   #MCF790$V_AB_ADPT,-    ; relative IOA number
50    38 AC     02       0682                 #MCF790$S_AB_ADPT, -
                         0686                 MCF790$L_MSTAT1(AP), R0
53    0000'CF40     D0   0686         MOVL    W^ABUS_VA[R0],R3       ; base address of SBIA
      51   F8 AC    7D   068C         MOVQ    -8(AP),R1              ; Get PC-PSL pointer and mask in R1,R2
                23  11   0690         BRB     LOGSBI                 ; Go to the SBIA logger
                         0692
                    05   0692   40$:  RSB
```

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

```
                                  06B5          .SBTTL  SBIA logging
                                  06B5  ;++
                                  06B5  ; This routine is used to log SBIA (IOA) and SBI registers and related
                                  06B5  ; information.
                                  06B5  ;
                                  06B5  ; INPUTS:
                                  06B5  ;         R1 -> Address of PC, PSL
                                  06B5  ;         R3 -> Base adress of IOA registers
                                  06B5  ;--
             52      01     D0    06B5  LOGSBI: MOVL    #MCHK$M_LOG,R2              ; Mask to test if logging is enabled
              00000000'GF    16    06B8          JSB     G^EXE$MCHK_TEST             ; Enabled?
                  03 50     E9    06BE          BLBC    R0,10$                      ; yes
                    00C9    31    06C1  5$:     BRW     60$                         ; go clear errors
             55      51     D0    06C4  10$:    MOVL    R1,R5                       ; Save address of PC, PSL
                                  06C7  ;                  adapters   silo      registers      overhead
                000000E4          06C7          LOG_SIZE =  <16*4> + <16*4>  +  <21*4>  +  EMB$B_MC_SUMCOD
     51     000000E4 8F     D0    06C7          MOVL    #LOG_SIZE,R1                ; Need this much for error log buffer
              00000000'GF    16    06CE          JSB     G^ERL$ALLOCEMB              ; Request it
                  EA 50     E9    06D4          BLBC    R0,5$                       ; Can't get it - give up quietly
           04 A2     0D     B0    06D7          MOVW    #EMB$K_SBIA, EMB$W_MC_ENTRY(R2) ; Entry type code
 54  52     000000E4 8F     C1    06DB          ADDL3   #LOG_SIZE,R2,R4             ; Address of end of error log buffer
             74      65     7D    06E3          MOVQ    (R5),-(R4)                  ; PC, PSL
     56     00000000'GF     D0    06E6          MOVL    G^EXE$GL_CONFREGL,R6        ; Array of NEXUS device type codes
     55     00000000'GF     D0    06ED          MOVL    G^MMG$GL_SBICONF,R5         ; Array of adapter VA's
                     50     D4    06F4          CLRL    R0                          ; Start with the first SBIA
  00000000'EF40      53     D1    06F6  15$:    CMPL    R3,ABUS_VA[R0]              ; Is it this one?
                     08     13    06FE          BEQL    20$
               F2 50 04     F2    0700          AOBLSS  #4,R0,15$
                                  0704          BUG_CHECK SBIAERROR,FATAL           ; No match
 50     00000000'EF40       9A    0708  20$:    MOVZBL  ABUS_INDEX[R0],R0           ; Index in arrays for this SBIA
         50     50   02     78    0710          ASHL    #2,R0,R0                    ; Get longword offset
             56      50     C0    0714          ADDL    R0,R6                       ; EXE$GL_CONFREGL
             55      50     C0    0717          ADDL    R0,R5                       ; MMG$GL_SBICONF
             50      0F     D0    071A          MOVL    #15,R0                      ; Index of last possible item on SBI
                     74     D4    071D  30$:    CLRL    -(R4)                       ; Assume no adaptor here
          51       6540     D0    071F          MOVL    (R5)[R0],R1                 ; Get VA of controller/adapter
                     08     18    0723          BGEQ    40$                         ; GEQ implies no valid system VA.
                   6640     D5    0725          TSTL    (R6)[R0]                    ; Test adapter type
                     03     13    0728          BEQL    40$                         ; If eql, no adaptor here
             64      61     D0    072A          MOVL    (R1),(R4)                   ; Store adaptor CSR0 on stack
                  ED 50     F4    072D  40$:    SOBGEQ  R0,30$                      ; Loop thru all possible 16
             50      0F     D0    0730          MOVL    #15,R0                      ; Set up count of times to read silo
       74         30 A3     D0    0733  50$:    MOVL    SBIA$L_SBISILO(R3),-(R4)    ; Save information for error logger
                  F9 50     F4    0737          SOBGEQ  R0,50$                      ; Loop thru all 16
       74         38 A3     D0    073A          MOVL    SBIA$L_TMOADDRS(R3),-(R4)   ; SBI timeout address
       74         34 A3     D0    073E          MOVL    SBIA$L_SBIERR(R3),-(R4)     ; SBI error register
       74         44 A3     D0    0742          MOVL    SBIA$L_MAINT(R3),-(R4)      ; SBI maintenance register
       74         40 A3     D0    0746          MOVL    SBIA$L_SILOCMP(R3),-(R4)    ; SBI silo comparator
       74         3C A3     D0    074A          MOVL    SBIA$L_SBISTS(R3),-(R4)     ; SBI fault/status register
             74      63     D0    074E          MOVL    SBIA$L_CR(R3),-(R4)         ; IOA_CF (configuration register)
       74         04 A3     D0    0751          MOVL    SBIA$L_CSR(R3),-(R4)        ; IOA_CS (control/status register)
       74         08 A3     D0    0755          MOVL    SBIA$L_SUMRY(R3),-(R4)      ; IOA_ES (error summary register)
       74         0C A3     D0    0759          MOVL    SBIA$L_DIAGNOS(R3),-(R4)    ; IOA_DC (diagnostic control register)
       74         10 A3     D0    075D          MOVL    SBIA$L_DMAICA(R3),-(R4)     ; DMAI cmd/address register
       74         14 A3     D0    0761          MOVL    SBIA$L_DMAIID(R3),-(R4)     ; DMAI ID register
       74         18 A3     D0    0765          MOVL    SBIA$L_DMAACA(R3),-(R4)     ; DMAA cmd/address register
       74         1C A3     D0    0769          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$B_MC_SUMCOD,-(R4) ; Frame size
          00000000'GF   16  0787  1373          JSB    G^ERL$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         .SBTTL  Single Bit (CRD) error handling
                              07A3 ;++
                              07A3 ; This routine logs single bit errors.
                              07A3 ; 16 errors are accumulated before an error log entry is made.
                              07A3 ; 3 errors in 1 second causes reporting to be turned off for 5 minutes
                              07A3 ;
                              07A3 ; Due to a hardware bug, we must never run with CRD interrupts turned off.
                              07A3 ; Doing so can cause us to miss double-bit errors.  Instead, we will only
                              07A3 ; disable logging of the errors.
                              07A3 ;--
                              07A3         .ALIGN  LONG
                              07A4 EXE$INT54::
        70 00E8'CF      E8    07A4         BLBS    W^CRD_FLAGS,20$          ; Is logging turned off
                07      BB    07A9         PUSHR   #^M<R0,R1,R2>
           0068'CF      D6    07AB         INCL    W^CRD_TOTAL              ; Count the error
     50    01AC'CF      D0    07AF         MOVL    W^CRD_NEXT,R0            ; Next slot in CRD log buffer
                0A      12    07B4         BNEQ    10$                      ; Is it initialized
     50    00EC'CF      9E    07B6         MOVAB   W^CRD_FIRST_ENTRY,R0     ; First slot in CRD log buffer
   01AC'CF      50      D0    07BB         MOVL    R0,W^CRD_NEXT            ; Initialize it
 0000004E 8F    27      DA    07C0 10$:    MTPR    #^X27,#PR790$_LSPA       ; MDECC code to scratchpad address
 80   0000004F 8F       DB    07C7         MFPR    #PR790$_RSPD,(R0)+       ; Save MDECC
 0000004E 8F    2A      DA    07CE         MTPR    #^X2A,#PR790$_LSPA       ; MEAR code to scratchpad address
 80   0000004F 8F       DB    07D5         MFPR    #PR790$_RSPD,(R0)+       ; Save MEAR
 0000004E 8F    26      DA    07DC         MTPR    #^X26,#PR790$_LSPA       ; MSTAT2 code to scratchpad address
 80   0000004F 8F       DB    07E3         MFPR    #PR790$_RSPD,(R0)+       ; Save MSTAT2
             51 1B      DB    07EA         MFPR    #PR790$_TODR,R1          ; Current time - 10 millisecond units
 52   51   0064'CF      C3    07ED         SUBL3   W^CRD_OLD2,R1,R2         ; Time for three errors
 00000064 8F    52      D1    07F3         CMPL    R2,#CRD_THRESHOLD        ; Too quick
                6B      1F    07FA         BLSSU   CRD_OFF                  ; Yes
 0064'CF   0060'CF      D0    07FC         MOVL    W^CRD_OLD1,W^CRD_OLD2    ; Save new times
   0060'CF      51      D0    0803         MOVL    R1,W^CRD_OLD1
     51    01AC'CF      9E    0808         MOVAB   W^CRD_BUF_END,R1         ; End of buffer
           51   50      D1    080D         CMPL    R0,R1                    ; Is buffer full?
                08      1E    0810         BGEQU   CRD_LOG                  ; Yes
   01AC'CF      50      D0    0812         MOVL    R0,W^CRD_NEXT            ; Next buffer slot
                07      BA    0817         POPR    #^M<R0,R1,R2>
                02            0819 20$:    REI
                              081A ;
                              081A ; Log the current buffer and clear it for reuse.
                              081A ;
                              081A CRD_LOG:
                38      BB    081A         PUSHR   #^M<R3,R4,R5>
           52   01      D0    081C         MOVL    #MCHK$M_LOG,R2           ; Mask to test if logging is enabled
       00000000'GF      16    081F         JSB     G^EXE$MCHK_TEST          ; Enabled?
           28   50      E8    0825         BLBS    R0,10$                   ; No - just go clear buffer
 51    000000D8 8F      D0    0828         MOVL    #CRD_LOG_SIZE + EMB$B_MC_SUMCOD,R1 ; Error log buffer size
       00000000'GF      16    082F         JSB     G^ERL$ALLOCEMB           ; Request it
           18   50      E9    0835         BLBC    R0,10$                   ; Can't get it - just clear the buffer
 00000004'EF    0E      B0    0838         MOVW    #EMB$K_CRD, EMB$W_MC_ENTRY ; Entry type code
                52      DD    083F         PUSHL   R2
   00E4'CF      51      28    0841         MOVC3   R1,W^CRD_BUFFER,-        ; Move log to buffer
             10 A2            0846                 EMB$B_MC_SUMCOD(R2)
                04      BA    0848         POPR    #^M<R2>
       00000000'GF      16    084A         JSB     G^ERL$RELEASEMB
 50    00EC'CF          9E    0850 10$:    MOVAB   W^CRD_FIRST_ENTRY,R0     ; First slot in CRD log buffer
 01AC'CF        50      D0    0855         MOVL    R0,W^CRD_NEXT            ; Start here next time
 00   60        00      2C    085A         MOVC5   #0,(R0),#0,-             ; Clear the buffer
```

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

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

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

```
ABORT_BITS                 000000DC R      02     EMB$K_SBIA                 = 0000000D
ABUS_CYCLE                 000000D8 R      02     EMB$W_HD_ENTRY             = 00000004
ABUS_INDEX                 ********    X   03     EMB$W_MC_ENTRY             = 00000004
ABUS_TYPE                  ********    X   03     ERL$ALLOCEMB                 ********    X   03
ABUS_VA                    ********    X   03     ERL$RELEASEMB                ********    X   03
ADP$L_LINK               = 00000004               EXE$AB_MEMERR                00000070 RG     02
ADP$L_UBASCB             = 00000044               EXE$GL_CONFREGL              ********    X   03
ADP$W_ADPTYPE            = 0000000E               EXE$GL_FLAGS                 ********    X   03
AT$_UBA                    ********    X   03     EXE$GL_MCHKERRS              ********    X   03
BAD_MCHK                   00000062 R      03     EXE$INT54                    000007A4 RG     03
BAD_MEM                    00000313 R      03     EXE$INT58                    00000608 RG     03
BUG$_BADMCKCOD             ********    X   03     EXE$INT5C                    00000608 RG     03
BUG$_MACHINECHK            ********    X   03     EXE$INT60                    00000620 RG     03
BUG$_RDSNONRES             ********    X   03     EXE$MCHECK                   ********    X   03
BUG$_SBIAERROR             ********    X   03     EXE$MCHK                     00000000 RG     03
BUGCHECK                   000000F2 R      03     EXE$MCHK_BUGCHK              ********    X   03
BUGCHECK_NOLOG             000000F5 R      03     EXE$MCHK_ERRCNT              00000000 RG     02
BUGCHECK_POP               000000EF R      03     EXE$MCHK_TEST                ********    X   03
CACHE_ERR                  000004A1 R      03     EXE$V_CRDENABL               ********    X   03
CACHE_MSG                  000001E8 R      02     E_ERR                        00000073 R      03
CACHE_OFF                  0000053E R      03     FBOX_MSG                     000001B0 R      02
CACHE_THRESHOLD          = 0000000A               FBOX_OLD1                    0000000C R      02
CACHE_TOTAL                00000050 R      02     FBOX_OLD2                    00000010 R      02
CHECK_MBOX_1D              0000007B R      03     FBOX_SERV                    000002A9 R      03
CON$KEEPALIVE              0000089D R      03     FBOX_THRESHOLD             = 0000000A
CRD_BUFFER                 000000E4 R      02     FBOX_TOTAL                   00000014 R      02
CRD_BUF_END                000001AC R      02     F_ERR                        0000006E R      03
CRD_FIRST_ENTRY            000000EC R      02     IBOX_OLD1                    00000000 R      02
CRD_FLAGS                  000000E8 R      02     IBOX_OLD2                    00000004 R      02
CRD_LOG                    0000081A R      03     IBOX_SERV                    0000010A R      03
CRD_LOG_SIZE             = 000000C8               IBOX_THRESHOLD             = 0000000A
CRD_NEXT                   000001AC R      02     IBOX_TOTAL                   00000008 R      02
CRD_OFF                    00000867 R      03     IOC$BROADCAST                ********    X   03
CRD_OLD1                   00000060 R      02     IOC$GL_ADPLIST               ********    X   03
CRD_OLD2                   00000064 R      02     IPL$_ASTDEL                = 00000002
CRD_REENAB_TIME          = 0000012C               I_ERR                        00000078 R      03
CRD_THRESHOLD            = 00000064               KEEPALIVE_TIME             = 0000005A
CRD_TIMER                  0000006C R      02     KEEPALIVE_TIMER              000000B0 R      02
CRD_TOTAL                  00000068 R      02     LOGSBI                       000006B5 R      03
CSH_A_OLD1                 00000040 R      02     LOG_MCHECK                   00000638 R      03
CSH_A_OLD2                 00000044 R      02     LOG_SIZE                   = 000000E4
CSH_B_OLD1                 00000048 R      02     MBOX_1D_CPU                  0000045E R      03
CSH_B_OLD2                 0000004C R      02     MBOX_1D_DMA                  00000575 R      03
CSWP$M_C0ENA             = 00000001               MBOX_1D_OLD1                 00000054 R      02
CSWP$M_C1ENA             = 00000002               MBOX_1D_OLD2                 00000058 R      02
CSWP$M_VAL               = 00000004               MBOX_1D_SERV                 00000431 R      03
EBOX_OLD1                  00000018 R      02     MBOX_1D_THRESHOLD          = 0000000A
EBOX_OLD2                  0000001C R      02     MBOX_1D_TOTAL                0000005C R      02
EBOX_SERV                  000002E7 R      03     MBOX_FE_OLD                  00000024 R      02
EBOX_THRESHOLD           = 0000000A               MBOX_FE_PC                   00000028 R      02
EBOX_TOTAL                 00000020 R      02     MBOX_FE_PHY_ADR              00000030 R      02
ECC$REENABLE               0000086E RG     03     MBOX_FE_SERV                 0000012C R      03
EMB$B_MC_SUMCOD          = 00000010               MBOX_FE_THRESHOLD          = 00000002
EMB$C_CRBT               = 00000011               MBOX_FE_TOTAL                0000002C R      02
EMB$C_HD_LENGTH          = 00000010               MCF790$B_MCHK_CODE         = 00000004
EMB$K_CRD                = 0000000E               MCF790$C_ABUS              = 00000008
EMB$K_MC                 = 00000002               MCF790$C_ABUS_REFL         = 0000000F
```

```
MCF790$C_ABUS_WRT           = 00000004      MCF790$V_CSH_ERR            = 00000002
MCF790$C_CP_READ            = 0000000E      MCF790$V_CSH_TAG_PE         = 00000006
MCF790$C_CP_WRT             = 0000000D      MCF790$V_CSH_W              = 00000004
MCF790$C_EBOX               = 00000002      MCF790$V_CYCLE_TYP          = 0000001A
MCF790$C_EBOX_PORT          = 00000002      MCF790$V_DBL_BIT            = 00000014
MCF790$C_FBOX               = 00000001      MCF790$V_DEST_CP            = 0000001E
MCF790$C_IBOX               = 00000003      MCF790$V_EDP_PE             = 00000009
MCF790$C_MBOX_FE            = 00000004      MCF790$V_FBOX               = 0000001C
MCF790$C_NOP                = 00000000      MCF790$V_IBOX_ERR           = 0000000D
MCF790$C_OP_PORT            = 00000001      MCF790$V_MBOX_1D            = 00000007
MCF790$C_WRITE_REG          = 00000002      MCF790$V_MBOX_FE            = 0000000F
MCF790$L_EBCS               = 0000000C      MCF790$V_MBOX_INT           = 0000000E
MCF790$L_EHSR               = 00000004      MCF790$V_MUL_ERR            = 00000007
MCF790$L_ESASAV             = 0000002C      MCF790$V_NXM                = 00000003
MCF790$L_IVASAV             = 00000024      MCF790$V_SBIA               = 00000005
MCF790$L_MDECC              = 00000040      MCF790$V_SBIA_ERR           = 00000006
MCF790$L_MEAR               = 00000050      MCHK$GL_LOG                   00000693 RG      03
MCF790$L_MSTAT1             = 00000038      MCHK$M_LOG                  = 00000001
MCF790$L_MSTAT2             = 0000003C      MCHK$M_MCK                  = 00000002
MCF790$L_PC                 = 0000005C      MCHK$M_NEXM                 = 00000004
MCF790$L_PSL                = 00000060      MCHK_ABORT                  = 00000001
MCF790$L_VIBASAV            = 00000028      MCHK_EXIT                     00000097 R       03
MCF790$M_ABORTS             = 0000001E      MISC_MBOX_1D                  0000051C R       03
MCF790$M_AB_ADR_PE          = 00080000      MMG$AL_SYSPCB                 ********    X    03
MCF790$M_AB_CM_PE           = 00100000      MMG$DELWSLEX                  ********    X    03
MCF790$M_AB_DAT_PE          = 00200000      MMG$GL_MAXPFN                 ********    X    03
MCF790$M_AUTO_SHUT          = 00020000      MMG$GL_SBICONF                ********    X    03
MCF790$M_BAD_DATA           = 00400000      MMG$GL_SPTBASE                ********    X    03
MCF790$M_CPR_PE_A           = 00400000      MMG$GW_BIGPFN                 ********    X    03
MCF790$M_CPR_PE_B           = 00800000      MMG$REFCNTNEG                 ********    X    03
MCF790$M_CSH_TAG_PE         = 00000040      MMG$RELPFN                    ********    X    03
MCF790$M_CSH_TAG_W          = 00000020      M_ERR                         00000069 R       03
MCF790$M_DBL_BIT            = 00100000      OPA$UCB0                      ********    X    03
MCF790$M_ECS_PE             = 00000800      PCB$L_PHD                   = 0000006C
MCF790$M_EDP_PE             = 00000200      PFN$AB_STATE                  ********    X    03
MCF790$M_EMCR_PE            = 00001000      PFN$AB_TYPE                   ********    X    03
MCF790$M_IO_RD              = 00000002      PFN$AL_PTE                    ********    X    03
MCF790$M_MEM_WRT            = 00000004      PFN$AW_REFCNT                 ********    X    03
MCF790$M_RSRC_REM           = 00000010      PFN$AX_WSLX                   ********    X    03
MCF790$M_SBIA               = 00000020      PFN$C_PROCESS               = 00000000
MCF790$M_SBIA_ERR           = 00000040      PFN$C_SYSTEM                = 00000001
MCF790$M_TB_A_PE            = 00000200      PFN$M_BADPAG                = 00000020
MCF790$M_TB_B_PE            = 00000400      PFN$M_DELCON                = 00000010
MCF790$M_TB_TAG_PE          = 00000100      PFN$M_MODIFY                = 00000080
MCF790$M_TB_VAL_PE          = 00000800      PFN$S_PAGTYP                = 00000003
MCF790$M_USTK_PE            = 00000400      PFN$V_PAGTYP                = 00000000
MCF790$S_AB_ADPT            = 00000002      PHD$L_PCB                   = 00000078
MCF790$S_CYCLE_TYP          = 00000004      PR$_KSP                     = 00000000
MCF790$S_DEST_CP            = 00000002      PR$_P0BR                    = 00000008
MCF790$V_ABORTS             = 00000001      PR$_P1BR                    = 0000000A
MCF790$V_AB_ADPT            = 00000010      PR$_TBIA                    = 00000039
MCF790$V_AB_BAD_DAT         = 0000000E      PR790$_ACCS                 = 00000028
MCF790$V_ADR_PE             = 00000013      PR790$_CRBT                 = 00000048
MCF790$V_BAD_DATA           = 00000016      PR790$_CSWP                 = 00000042
MCF790$V_CP_IO_BUF          = 00000002      PR790$_EHSR                 = 0000004A
MCF790$V_CSH_DAT_BW         = 00000000      PR790$_LSPA                 = 0000004E
MCF790$V_CSH_DAT_NBW        = 00000003      PR790$_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$L_CR                00000000
SBIA$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_OLD1                  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
------------------                        --------------
_$255$DUA28:[SYSLOA.OBJ]790DEF.MLB;1                   5
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                        17
_$255$DUA28:[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
```

0397 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

LIOSUB750
LIS

MCHECKUV1
LIS

MCHECK730
LIS

MCHECK750
LIS

LIOSUB730
LIS

LIOSUB790
LIS

MCHECK780
LIS

MCF790
LIS

MCHECK790
LIS

LIOSUB780
LIS

LIOSUBUV1
LIS