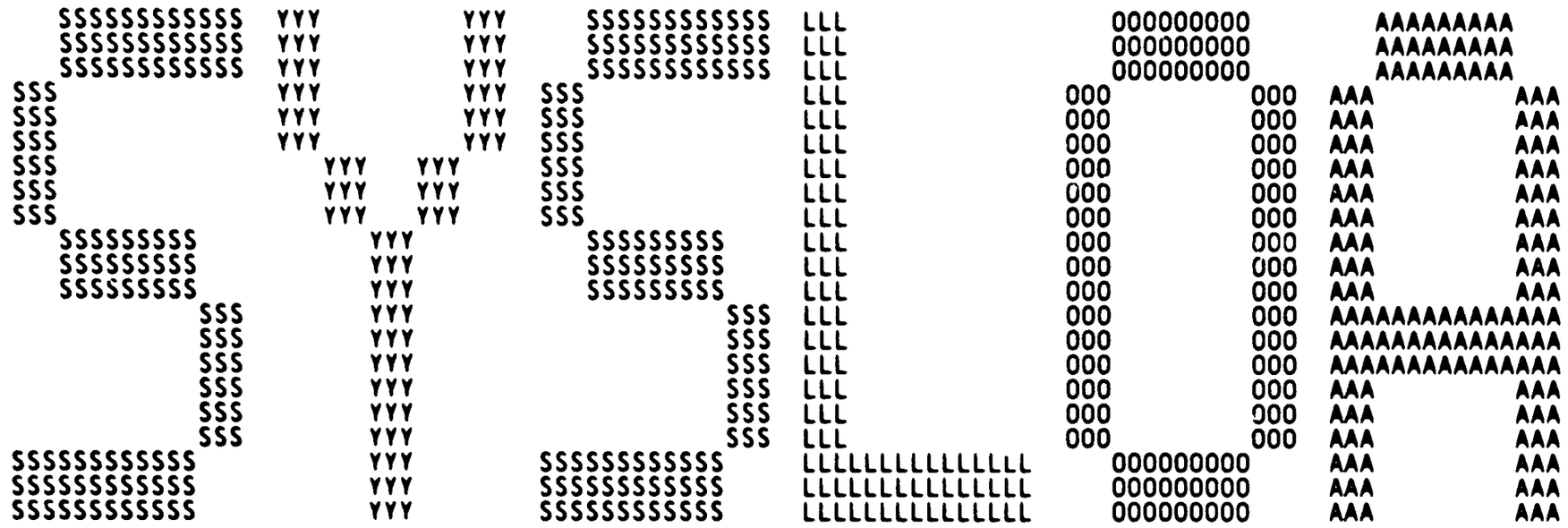
SSSSSSSSSSS YYY YYY SSSSSSSSSSS LLL 000000000 AAAAAAAAA
SSSSSSSSSSS YYY YYY SSSSSSSSSSS LLL 000000000 AAAAAAAAA
SSSSSSSSSSS YYY YYY SSSSSSSSSSS LLL 000000000 AAAAAAAAA
SSS YYY YYY SSS LLL 000 000 AAA AAA
SSS YYY YYY SSS LLL 000 000 AAA AAA
SSS YYY YYY SSS LLL 000 000 AAA AAA
SSS YYY YYY SSS LLL 000 000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

```
MM      MM    CCCCCCCC  FFFFFFFFFF  77777777      999999      000000
MM      MM    CCCCCCCC  FFFFFFFFFF  77777777      999999      000000
MMMM  MMMM  CC          FF                  77  99      99  00      00
MMMM  MMMM  CC          FF                  77  99      99  00      00
MM  MM  MM  CC          FF                  77  99      99  00    0000
MM  MM  MM  CC          FF                  77  99      99  00    0000
MM      MM  CC          FFFFFFFF          77      99999999  00  00  00
MM      MM  CC          FFFFFFFF          77      99999999  00  00  00
MM      MM  CC          FF              77              99  0000    00
MM      MM  CC          FF              77              99  0000    00
MM      MM  CC          FF            77                99  00      00      ....
MM      MM  CC          FF            77                99  00      00      ....
MM      MM    CCCCCCCC  FF          77              999999      000000      ....
MM      MM    CCCCCCCC  FF          77              999999      000000      ....

  SSSSSSSS  DDDDDDDD    LL
  SSSSSSSS  DDDDDDDD    LL
SS          DD      DD  LL
SS          DD      DD  LL
SS          DD      DD  LL
SS          DD      DD  LL
  SSSSSS    DD      DD  LL
  SSSSSS    DD      DD  LL
        SS  DD      DD  LL
        SS  DD      DD  LL
        SS  DD      DD  LL
        SS  DD      DD  LL
SSSSSSSS    DDDDDDDD    LLLLLLLLLL
SSSSSSSS    DDDDDDDD    LLLLLLLLLL
```

```
{
{ Version:      'V04-000'
{
{****************************************************************************
{*                                                                          *
{*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
{*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
{*  ALL RIGHTS RESERVED.                                                    *
{*                                                                          *
{*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
{*  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE          *
{*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER     *
{*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
{*  OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY       *
{*  TRANSFERRED.                                                            *
{*                                                                          *
{*  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE    *
{*  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT        *
{*  CORPORATION.                                                            *
{*                                                                          *
{*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS     *
{*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
{*                                                                          *
{*                                                                          *
{****************************************************************************

{++
{
{ FACILITY:  VAX/VMS CPU-dependent Code Macro Libraries
{
{ ABSTRACT:
{
{       This file contains the SDL source for 11/790 machine check frame
{       definitions.
{
{ ENVIRONMENT:
{
{       n/a
{
{--
{
{
{ AUTHOR:  Wayne Cardoza          CREATION DATE:  01-Nov-1982
{
{ MODIFIED BY:
{
{       V03-011 WMC0008         Wayne Cardoza           23-Jul-1984
{               Still more spec changes.
{
{       V03-010 WMC0007         Wayne Cardoza           08-Jul-1984
{               Assorted spec changes.
{
{       V03-009 WMC0006         Wayne Cardoza           30-May-1983
{               Minor changes and corrections.
{
{       V03-008 WMC0005         Wayne Cardoza           22-FEB-1983
```

```
{               Spec changes to MSTAT1, MSTAT2, MDECC
{
{       V03-007 WMC0004         Wayne Cardoza           08-Feb-1983
{               Rearrange EHSR
{
{       V03-006 WMC0003         Wayne Cardoza           20-Dec-1982
{               Separate PAMM code from cache bit
{
{       V03-005 WMC0002         Wayne Cardoza           24-Nov-1982
{               Add the VMS type code definitions.
{
{       V03-004 WMC0001         Wayne Cardoza           14-Nov-1982
{               Changes to MDECC, MSTAT1
{--
```

```
module $MCF790DEF;

aggregate MCF790 structure prefix MCF790$;
    SIZE longword unsigned;              /* size in bytes of frame
    EHSR_OVERLAY union;          /* error handling status register
        EHSR longword unsigned; /* entire register
        EHSR_BITS structure;
            EHSR_OVERLAY_1 union;
                MCHK_CODE byte unsigned; /* VMS puts a code here
                EHSR_BITS_1 structure;
                    SERV_TYPE bitfield mask length 3; /* VMS service type
                    FILLTA bitfield length 1 fill prefix MCF790 tag $$;
                    RSRC_REM bitfield mask; /* Resource removed from service
                    SBIA_bitfield mask; /* full SBIA log follows
                    SBIA_ERR bitfield mask; /* SBIA error summary included
                    MBOX_1D bitfield mask; /* MBOX 1D error included
                end EHSR_BITS_1;
            end EHSR_OVERLAY_T;
            TRAP_VEC bitfield mask length 8; /* trap vector
            FILLT bitfield length 1 fill prefix MCF790 tag $$;
            AUTO_SHUT bitfield mask;  /* Severe error flag
            MEAR_SAV bitfield mask;   /* meaningful to microcode
            ICS bitfield mask;        /* ICS correction
            IDRAM bitfield mask;      /* IDRAM correction
            FDRAM bitfield mask;      /* FDRAM correction
            FBACS bitfield mask;      /* FBACS correction
            FBMCS bitfield mask;      /* FBMCS correction
            IBOX_GPR bitfield mask;   /* IBOX GPR correction
            EBOX_SPBA bitfield mask;  /* EBOX SP B to A
            EBOX_SPAB bitfield mask;  /* EBOX SP A to B
            FBOX_SP bitfield mask;    /* FBOX SP correction
            FBOX bitfield mask;       /* FBOX service
            VMS_ENT bitfield mask;    /* VMS entered
            EHM_ENT bitfield mask;    /* EHM entered
            MBOX bitfield mask;       /* MBOX service
        end EHSR_BITS;
end EHSR_OVERLAY;
EVMQSAV longword unsigned;  /* virtual address - EBOX port requests
EBCS_OVERLAY union;         /* EBOX control status register
    EBCS longword unsigned; /* entire register
    EBCS_BITS structure;
        EBCS_OVERLAY_1 union;
            EBCS_BITS_2 structure;
                FILL2 bitfield fill prefix MCF790 tag $$;
                IO_RD bitfield mask;        /* IO read abort
                MEM_WRT bitfield mask;      /* memory write abort
                STA_MOD bitfield mask;      /* state modified abort
                EB_ABT bitfield mask;       /* EBOX abort
                FILL3 bitfield length 3 fill prefix MCF790 tag $$;
                WBUS_CHK bitfield mask;     /* WBUS to EDP error
                EDP_PE bitfield mask;       /* EBOX data path parity error
                USTR_PE bitfield mask;      /* EBOX microstack
                ECS_PE bitfield mask;       /* EBOX control store
                EMCR_PE bitfield mask;      /* EBOX memory control RAM
                IBOX_ERR bitfield mask;     /* IBOX hardware error
                MBOX_INT bitfield mask;     /* MBOX interrupt request
```

```
                MBOX_FE bitfield mask;      /* MBOX fatal error
            end EBCS_BITS_2;
            EBCS_BITS_3 structure;
                FILL2A bitfield fill prefix MCF790 tag $$;
                ABORTS bitfield mask length 4;
                FILL3A bitfield length 3 fill prefix MCF790 tag $$;
                DIAG_ERR bitfield mask;     /* diagnostic error flag
            end EBCS_BITS_3;
        end EBCS_OVERLAY_T;
        FILL4 bitfield length 4 fill prefix MCF790 tag $$;
        PME bitfield mask;          /* performance measurement enable
        FILL5 bitfield length 6 fill prefix MCF790 tag $$;
        ICS_EF bitfield mask;       /* IBOX control store error
        IDRAM_EF bitfield mask;     /* IBOX dispatch RAM error
        FBMCS_EF bitfield mask;     /* FBOX FBM control store error
        FBACS_EF bitfield mask;     /* FBOX FBA control store error
        FDRAM_EF bitfield mask;     /* FBOX dispatch RAM error
    end EBCS_BITS;
end EBCS_OVERLAY;
EDPSR_OVERLAY union;        /* EBOX data path status register
    EDPSR longword unsigned;/* entire register
    EDPSR_BITS structure;
        B_RAM_PE bitfield mask;     /* scratchpad to BMUX error
        A_WBUS_PE bitfield mask;    /* WBUS to AMUX error
        A_RAM_PE bitfield mask;     /* scratchpad to AMUX error
        OPER_CHK bitfield mask;     /* operand parity error
        FILL51 bitfield fill prefix MCF790 tag $$;
        RSLT_CHK bitfield mask;     /* result parity error
        B_OPBUS bitfield mask;      /* OPBUS to BMUX error
        B_WBUS bitfield mask;       /* WBUS to BMUX error
        EDP_MISC bitfield mask;     /* misc source parity error
        FILL6 bitfield length 2 fill prefix MCF790 tag $$;
        WREG bitfield mask;         /* W register parity error
        VMQ_BYTE bitfield mask length 4; /* VMQ byte in error
        FILL7 bitfield length 8 fill prefix MCF790 tag $$;
        AMX_BYTE bitfield mask length 4; /* AMUX byte in error
        BMX_BYTE bitfield mask length 4; /* BMUX byte in error
    end EDPSR_BITS;
end EDPSR_OVERLAY;
CSLINT_OVERLAY union;       /* console/interrupt register
    CSLINT longword unsigned; /* entire register
    CSLINT_BITS structure;
        CADR bitfield mask length 6; /* console bus address
        CWRT bitfield mask;         /* console bus write
        CCLK bitfield mask;         /* console bus clock
        CDAT bitfield mask length 8; /* console bus data
        IPR bitfield mask length 4; /* interrupt priority request level
        INT_SRC bitfield mask;      /* IPR due to internal source
        IOA bitfield mask length 2; /* I/O adapter with highest IPR
        CSL_TTX bitfield mask;      /* console terminal transmit
        CSL_TRX bitfield mask;      /* console terminal receive
        CSL_RL bitfield mask;       /* console RL
        INT_TMR bitfield mask;      /* interval timer interrupt
        INT_MBOX bitfield mask;     /* MBOX interrupt
        CPU_PF bitfield mask;       /* CPU powerfail interrupt
        CSL_HP bitfield mask;       /* console halt pending
```

```
        end CSLINT_BITS;
    end CSLINT_OVERLAY;
    IBESR_OVERLAY union;        /* IBOX error/status register
        IBESR longword unsigned; /* entire register
        IBESR_BITS structure;
            FILL8 bitfield length 8 fill prefix MCF790 tag $$;
            UOP_SEL bitfield mask length 2; /* OP BUS data source
            SRC_IMD bitfield mask;      /* OP BUS source was IMD
            UTPR bitfield mask length 3; /* processor port causing microtrap
            FILL9 bitfield length 7 fill prefix MCF790 tag $$;
            ICS_PE bitfield mask;       /* IBOX control store parity error
            IDRAM_PE bitfield mask;     /* DRAM
            IAMUX_PE bitfield mask;     /* AMUX whren GPR selected
            RLOG_PE bitfield mask;      /* unwinding RLOG
            IBUF_PE bitfield mask;      /* error on byte-1, byte-0, or R-mode finder
            IBMUX_PE bitfield mask;     /* output of ALU BMUX
            RSV_MODE bitfield mask;     /* reserved mode
            IWBUS_PE bitfield mask;     /* WBUS error detected by IBOX
            IAMUX_EC bitfield mask length 2; /*
        end IBESR_BITS;
    end IBESR_OVERLAY;
    EBXWD1 longword unsigned;   /* EBOX write data 1
    EBXWD2 longword unsigned;   /* EBOX write data 2
    IVASAV longword unsigned;   /* virtual address for OP port requests
    VIBASAV longword unsigned;  /* virtual address of next IBUF port request
    ESASAV longword unsigned;   /* PC during EBOX execution and result storage
    ISASAV longword unsigned;   /* PC of instruction OP port working on
    CPC longword unsigned;      /* PC of instruction evaluated in IBUFFER
    MSTAT1_OVERLAY union;       /* MBOX status register 1
        MSTAT1 longword unsigned;       /* entire register
        MSTAT1_BITS structure;
            CSH_DAT_BW bitfield mask;   /* datapath parity error on byte write
            ARR_CYCL bitfield mask;     /* error detectod on array refill to cache
            CSH_ERR bitfield mask;      /* indicates which cache had the error
            CSH_DAT_NBW bitfield mask;  /* datapath parity error, non byte write
            WRT_DAT_PE bitfield mask length 4; /* MDBUS parity error on write data
            TB_TAG_PE bitfield mask;    /* error on address tag
            TB_A_PE bitfield mask;      /* error on PTE
            TB_B_PE bitfield mask;      /* error on PTE
            TB_VAL_PE bitfield mask;    /* error in valid bit
            CSH_HIT bitfield mask length 4; /* cache hit/miss history
            AB_ADPT bitfield mask length 2; /* ABUS adapter in error
            AB_CYCL bitfield mask;      /* ABUS cycle in error
            AB_ADR_PE bitfield mask;    /* ABUS physical address in error
            AB_CM_PE bitfield mask;     /* ABUS cntrl/mask parity error
            AB_DAT_PE bitfield mask;    /* ABUS data parity error
            CPR_PE_A bitfield mask;     /* cycle parameter RAM error (A)
            CPR_PE_B bitfield mask;     /* cycle parameter RAM error (B)
            WDCNT bitfield mask length 2; /* longword in error
            CYCLE_TYP bitfield mask length 4; /* MBOX cycle type
            DEST_CP bitfield mask length 2; /* port being serviced
        end MSTAT1_BITS;
    end MSTAT1_OVERLAY;
    MSTAT2_OVERLAY union;       /* MBOX status register 2
        MSTAT2 longword unsigned; /* entire register
        MSTAT2_BITS structure;
```

```
            FILL95 bitfield length 1 fill prefix MCF790 tag $$;
            MBOX_LCK bitfield mask;     /* error while lock asserted
            CP_IO_BUF bitfield mask;    /* error on CPU to IO request
            NXM bitfield mask;          /* non-existent memory
            CSH_W bitfield mask;        /* selected cache entry was modified
            CSH_TAG_W bitfield mask;    /* error in cache written bit
            CSH_TAG_PE bitfield mask;   /* error in cache tag
            MUL_ERR bitfield mask;      /* multiple MBOX errors
            SBIA_STAT bitfield mask length 6; /* SBIA diagnostic status
            AB_BAD_DAT bitfield mask;   /* ABUS bad data flag received
            SBIA_CPBW bitfield mask;    /* SBIA error was on CP byte write
            PAMM_DATA bitfield mask length 4; /* PAMM code
            PAMM_CACHE bitfield mask;   /* PAMM cache disable bit
        end MSTAT2_BITS;
    end MSTAT2_OVERLAY;
    MDECC_OVERLAY union;        /* MBOX data ECC register
        MDECC longword unsigned; /* entire register
        MDECC_BITS structure;
            ECC_DIAG bitfield mask length 8; /* force errors
            FILL115 bitfield length 1 fill prefix MCF790 tag $$;
            SYNDRM bitfield mask length 6; /* error data syndrome
            PAR_INV bitfield mask;      /* indicates parity is being inverted
            FILL11 bitfield length 3 fill prefix MCF790 tag $$;
            ADR_PE bitfield mask;       /* data address parity error
            DBL_BIT bitfield mask;      /* double bit error
            SNG_ERR bitfield mask;      /* single bit error
            BAD_DATA bitfield mask;     /* bad data flag
            DATA_MUL bitfield mask;     /* multiple errors
        end MDECC_BITS;
    end MDECC_OVERLAY;
    MERG longword unsigned;     /* MBOX error generator register
    CSHCTL_OVERLAY union;       /* MBOX cache control register
        CSHCTL longword unsigned; /* entire register
        CSHCTL_BITS structure;
            CSH_0_ENB bitfield mask;    /* cache 0 enable
            CSH_1_ENB bitfield mask;    /* cache 1 enable
            FRC_HIT bitfield mask;      /* force cache hit
            FRC_MISS bitfield mask;     /* force cache miss
        end CSHCTL_BITS;
    end CSHCTL_OVERLAY;
    MEDR longword unsigned;     /* data word used during error
    MEAR longword unsigned;     /* physical address in latch during error
    FBXERR_OVERLAY union;       /* FBOX error register
        FBXERR longword unsigned;       /* entire register
        FBXERR_BITS structure;
            FBOX_ERR bitfield;          /* There is an error - rest of bits valid
            FILL12 bitfield length 1 fill prefix MCF790 tag $$;
            TEST bitfield mask;         /* error during self test
            FILL13 bitfield length 11 fill prefix MCF790 tag $$;
            DATA_TYP bitfield mask length 2; /* data type during error
            FILL14 bitfield length 1 fill prefix MCF790 tag $$;
            FBOX_GPR bitfield mask;     /* error reading scratchpad
            FBOX_SLF bitfield mask;     /* error during self test
            FBOX_DRAM bitfield mask;    /* DRAM parity error
            FBOX_FBA_CS bitfield mask;  /* error in adder control store
            FBOX_FBM_CS bitfield mask;  /* error in multiplier control store
```

```
        end FBXERR_BITS;
    end FBXERR_OVERLAY;
    CSES longword unsigned;     /* control store error status register
    PC longword unsigned;
    PSL longword unsigned;

                                /* MBOX cycle types
    constant(
        NOP,
        READ_REG,                       /* read register
        WRITE_REG,                      /* write register
        WRITE_BAK,                      /* write back
        ABUS_WRT,                       /* ABUS array write
        DATA_COR,                       /* data correction
        CLR_CSH,                        /* clear cache
        TB_PROBE,                       /* TB probe
        ABUS,                           /* ABUS
        CP_REFL,                        /* CP refill
        INVAL_TB,                       /* invalidate TB
        TB_CYCLE,                       /* TB cycle
        CP_BYT_WRT,                     /* CP byte write
        CP_WRT,                         /* CP write
        CP_READ,                        /* CP read
        ABUS_REFL                       /* ABUS refill
    ) equals 0 increment 1 prefix MCF790 tag $C;

                                /* DEST CP (port) codes
    constant(
        IBF_PORT_0,                     /* IBUF port
        OP_PORT,                        /* OP fetch port
        EBOX_PORT,                      /* EBOX port
        IBF_PORT_3                      /* IBUF port
    ) equals 0 increment 1 prefix MCF790 tag $C;

                                /* VMS machine check service codes
    constant(
        FBOX,                           /* FBOX
        EBOX,                           /* EBOX
        IBOX,                           /* IBOX
        MBOX_FE                         /* MBOX fatal error
    ) equals 1 increment 1 prefix MCF790 tag $C;

end MCF790;
end_module $MCF790DEF;
```

ADPSUB780
LIS

ACKMSG
LIS

MCF790
SDL

MCDEF
MDL

ADPERR750
LIS

ADPSUB730
LIS

CSPDEF
SDL

CLUMBX
SDL

ADPERR780
LIS

ADPSUB750
LIS

CLUSTRMAC
MAR

CLUSTER
SDL