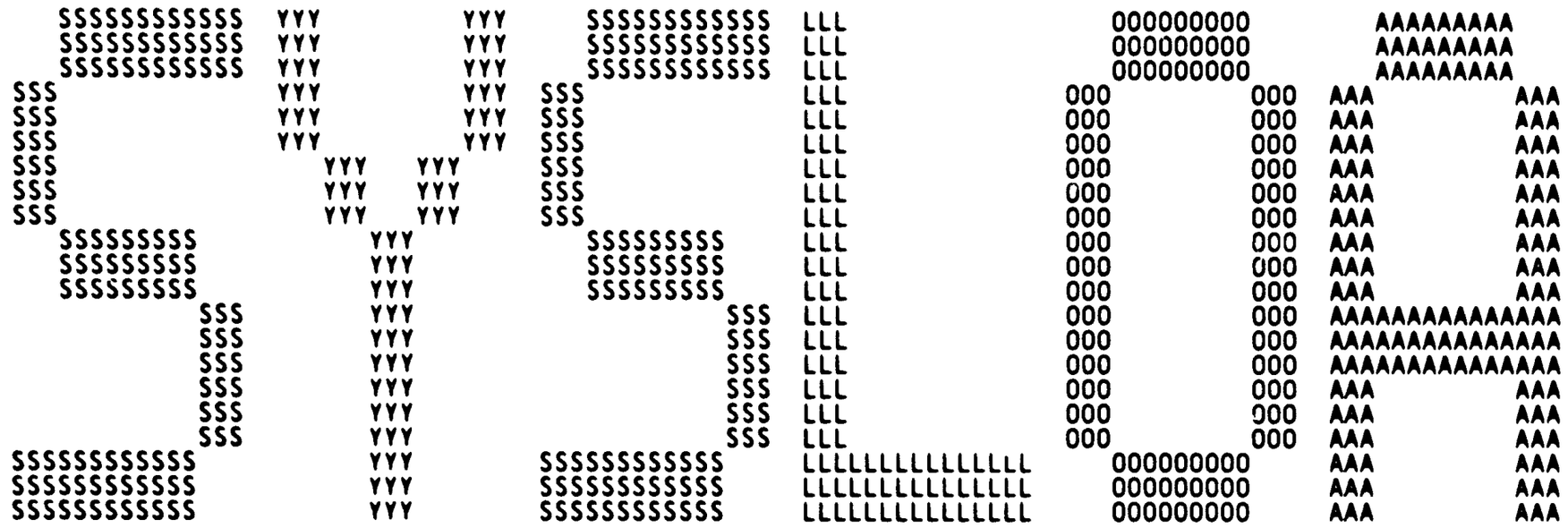
SSSSSSSSSSS YYY YYY SSSSSSSSSSS LLL 000000000 AAAAAAAAA
SSSSSSSSSSS YYY YYY SSSSSSSSSSS LLL 000000000 AAAAAAAAA
SSSSSSSSSSS YYY YYY SSSSSSSSSSS LLL 000000000 AAAAAAAAA
SSS YYY YYY SSS LLL 000 000 AAA AAA
SSS YYY YYY SSS LLL 000 000 AAA AAA
SSS YYY YYY SSS LLL 000 000 AAA AAA
SSS YYY YYY SSS LLL 000 000 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

```
  CCCCCCCC    SSSSSSSS  PPPPPPPP    DDDDDDDD    EEEEEEEEEE  FFFFFFFFFF
  CCCCCCCC    SSSSSSSS  PPPPPPPP    DDDDDDDD    EEEEEEEEEE  FFFFFFFFFF
CC            SS        PP      PP  DD      DD  EE          FF
CC            SS        PP      PP  DD      DD  EE          FF
CC            SS        PP      PP  DD      DD  EE          FF
CC            SS        PP      PP  DD      DD  EE          FF
CC              SSSSSS  PPPPPPPP    DD      DD  EEEEEEEE    FFFFFFFF
CC              SSSSSS  PPPPPPPP    DD      DD  EEEEEEEE    FFFFFFFF
CC                    SS PP        DD      DD  EE          FF
CC                    SS PP        DD      DD  EE          FF
CC                    SS PP        DD      DD  EE          FF              ....
CC                    SS PP        DD      DD  EE          FF              ....
  CCCCCCCC  SSSSSSSS    PP        DDDDDDDD    EEEEEEEEEE  FF              ....
  CCCCCCCC  SSSSSSSS    PP        DDDDDDDD    EEEEEEEEEE  FF              ....

  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
```

```
{ module        $CSPDEF;
{
{ 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 Cluster Server Process
{*
{* ABSTRACT:     CSP Internal Data Structure Definitions
{*
{* AUTHOR:       Paul R. Beck
{*
{* DATE:          3-MAR-1983     Last Edit:  29-JUL-1983 17:47:11
{*
{* REVISION HISTORY:
{
{       V03-007 ADE0002         Alan D. Eldridge        28-Feb-1983
{               Add CSP$_LOCAL and CSD$K_QUORUM.
{
{       V03-006 ADE0001         Alan D. Eldridge         7-Dec-1983
{               Add CLX$A_P0_CSD and CLX$A_S0_CSD.  Removed CLX$W_NETCHAN.
{               Add CLX$L_P1 thru P4 for use by client code.
{
{               Add CSD$W_STS, remove CSD$W_NET_CHAN, and rearrange fields
{               for proper alignment.
{
{               Add CLX$V_RESUME_REQ and changed CLX$V_MUTEX semantics.
{
{               Add $CSPDEF codes.
{
{       V03-005 PRB0249         Paul Beck                9-Sep-1983  18:11
{               Add CSD$L_P1 : CSD$L_P8 default offset definitions.
{
{       V03-004 CWH3001         CW Hobbs                        30-Jul-1983
```

```
{               Add OPCOM and MOUNT clients.
{
{       V03-003 PRB0232         Paul Beck               14-JUL-1983 11:36
{               Add support for builtin stack in CLX structure.
{
{       V03-002 PRB0202         Paul Beck               19-MAY-1983 22:39:49
{               Add client constant for RCP control.
{               Change "CTX" to "CLX".
{
{       V03-001 JLV0248         Jake VanNoy             29-APR-1983
{               Add offsets for BRKTHRU. Add client constants.
{
{*--
```

```
MODULE  $CSPDEF ;
/*
/* CSP - Codes used for communication between the CSP process and the loadable
/*       Exec code
/*

    CONSTANT
            (nop                        /* No-op.  Should never be used
            ,init                       /* Initialize loadable code and data
            ,abort                      /* CSP process is gone or going away
            ,badcsd                     /* Something wrong with CSD structure
            ,done                       /* Block transfer exchange has finished
            ,reply                      /* Send a block transfer and terminate
            ,reject                     /* Don't engage in block transfer
            ,local                      /* Request to pass local CSD to CSP
            ) EQUALS 0 INCREMENT 1
              PREFIX csp TAG $ ;

    CONSTANT csp$k_max_flwctl EQUALS    8 ;  /* Maximum allowed active requests
    CONSTANT csp$k_max_csdlng EQUALS 4096 ;  /* Maximum allowed CSD length

END_MODULE ;
```

MCF

end
end

```
MODULE  $CLXDEF;            /*
/*+
/* CLX - Scheduling context for CSP
/*
/* Registers are saved by dint of the CALL to the routine which
/* creates the CLX block, and are restored by the RET. R0 and R1
/* must be saved and restored explicitly.
/*
/*-
AGGREGATE $clxdef STRUCTURE PREFIX clx$;

    flink       ADDRESS;                /* forward link
    blink       ADDRESS;                /* reverse link
    size        WORD UNSIGNED;          /* size of structure
    type        BYTE UNSIGNED;          /* structure type (DYN$C_xxx)

    flags_overlay UNION;
        flags   BYTE UNSIGNED;          /* Flags
        flags_bits      STRUCTURE;      /*
            queued      BITFIELD MASK;  /* CLX queue linkage in used
            mutex       BITFIELD MASK;  /* mutex between RESUME and WAIT routine
            resume_req  BITFIELD MASK;  /* signal flag between RESUME and WAIT
            local_stack BITFIELD MASK;  /* local stack in use.
            filler      BITFIELD LENGTH 8-^ FILL;
        END     flags_bits;
    END         flags_overlay;

    s0_csd      ADDRESS;                /* Pointer to S0 space CSD
    p0_csd      ADDRESS;                /* Pointer to P0 space CSD
    P1          LONGWORD UNSIGNED;      /* Parameter for use by client code
    P2          LONGWORD UNSIGNED;      /* Parameter for use by client code
    P3          LONGWORD UNSIGNED;      /* Parameter for use by client code
    P4          LONGWORD UNSIGNED;      /* Parameter for use by client code
    index       LONGWORD UNSIGNED;      /* context ID for debugging purposes
    r0          LONGWORD UNSIGNED;      /* saved R0
    r1          LONGWORD UNSIGNED;      /* saved R1

    iosb_overlay    UNION;
        iosb        QUADWORD UNSIGNED; /* I/O status block
        iosb_structure  STRUCTURE;
            iosb_stat   WORD UNSIGNED;      /* IOSB completion status
            iosb_size   WORD UNSIGNED;      /* IOSB size of completed operation
            iosb_devdata LONGWORD UNSIGNED; /* IOSB device-dependent data
        END     iosb_structure;
    END         iosb_overlay;

    stacksize   LONGWORD UNSIGNED;      /* size of saved stack + registers
    stack       ADDRESS;                /* addr of saved stack area
    stackblock  STRUCTURE;              /* allow local stack of 64 longwords
        local_stack     BYTE UNSIGNED
                        DIMENSION 256 TAG B;
    END stackblock;

    CONSTANT local_stack EQUALS 256;
    CONSTANT "length" EQUALS .;         /* length of data structure
```

```
END $clxdef;
END_MODULE;
```

```
MODULE  $CSDDEF;            /*
/**
/* CSD - Cluster Server Data
/*
/* Defines client data to send to the cluster server in the indicated node.
/*-
AGGREGATE $csddef STRUCTURE PREFIX csd$;

    messages    UNION;

    maincsd STRUCTURE;

    flink       ADDRESS;                /* forward link
    blink       ADDRESS;                /* reverse link
    size        WORD UNSIGNED;          /* size of structure
    type        BYTE UNSIGNED;          /* structure type    (DYN$C_CLU)
    subtype     BYTE UNSIGNED;          /* structure subtype (DYN$C_CSD)
    constant    DYN$C_CSD equals 100 prefix "" tag ""; /* *** TEMPORARY ***
/*
/* Caller-supplied data
/*
    code        WORD UNSIGNED;          /* client code - who gets the msg in CSP
    csid_overlay     UNION;
        csid            LONGWORD UNSIGNED; /* Cluster ID
        csid_structure  STRUCTURE;
            csid_index  WORD UNSIGNED;     /* index into CLU$GL_CLUSVEC
            csid_seq    WORD UNSIGNED;     /* sequence number
        END     csid_structure;
    END csid_overlay;

    sendlen     LONGWORD UNSIGNED;      /* length of send buffer
    sendoff     LONGWORD UNSIGNED;      /* offset (self relative) to send buffer
    recvlen     LONGWORD UNSIGNED;      /* length of recv buffer
    recvoff     LONGWORD UNSIGNED;      /* offset (self relative) to recv buffer
    astadr      ADDRESS;                /* address of caller AST
    user_astadr ADDRESS;                /* address of caller's caller's AST
    user_astprm LONGWORD UNSIGNED;      /* param...
    user_iosb   ADDRESS;                /* caller's caller's IOSB
    user_efn    BYTE UNSIGNED;          /*
    efn         BYTE UNSIGNED;
/*
/* Internal data (filled in by EXE$ALLOC_CSD)
/*
    sts_overlay     UNION;
        sts             WORD UNSIGNED;  /* Status flags
        sts_bits        STRUCTURE ;
            sts_rem     BITFIELD MASK ; /* Request is from remote node
        END     sts_bits ;
    END         sts_overlay ;

    ipid_overlay     UNION;
        ipid            LONGWORD UNSIGNED; /* Local Process ID
        ipid_structure  STRUCTURE;
            ipid_index  WORD UNSIGNED;     /* index
            ipid_seq    WORD UNSIGNED;     /* sequence number
        END     ipid_structure;
```

```
    END         ipid_overlay;

    iosb_overlay     UNION;
        int_iosb         QUADWORD UNSIGNED; /* Internal I/O status block
        iosb_structure   STRUCTURE;
            iosb_stat    WORD UNSIGNED;     /* IOSB completion status
            iosb_size    WORD UNSIGNED;     /* IOSB size of completed operation
            iosb_devdata LONGWORD UNSIGNED; /* IOSB device-dependent data
        END     iosb_structure;
    END         iosb_overlay;

    procpriv    QUADWORD UNSIGNED;      /* process privileges of caller
    procuic     LONGWORD UNSIGNED;      /* UIC of caller
    imgcnt      LONGWORD UNSIGNED;      /* used to detect new image
/*
/* Start of caller-supplied message
/*
    CONSTANT    length EQUALS .;        /* start of buffers
/*
/* Define some default of  ets for client-supplied data
/*
    client_data UNION;
        data    BYTE UNSIGNED TAG "AB"; /* ...
        default_data STRUCTURE;         /* allow clients to specify...
            p1  LONGWORD UNSIGNED;      /* ... CSD$L_P1 - CSD$L_P8
            p2  LONGWORD UNSIGNED;      /* ... so they don't need to edit this
            p3  LONGWORD UNSIGNED;      /* ... file.
            p4  LONGWORD UNSIGNED;
            p5  LONGWORD UNSIGNED;
            p6  LONGWORD UNSIGNED;
            p7  LONGWORD UNSIGNED;
            p8  LONGWORD UNSIGNED;
        END     default_data;
    END         client_data;

    END maincsd;
/*
/* Client codes
/*
    CONSTANT (                          /* caller codes
        test,                           { test vector
        brkthru,                        { break though write
        rcpc,                           { RCP control
        jnl_master,                     { Journal failover (remastering)
        opcom,                          { Operator communications
        mount,                          { Cluster mount
        quorum                          { Quorum disk support
        ) EQUALS 1 INCREMENT 1;

/*
/* $BRKTHRU message definitions
/*

    brkthru STRUCTURE;

        brk_carcon      LONGWORD UNSIGNED;      /* carriage control
```

```
        brk_flags       LONGWORD UNSIGNED;      /* flags
        brk_sendto      CHARACTER LENGTH 16;    /* counted string
        brk_sndtyp      WORD UNSIGNED;          /* send to value
        brk_reqid       WORD UNSIGNED;          /* class request id
        brk_timout      WORD UNSIGNED;          /* seconds of timeout
        brk_msglen      WORD UNSIGNED;          /* message length
        brk_msgbuf      CHARACTER LENGTH 0;     /* text string

    END brkthru;

/*
/* ... next caller def goes here...
/*

    END messages;       /* end of main UNION

END $csddef;

END_MODULE;
```

ADPSUB780
LIS

ACKMSG
LIS

MCF790
SDL

MCDEF
MDL

ADPERR750
LIS

ADPSUB730
LIS

CSPDEF
SDL

CLUMBX
SDL

ADPERR780
LIS

ADPSUB750
LIS

CLUSTRMAC
MAR

CLUSTER
SDL