From: John A. Gallant
Digital Equipment Corp.
Open Systems Group
Nashua NH

X3T9.2/91-~~XX~~ 46 Rev 1

To: John Lohmeyer X3T9.2 Committee Chairman

Date: April 16, 1991
Subject: Changes and additions for the Common Access Method Doc, X3T9.2/90-186

1) Changing offsets for Peripheral Driver Pointer and Callback on Completion.

I would like to recommend that the Peripheral Driver Pointer field be swapped with the Callback on Completion field. This places the "bits" from the pointer field down below some "flags" fields in the other CCBs. This is a big help to the Peripheral Device drivers that reuse their allocated CCBs.

9.1 CAM Control Block to Request I/O

TABLE 9-1 SCSI I/O REQUEST CCB

Size	Dir	SCSI I/O Request
		- - - - - - OSD - - - - - -
1	O	LUN
4	O	CAM Flags (OSD)
4	O	Callback on Completion
4	O	Next CCB Pointer
4	O	Request Mapping Information (OSD)
4	O	Peripheral Driver Pointer
4	O	SG List/Data Buffer Pointer
4	O	Data Transfer Length
4	O	Sense Info Buffer Pointer
n	O	Private Data

2) Adding a field for the Autosense transfer residual count.

There is currently a field for the residual data transfer in the SCSI IO CCB. I would like to recommend that a single byte field for the residual count for the autosense transfer be added. Like the residual data transfer field this will be a twos compliment number for the number of autosense bytes that were not transferred. I have also added the word "Data" to the residual data transfer field to differentiate them.

TABLE 9-1 SCSI I/O REQUEST CCB

Size	Dir	SCSI I/O Request
		- - - - - - OSD - - - - - -
4	O	Sense Info Buffer Pointer
1	O	Sense Info Buffer Length
1	O	CDB Length
2	O	Number of Scatter/Gather entries
4		reserved (OSD)
1	I	SCSI Status
2		reserved (OSD)
1	I	Sense Info Residual Length
4	I	Data Transfer Residual Length
12	O	CDB
		- - - - - - OSD - - - - - -
4	O	Timeout Value
n	O	Private Data

The following changes will be needed in the description sections.

9.1.19 Data Transfer Residual Length

This field contains the difference in twos complement form of the number of data bytes transferred by the HBA compared with the number of bytes requested by the CCB.

9.1.23 Sense Info Residual Length

This field contains the difference in twos complement form of the number of sense bytes transferred by the HBA compared with the number of sense bytes requested in the Sense Info Buffer Length.

3) I would like to present a new C include file to replace the current one in Annex C. This include file represents the CAM document at Rev. 2.3.

```
/* ------------------------------------------------------------------------ */
/* cam.h                    Version 1.08                    Apr. 24, 1991 */
/*  This file contains the definitions and data structures for the CAM
    Subsystem interface.  The contents of this file should match what
    data structures and constants that are specified in the CAM document,
    X3T9.2/90-186 Rev 2.3 that is produced by the SCSI-2 committee.
```

Modification History

Version	Date	Who	Reason
1.00	03/28/90	jag	Creation date. Just to start the process.
1.01	05/23/90	jag	Modified for Rev. 1.7 April 1990
1.02	06/05/90	jag	Minor mods, added common prefix to all the structure fields. Added targer ID and LUN to the header.
1.03	10/18/90	jag	Updated to Rev. 2.1 October 1990
1.04	11/05/90	jag	Updated to Rev. 2.2 November 1990
1.05	11/07/90	jag	Changed u[char,short,long] to u_[char,short,long] for more SV,BSD consistancy. Replaced "int" in the CAM_EDT_ENTRY structure.
1.06	12/10/90	jag	Added SUCCESS/FAILURE to the OSD sect. Moved SIM_ENTRY to the main area. Made the async field in EDT a pointer. Forces long word alignment to the SIM private data space in the IO CCB.
1.07	03/22/91	jag	Updated to Rev. 2.3 January 1991. Added reserved fields, Target mode and HBA Engine defines.
1.08	04/24/91	jag	Added a formal struct label to all the typedefed structures. Made the first

```
                                        element in the S/G list a char pointer.
                                        Fixed CCB_PATHINQ, removed the u_long
                                        cam_feature_flags.
*/

/* ---------------------------------------------------------------------- */

/* Defines for the XPT function codes, Table 8-2 in the CAM spec. */

/* Common function commands, 0x00 - 0x0F */
#define XPT_NOOP          0x00    /* Execute Nothing */
#define XPT_SCSI_IO       0x01    /* Execute the requested SCSI IO */
#define XPT_GDEV_TYPE     0x02    /* Get the device type information */
#define XPT_PATH_INQ      0x03    /* Path Inquiry */
#define XPT_REL_SIMQ      0x04    /* Release the SIM queue that is frozen */
#define XPT_SASYNC_CB     0x05    /* Set Async callback parameters */
#define XPT_SDEV_TYPE     0x06    /* Set the device type information */

/* XPT SCSI control functions, 0x10 - 0x1F */
#define XPT_ABORT         0x10    /* Abort the selected CCB */
#define XPT_RESET_BUS     0x11    /* Reset the SCSI bus */
#define XPT_RESET_DEV     0x12    /* Reset the SCSI device, BDR */
#define XPT_TERM_IO       0x13    /* Terminate the I/O process */

/* HBA engine commands, 0x20 - 0x2F */
#define XPT_ENG_INQ       0x20    /* HBA engine inquiry */
#define XPT_ENG_EXEC      0x21    /* HBA execute engine request */

/* Target mode commands, 0x30 - 0x3F */
#define XPT_EN_LUN        0x30    /* Enable LUN, Target mode support */
#define XPT_TARGET_IO     0x31    /* Execute the target IO request */

#define XPT_FUNC          0x7F    /* TEMPLATE */
#define XPT_VUNIQUE       0x80    /* All the rest are vendor unique commands */

/* ---------------------------------------------------------------------- */

/* General allocation length defines for the CCB structures. */

#define IOCDBLEN          12      /* Space for the CDB bytes/pointer */
#define VUHBA             14      /* Vendor Unique HBA length */
#define SIM_ID            16      /* ASCII string len for SIM ID */
#define HBA_ID            16      /* ASCII string len for HBA ID */
#define SIM_PRIV          50      /* Length of SIM private data area */

/* Structure definitions for the CAM control blocks, CCB's for the
subsystem. */

/* Common CCB header definition. */
typedef struct ccb_header
{
    struct ccb_header *my_addr; /* The address of this CCB */
    u_short cam_ccb_len;        /* Length of the entire CCB */
    u_char cam_func_code;       /* XPT function code */
    u_char cam_status;          /* Returned CAM subsystem status */
    u_char cam_hrsvd0;          /* Reserved field, for alignment */
    u_char cam_path_id;         /* Path ID for the request */
    u_char cam_target_id;       /* Target device ID */
    u_char cam_target_lun;      /* Target LUN number */
    u_long cam_flags;           /* Flags for operation of the subsystem */
} CCB_HEADER;

/* Common SCSI functions. */

/* Union definition for the CDB space in the SCSI I/O request CCB */
typedef union cdb_un
{
    u_char *cam_cdb_ptr;            /* Pointer to the CDB bytes to send */
    u_char cam_cdb_bytes[ IOCDBLEN ];   /* Area for the CDB to send */
} CDB_UN;

/* Get device type CCB */
typedef struct ccb_getdev
{
    CCB_HEADER cam_ch;              /* Header information fields */
    char *cam_inq_data;             /* Ptr to the inquiry data space */
    u_char cam_pd_type;             /* Periph device type from the TLUN */
} CCB_GETDEV;

/* Path inquiry CCB */
typedef struct ccb_pathinq
{
    CCB_HEADER cam_ch;              /* Header information fields */
    u_char cam_version_num;         /* Version number for the SIM/HBA */
    u_char cam_hba_inquiry;         /* Mimic of INQ byte 7 for the HBA */
    u_char cam_target_sprt;         /* Flags for target mode support */
    u_char cam_hba_misc;            /* Misc HBA feature flags */
    u_short cam_hba_eng_cnt;        /* HBA engine count */
    u_char cam_vuhba_flags[ VUHBA ];/* Vendor unique capabilities */
    u_long cam_sim_priv;            /* Size of SIM private data area */
    u_long cam_async_flags;         /* Event cap. for Async Callback */
    u_char cam_hpath_id;            /* Highest path ID in the subsystem */
    u_char cam_initiator_id;        /* ID of the HBA on the SCSI bus */
    u_char cam_prsvd0;              /* Reserved field, for alignment */
    u_char cam_prsvd1;              /* Reserved field, for alignment */
    char cam_sim_vid[ SIM_ID ];     /* Vendor ID of the SIM */
    char cam_hba_vid[ HBA_ID ];     /* Vendor ID of the HBA */
    u_char *cam_osd_usage;          /* Ptr for the OSD specific area */
} CCB_PATHINQ;

/* Release SIM Queue CCB */
typedef struct ccb_relsim
{
    CCB_HEADER cam_ch;              /* Header information fields */
} CCB_RELSIM;

/* SCSI I/O Request CCB */
typedef struct ccb_scsiio
{
    CCB_HEADER cam_ch;              /* Header information fields */
    u_char *cam_pdrv_ptr;           /* Ptr used by the Peripheral driver */
    CCB_HEADER *cam_next_ccb;       /* Ptr to the next CCB for action */
    u_char *cam_req_map;            /* Ptr for mapping info on the Req. */
    void (*cam_cbfcnp)();           /* Callback on completion function */
    u_char *cam_data_ptr;           /* Pointer to the data buf/SG list */
    u_long cam_dxfer_len;           /* Data xfer length */
    u_char *cam_sense_ptr;          /* Pointer to the sense data buffer */
    u_char cam_sense_len;           /* Num of bytes in the Autosense buf */
    u_char cam_cdb_len;             /* Number of bytes for the CDB */
    u_short cam_sglist_cnt;         /* Num of scatter gather list entries */
    u_long cam_osd_rsvd0;           /* OSD Reserved field, for alignment */
    u_char cam_scsi_status;         /* Returned scsi device status */
    u_char cam_osd_rsvd1[3];        /* OSD Reserved field, for alignment */
    long cam_resid;                 /* Transfer residual length: 2's comp */
    CDB_UN cam_cdb_io;              /* Union for CDB bytes/pointer */
    u_long cam_timeout;             /* Timeout value */
    u_char *cam_msg_ptr;            /* Pointer to the message buffer */
    u_short cam_msgb_len;           /* Num of bytes in the message buf */
    u_short cam_vu_flags;           /* Vendor unique flags */
    u_char cam_tag_action;          /* What to do for tag queuing */
    u_char cam_iorsvd0[3];          /* Reserved field, for alignment */
    u_char cam_sim_priv[ SIM_PRIV ];    /* SIM private data area */
```

```
} CCB_SCSIIO;

/* Set Async Callback CCB */
typedef struct ccb_setasync
{
    CCB_HEADER cam_ch;                  /* Header information fields */
    u_long cam_async_flags;             /* Event enables for Callback resp */
    void (*cam_async_func)();           /* Async Callback function address */
    u_char *pdrv_buf;                   /* Buffer set aside by the Per. drv */
    u_char pdrv_buf_len;                /* The size of the buffer */
} CCB_SETASYNC;

/* Set device type CCB */
typedef struct ccb_setdev
{
    CCB_HEADER cam_ch;                  /* Header information fields */
    u_char cam_dev_type;                /* Val for the dev type field in EDT */
} CCB_SETDEV;

/* SCSI Control Functions. */

/* Abort XPT Request CCB */
typedef struct ccb_abort
{
    CCB_HEADER cam_ch;                  /* Header information fields */
    CCB_HEADER *cam_abort_ch;           /* Pointer to the CCB to abort */
} CCB_ABORT;

/* Reset SCSI Bus CCB */
typedef struct ccb_resetbus
{
    CCB_HEADER cam_ch;                  /* Header information fields */
} CCB_RESETBUS;

/* Reset SCSI Device CCB */
typedef struct ccb_resetdev
{
    CCB_HEADER cam_ch;                  /* Header information fields */
} CCB_RESETDEV;

/* Terminate I/O Process Request CCB */
typedef struct ccb_termio
{
    CCB_HEADER cam_ch;                  /* Header information fields */
    CCB_HEADER *cam_termio_ch;          /* Pointer to the CCB to terminate */
} CCB_TERMIO;

/* Target mode structures. */

typedef struct ccb_en_lun
{
    CCB_HEADER cam_ch;                  /* Header information fields */
    u_short cam_grp6_len;               /* Group 6 VU CDB length */
    u_short cam_grp7_len;               /* Group 7 VU CDB length */
    u_char *cam_ccb_listptr;            /* Pointer to the target CCB list */
    u_short cam_ccb_listcnt;            /* Count of Target CCBs in the list */
} CCB_EN_LUN;

/* HBA engine structures. */

typedef struct ccb_eng_inq
{
    CCB_HEADER cam_ch;                  /* Header information fields */
    u_short cam_eng_num;                /* The number for this inquiry */
    u_char cam_eng_type;                /* Returned engine type */
    u_char cam_eng_algo;                /* Returned algorithm type */
    u_long cam_eng_memory;              /* Returned engine memory size */
} CCB_ENG_INQ;

typedef struct ccb_eng_exec     /* NOTE: must match SCSIIO size */
{
    CCB_HEADER cam_ch;                  /* Header information fields */
    u_char *cam_pdrv_ptr;               /* Ptr used by the Peripheral driver */
    u_long cam_engrsvd0;                /* Reserved field, for alignment */
    u_char *cam_req_map;                /* Ptr for mapping info on the Req. */
    void (*cam_cbfcnp)();               /* Callback on completion function */
    u_char *cam_data_ptr;               /* Pointer to the data buf/SG list */
    u_long cam_dxfer_len;               /* Data xfer length */
    u_char *cam_engdata_ptr;            /* Pointer to the engine buffer data */
    u_char cam_engrsvd1;                /* Reserved field, for alignment */
    u_char cam_engrsvd2;                /* Reserved field, for alignment */
    u_short cam_sglist_cnt;             /* Num of scatter gather list entries */
    u_long cam_dmax_len;                /* Destination data maximum length */
    u_long cam_dest_len;                /* Destination data length */
    long cam_src_resid;                 /* Source residual length: 2's comp */
    u_char cam_engrsvd3[12];            /* Reserved field, for alignment */
    u_long cam_timeout;                 /* Timeout value */
    u_long cam_engrsvd4;                /* Reserved field, for alignment */
    u_short cam_eng_num;                /* Engine number for this request */
    u_short cam_vu_flags;               /* Vendor unique flags */
    u_char cam_engrsvd5;                /* Reserved field, for alignment */
    u_char cam_engrsvd6[3];             /* Reserved field, for alignment */
    u_char cam_sim_priv[ SIM_PRIV ];    /* SIM private data area */
} CCB_ENG_EXEC;

/* The CAM_SIM_ENTRY definition is used to define the entry points for
the SIMs contained in the SCSI CAM subsystem.  Each SIM file will
contain a declaration for it's entry.  The address for this entry will
be stored in the cam_conftbl[] array along will all the other SIM
entries. */

typedef struct cam_sim_entry
{
    long (*sim_init)();             /* Pointer to the SIM init routine */
    long (*sim_action)();           /* Pointer to the SIM CCB go routine */
} CAM_SIM_ENTRY;

/* ------------------------------------------------------------------------ */

/* Defines for the CAM status field in the CCB header. */

#define CAM_REQ_INPROG          0x00    /* CCB request is in progress */
#define CAM_REQ_CMP             0x01    /* CCB request completed w/out error */
#define CAM_REQ_ABORTED         0x02    /* CCB request aborted by the host */
#define CAM_UA_ABORT            0x03    /* Unable to Abort CCB request */
#define CAM_REQ_CMP_ERR         0x04    /* CCB request completed with an err */
#define CAM_BUSY                0x05    /* CAM subsystem is busy */
#define CAM_REQ_INVALID         0x06    /* CCB request is invalid */
#define CAM_PATH_INVALID        0x07    /* Path ID supplied is invalid */
#define CAM_DEV_NOT_THERE       0x08    /* SCSI device not installed/there */
#define CAM_UA_TERMIO           0x09    /* Unable to Terminate I/O CCB req */
#define CAM_SEL_TIMEOUT         0x0A    /* Target selection timeout */
#define CAM_CMD_TIMEOUT         0x0B    /* Command timeout */
#define CAM_MSG_REJECT_REC      0x0D    /* Message reject received */
#define CAM_SCSI_BUS_RESET      0x0E    /* SCSI bus reset sent/received */
#define CAM_UNCOR_PARITY        0x0F    /* Uncorrectable parity err occurred */
#define CAM_AUTOSENSE_FAIL      0x10    /* Autosense: Request sense cmd fail */
#define CAM_NO_HBA              0x11    /* No HBA detected Error */
#define CAM_DATA_RUN_ERR        0x12    /* Data overrun/underrun error */
#define CAM_UNEXP_BUSFREE       0x13    /* Unexpected BUS free */
#define CAM_SEQUENCE_FAIL       0x14    /* Target bus phase sequence failure */
#define CAM_CCB_LEN_ERR         0x15    /* CCB length supplied is inadequate */
```

```
#define CAM_PROVIDE_FAIL        0x16    /* Unable to provide requ. capability */
#define CAM_BDR_SENT            0x17    /* A SCSI BDR msg was sent to target */
#define CAM_REQ_TERMIO          0x18    /* CCB request terminated by the host */

#define CAM_LUN_INVALID         0x38    /* LUN supplied is invalid */
#define CAM_TID_INVALID         0x39    /* Target ID supplied is invalid */
#define CAM_FUNC_NOTAVAIL       0x3A    /* The requ. func is not available */
#define CAM_NO_NEXUS            0x3B    /* Nexus is not established */
#define CAM_IID_INVALID         0x3C    /* The initiator ID is invalid */
#define CAM_CDB_RECVD           0x3E    /* The SCSI CDB has been received */
#define CAM_SCSI_BUSY           0x3F    /* SCSI bus busy */

#define CAM_SIM_QFRZN           0x40    /* The SIM queue is frozen w/this err */
#define CAM_AUTOSNS_VALID       0x80    /* Autosense data valid for target */

#define CAM_STATUS_MASK         0x3F    /* Mask bits for just the status # */

/* ------------------------------------------------------------------------ */

/* Defines for the CAM flags field in the CCB header. */

#define CAM_DIR_RESV        0x00000000  /* Data direction (00: reserved) */
#define CAM_DIR_IN          0x00000040  /* Data direction (01: DATA IN) */
#define CAM_DIR_OUT         0x00000080  /* Data direction (10: DATA OUT) */
#define CAM_DIR_NONE        0x000000C0  /* Data direction (11: no data) */
#define CAM_DIS_AUTOSENSE   0x00000020  /* Disable autosense feature */
#define CAM_SCATTER_VALID   0x00000010  /* Scatter/gather list is valid */
#define CAM_DIS_CALLBACK    0x00000008  /* Disable callback feature */
#define CAM_CDB_LINKED      0x00000004  /* The CCB contains a linked CDB */
#define CAM_QUEUE_ENABLE    0x00000002  /* SIM queue actions are enabled */
#define CAM_CDB_POINTER     0x00000001  /* The CDB field contains a pointer */

#define CAM_DIS_DISCONNECT  0x00008000  /* Disable disconnect */
#define CAM_INITIATE_SYNC   0x00004000  /* Attempt Sync data xfer, and SDTR */
#define CAM_DIS_SYNC        0x00002000  /* Disable sync, go to async */
#define CAM_SIM_QHEAD       0x00001000  /* Place CCB at the head of SIM Q */
#define CAM_SIM_QFREEZE     0x00000800  /* Return the SIM Q to frozen state */
#define CAM_ENG_SYNC        0x00000400  /* Flush resid bytes before cmplt */

#define CAM_ENG_SGLIST      0x00800000  /* The SG list is for the HBA engine */
#define CAM_CDB_PHYS        0x00400000  /* CDB pointer is physical */
#define CAM_DATA_PHYS       0x00200000  /* SG/Buffer data ptrs are physical */
#define CAM_SNS_BUF_PHYS    0x00100000  /* Autosense data ptr is physical */
#define CAM_MSG_BUF_PHYS    0x00080000  /* Message buffer ptr is physical */
#define CAM_NXT_CCB_PHYS    0x00040000  /* Next CCB pointer is physical */
#define CAM_CALLBCK_PHYS    0x00020000  /* Callback func ptr is physical */

#define CAM_DATAB_VALID     0x80000000  /* Data buffer valid */
#define CAM_STATUS_VALID    0x40000000  /* Status buffer valid */
#define CAM_MSGB_VALID      0x20000000  /* Message buffer valid */
#define CAM_TGT_PHASE_MODE  0x08000000  /* The SIM will run in phase mode */
#define CAM_TGT_CCB_AVAIL   0x04000000  /* Target CCB available */
#define CAM_DIS_AUTODISC    0x02000000  /* Disable autodisconnect */
#define CAM_DIS_AUTOSRP     0x01000000  /* Disable autosave/restore ptrs */

/* ------------------------------------------------------------------------ */

/* Defines for the SIM/HBA queue actions.  These value are used in the
SCSI I/O CCB, for the queue action field. [These values should match the
defines from some other include file for the SCSI message phases.  We may
not need these definitions here. ] */

#define CAM_SIMPLE_QTAG         0x20        /* Tag for a simple queue */
#define CAM_HEAD_QTAG           0x21        /* Tag for head of queue */
#define CAM_ORDERED_QTAG        0x22        /* Tag for ordered queue */
```

```
/* ------------------------------------------------------------------------ */

/* Defines for the timeout field in the SCSI I/O CCB.  At this time a value
of 0xF-F indicates a infinite timeout.  A value of 0x0-0 indicates that the
SIM's default timeout can take effect. */

#define CAM_TIME_DEFAULT        0x00000000      /* Use SIM default value */
#define CAM_TIME_INFINITY       0xFFFFFFFF      /* Infinite timeout for I/O */

/* ------------------------------------------------------------------------ */

/* Defines for the Path Inquiry CCB fields. */

#define CAM_VERSION             0x23    /* Binary value for the current ver */

#define PI_MDP_ABLE             0x80    /* Supports MDP message */
#define PI_WIDE_32              0x40    /* Supports 32 bit wide SCSI */
#define PI_WIDE_16              0x20    /* Supports 16 bit wide SCSI */
#define PI_SDTR_ABLE            0x10    /* Supports SDTR message */
#define PI_LINKED_CDB           0x08    /* Supports linked CDBs */
#define PI_TAG_ABLE             0x02    /* Supports tag queue message */
#define PI_SOFT_RST             0x01    /* Supports soft reset */

#define PIT_PROCESSOR           0x80    /* Target mode processor mode */
#define PIT_PHASE               0x40    /* Target mode phase cog. mode */

#define PIH_SCANHILO            0x80    /* Bus scans from ID 7 to ID 0 */
#define PIH_NOREMOVE            0x40    /* Removable dev not included in scan */
#define PIH_NOINQUIRY           0x20    /* Inquiry data not kept by XPT */

/* ------------------------------------------------------------------------ */

/* Defines for Asynchronous Callback CCB fields. */

#define AC_FOUND_DEVICES        0x80    /* During a rescan new device found */
#define AC_SIM_DEREGISTER       0x40    /* A loaded SIM has de-registered */
#define AC_SIM_REGISTER         0x20    /* A loaded SIM has registered */
#define AC_SENT_BDR             0x10    /* A BDR message was sent to target */
#define AC_SCSI_AEN             0x08    /* A SCSI AEN has been received */
#define AC_UNSOL_RESEL          0x02    /* A unsolicited reselection occurred */
#define AC_BUS_RESET            0x01    /* A SCSI bus RESET occurred */

/* ------------------------------------------------------------------------ */

/* Typedef for a scatter/gather list element. */

typedef struct sg_elem
{
    u_char *cam_sg_address;             /* Scatter/Gather address */
    u_long cam_sg_count;                /* Scatter/Gather count */
} SG_ELEM;

/* ------------------------------------------------------------------------ */

/* Defines for the HBA engine inquiry CCB fields. */

#define EIT_BUFFER              0x00    /* Engine type: Buffer memory */
#define EIT_LOSSLESS            0x01    /* Engine type: Lossless compression */
#define EIT_LOSSLY              0x02    /* Engine type: Lossly compression */
#define EIT_ENCRYPT             0x03    /* Engine type: Encryption */

#define EAD_VUNIQUE             0x00    /* Eng algorithm ID: vendor unique */
#define EAD_LZ1V1               0x00    /* Eng algorithm ID: LZ1 var. 1*/
#define EAD_LZ2V1               0x00    /* Eng algorithm ID: LZ2 var. 1*/
#define EAD_LZ2V2               0x00    /* Eng algorithm ID: LZ2 var. 2*/
```

```
/* ---------------------------------------------------------------------- */
/* ---------------------------------------------------------------------- */

/* Unix OSD defines and data structures. */

#define INQLEN  36              /* Inquiry string length to store. */

#define CAM_SUCCESS     0       /* For signaling general success */
#define CAM_FAILURE     1       /* For signaling general failure */

#define CAM_FALSE       0       /* General purpose flag value */
#define CAM_TRUE        1       /* General purpose flag value */

#define XPT_CCB_INVALID -1      /* for signaling a bad CCB to free */

/* General Union for Kernel Space allocation.  Contains all the possible CCB
structures.  This union should never be used for manipulating CCB's its only
use is for the allocation and deallocation of raw CCB space. */

typedef union ccb_size_union
{
    CCB_SCSIIO    csio;     /* Please keep this first, for debug/print */
    CCB_GETDEV    cgd;
    CCB_PATHINQ   cpi;
    CCB_RELSIM    crs;
    CCB_SETASYNC  csa;
    CCB_SETDEV    csd;
    CCB_ABORT     cab;
    CCB_RESETBUS  crb;
    CCB_RESETDEV  crd;
    CCB_TERMIO    ctio;
    CCB_EN_LUN    cel;
    CCB_ENG_INQ   cei;
    CCB_ENG_EXEC  cee;
} CCB_SIZE_UNION;

/* The typedef for the Async callback information.  This structure is used to
store the supplied info from the Set Async Callback CCB, in the EDT table
in a linked list structure. */

typedef struct async_info
{
    struct async_info *cam_async_next;  /* pointer to the next structure */
    u_long cam_event_enable;            /* Event enables for Callback resp */
    void (*cam_async_func)();           /* Async Callback function address */
    u_long cam_async_blen;              /* Length of "information" buffer */
    u_char *cam_async_ptr;              /* Address for the "information */
} ASYNC_INFO;

/* The CAM EDT table contains the device information for all the
devices, SCSI ID and LUN, for all the SCSI busses in the system.  The
table contains a CAM_EDT_ENTRY structure for each device on the bus.
*/

typedef struct cam_edt_entry
{
    long cam_tlun_found;        /* Flag for the existence of the target/LUN */
    ASYNC_INFO *cam_ainfo;      /* Async callback list info for this B/T/L */
    u_long cam_owner_tag;       /* Tag for the peripheral driver's ownership */
    char cam_inq_data[ INQLEN ];/* storage for the inquiry data */
} CAM_EDT_ENTRY;

/* ---------------------------------------------------------------------- */
```

John A. Gallant jag@decvax.dec.com
Senior Software Engineer - Ultrix Engineering Group

Digital Equipment Corp. (603) 881-2472

"A beautiful theory, killed by a nasty, ugly, little fact."
Thomas H. Huxly