To: X3T10 Membership X3T10/95-136r0
Subj: CAM working group meeting minutes 1/10/95
Date: 95-01-25 09:23:36 EST
From: dallas@zk3.dec.com

Minutes for CAM working group 01/10/95
Accredited Standards Committee
X3, Information Processing Systems

Doc. No.: X3T10/95-136r0
Date: January 24, 1995
Project: CAM-3/990D
Ref. Doc.: X3T10/792D-R12

To: Membership of X3T10
From: William Dallas

Subject: Minutes of January 1995 CAM-3 Working Group Meeting

1. Opening Remarks
2. Attendance and Membership
3. Approval of Agenda
4. Discussion of CAM-3 direction and structures.
5. Actions of the working group
6. Meeting Schedule
7. Adjournment

Results of Meeting

1. Opening Remarks

William Dallas the Editor of the CAM-3 document, called the meeting to order at 9:20 am, Tuesday, January 10 1994. He thanked Steve Finch of Silicon Systems for hosting the meeting.

As is customary, the people attending introduced themselves. A copy of the attendance list was circulated for attendance. It was stated that the meeting had been authorized by X3T10 and would be conducted under the X3 rules. The minutes of this meeting will be posted to the SCSI BBS and the SCSI Reflector and will be included in the next committee mailing.

2. Attendance and Membership

Attendance at working group meetings does not count toward minimum attendance requirements for X3T10 membership. Working group meetings are open to any person or company to attend and to express their opinion on the subjects being discussed.

The following people attended the meeting:

Name	Org	Phone Number -or- Electronic Mail Address
Gary Stephens	FSI	6363897@mcimail.com
Lansing Sloan	LLNL	ljsloan@llnl.gov
Paul Boulay	Hitachi Computer Products	408.986.9770 ext. 205
Charles Monia	Digital Equipment Corp.	monia@shr.dec.com
Jeanne Martin	LLNL	jtm@llnl.gov

```
Bill Dallas         Digital Equipment Corp.       dallas@zk3.dec.com
     6  people present
```

3. Approval of Agenda

The agenda developed at the meeting was approved.

4. Discussion of CAM-3 direction and structures.

Bill Dallas presented some ideas that were posted to the SCSI reflector on the direction of CAM-3 and some structure definition. The working group discussed these ideas and felt the method of structure definition and some of the rules associated were worth while. The editor will be starting to merge the definitions, rules and structures into the CAM-3. The SCSI reflector posting is included after these minutes.

The working group discussed expanding the role of the XPT to allow greater transportability of peripheral drivers and the SIMs. The expanded role of the XPT will take the direction of providing standard interfaces to services that operating provide. These general operating services (e.g., kmalloc(sizeof(buffer), flags)) are normally provided in all operating systems but have different call names and parameters. The XPT will be providing a specific call names and parameters to these services.

The working group discussed changing the project name from CAM-2 to CAM-3 to match the work for SCSI-3. The working group felt this was appropriate and the editor presented this to the general SCSI working. John Lohmeyer X310 Chair commented that the project name change is allowed.

It should be noted that all changes and additions to the document is subject to working group approval (e.g., all technical changes made by the editior will have working group approval before presentation to X3T10).

5. Actions of the Working Group

Project name change from CAM-2 to CAM-3. Agreed to general CCB structure and rules. Agreed to expand the role of the XPT.

6. Meeting Schedule

The next meeting of the CAM-3 Working Group is planned for March 7, 1995 in Newport Beach, Ca. The meeting is expected to run from 9:00am-1:00pm.

The topics shall be CAM-3 changes and CAM-3 features/functionality.

Comments from Marc E. Gauthier of the Institute for Information Technology National Research Council Canada.

7. Adjournment

The meeting was adjourned at 12:00pm. on Tuesday, January 10, 1995.

Attachment:

The CAM-2 working will be meeting on January 10, 1995 to discuss the options for CCB structure sizing and some general rules. The CAM-2 working group will then present to the general SCSI working group its thoughts on the structure and sizing of the CCBs and the reasoning for its

selection. It is hoped that the general SCSI working will comment on the selection (e.g., approve, disapprove, enhance the CCB, sizing and general rules or have specific ideas).

The migration of the CAM specification which is based upon the SCSI-2 specification to the CAM-2 specification and will be based on the SCSI-3 presents a number of problems. The largest problem that SCSI-3 presents to CAM is the converting from a dense addressing space to a extremely sparse addressing space. The expansion of the target and LUN from 8 bits to 64 bits will break all the currently written CAM software. The term break used in the previous sentence means that the currently written software will have to be modified and restructured to conform with CAM-2 and SCSI-3.

Some other problems with the CAM specification today is its lack of expansion capability for new features to both the CAM and SCSI specifications and its assumption of a 32 bit processor world. The CAM-2 working group will be working towards a specification that is easily expanded, fully dynamic and processor word size independent.

The following are some ideas that should help in solving some of these issues.

Processor word size independence: To allow transportability (not binary comparability) of CAM-2 peripheral drivers, and SIM/HBA's between different machine platforms and operating ystems, a cam types definition file (cam_types.h) will be provided by the supplier of the xpt. The cam_types.h header file shall define the CCB structures member sizes. While the CCB's size varies based on machine word size, they are of fixed size and structure member offsets are fixed for a specific machine word size.

The supplier of the xpt shall also the supply a CCB definitions file (cam.h) which shall use the defined CCBs as specified by CAM-2. The CCBs specified in CAM-2 shall specify specific member names of the CCB and the size of each member.

Alignment (address boundaries) of the CCB and its members shall also be specified by CAM-2. It shall be the responsibility of the xpt supplier to preserve those alignments as specified by the alignment rules. The alignment rules allows CCB members to at a known offset for a 16, 32, 64, etc bit processors.

I have chosen the C language to represent the types definition file. The cam_types.h file contents shall contain type definitions of the follow:

```
typedef char   I_8;                    /* 8 bits */
typedef unsigned char    U_8;      /* 8 bits */
typedef short  I_16;                        /* 16 bits     */
typedef unsigned short   U_16;          /* 16 bits     */
/* For different machine word sizes ( e.g., 32 or 64 bit )
#ifdef 32_BIT
 typedef  long I_32;                        /* 32 bits          */
typedef unsigned long  U_32;       /* 32 bits          */
#endif /* 32_BIT */
#ifdef 64_BIT
typedef int I_32                           /* 32 bits     */
typedef unsigned int U_32                  /* 32 bits     */

Alignment Rules:
   All (x)_8 (chars) shall being on a 8 bit address boundary and
   shall be 8 bits in length.

   All (x)_16 (shorts) shall being on a on a 16 bit address
   boundary and shall be 16 bits in length.
```

```
All x_32  (32 bits) shall being on a 32 bit address boundary
and shall be 32 bits in length.

All pointers shall begin on a machine word address boundary
shall be of machine word size.

All structures and unions and arrays shall begin and end on a
machine word boundary. If they don't they shall be padded out
to the machine word boundary.

  All CCBs shall begin on a machine word boundary.

Due to compiler differences between machine platforms and
operating systems if the next defined member type does not
align with the specified alignment then it shall be padded to
force alignment.

  Padding shall use the following type:
       U_8          :8;     /* Alignment Padding           */
```

I have looked at a number of different options for the CAM-2 CCBs from a GPP similar structure arrangement to what has been proposed/talked in the distant past CAM-2 Working Group meetings. I believe the following CCB Header definitions are the most optimal for CAM-2. The below CAM-2 header definitions allow for CAM and CAM-2 peripheral drivers and SIM/HBAs to co-exist in a system also allowing for backwards compatibility.

I have rejected the GPP similar structure arrangement due to the following reasons:

To large of a departure from the current definitions.

The use of too many data pointers. The use of pointers to describe data is very flexible but has many drawbacks. The allocation of the storage areas impacts O.S. performance the more you allocate the greater the impact. Has a tendency to be wasteful of a critical resource (system memory) where most operating systems have power of 2 kernel memory allocators so when you ask for a buffer of 36 bytes you get a 64 byte bucket.

```
  CAM (CAM-1) CCB header:
  /* Common CCB CAM header definition.
   * For 32 bit machines
   */
  typedef struct ccb_header {
       void *my_addr;                /* The address of this CCB */
       U_16 cam_ccb_len;             /* Length of the entire CCB */
       U_8 cam_func_code;            /* XPT function code */
       U_8 cam_status;               /* Returned CAM subsystem status
*/
       U_16 cam_hrsvd0;              /* Reserved */
       U_8 cam_path_id;              /* Path ID for the request */
       U_8 cam_target_id;            /* Target device ID */
       U_8 cam_target_lun;           /* Target LUN number */
       U_32 cam_flags;               /* Flags for operation of the
subsys
*/
  }CCB_HEADER;                       /* structure ends on 32 bit
boundary
```

```
*/

     CAM 2 CCB header:
     /* Common CCB CAM 2 header definition.
      * For 32 bit machines
     */
     typedef struct ccb_header2 {
          void *my_addr;         /* The address of this CCB */
          U_16 cam_ccb_len;      /* Length of the entire CCB */
          U_8 cam_func_code;     /* XPT function code/CAM 2 signifier */
          U_8 cam_status;        /* Returned CAM subsystem status */
          U_32 cam_path_mid;     /* Path ID for the request (Most
significant)
*/
          U_32 cam_path_lid;     /* Path ID for the request (Least
significant)
*/
          U_32 cam_target_mid;   /* Target device ID (Most significant)
*/
          U_32 cam_target_lid;   /* Target device ID (Least significant)
*/
          U_32 cam_target_mlun;  /* Target LUN number  (Most significant
*/
          U_32 cam_target_mlun;  /* Target LUN number  (Least
significant) */
          U_32 cam2_func_code;   /* The Real CAM 2 function code. */
          U_32 cam_flags;                    /* Flags for operation of
the
subsys */
          U_8 :8;                            /* Reserved for expansion
*/
          U_8 :8;                            /* Reserved for expansion
*/
          U_8 :8;                            /* Reserved for expansion
*/
          U_8 :8;                            /* Reserved for expansion
*/
          U_32 cam_target_flags              /* Target mode flags */
          U_8 :8;                            /* Reserved for expansion
*/
          U_8 :8;                            /* Reserved for expansion
*/
          U_8 :8;                            /* Reserved for expansion
*/
          U_8 :8;                            /* Reserved for expansion
*/
          U_8 :8;                            /* Reserved for expansion
*/
          U_8 :8;                            /* Reserved for expansion
*/
          U_8 :8;                            /* Reserved for expansion
*/
          U_8 :8;                            /* Reserved for expansion
*/
```

```
        /* The above reservation of 8 bytes is done to allow for
         * future routing of the request over a communication path
         * (e.g., network).  The boundaries of  what a host (system)
         * is has rapidly changed from what it was 2 years ago. While I
         * haven't thought this out completely yet (maybe 16 bytes is
         * needed source and destination addresses) I have placed it
         * here as a marker to provoke some thought into it.
         */

      }CCB_HEADER2;     /* structure ends on 32 bit boundary */
```

Note there is a restriction that there can only be 0x0 to 0xef SIM/HBAs in CAM-2. This restriction will be lifted when CAM-3 (if there is one) is defined. When this occurs it is expected that there all SIM/HBA's must be migrated to CAM-2.

Rules for the CAM 2 XPT:

The XPT shall support both CAM and CAM-2 CCBs.

The XPT shall in xpt_action() determine if this is a CAM or CAM-2 function and route accordingly.

The XPT shall be addressed by a cam_path_id of 0xFF (CAM) and a cam_path_mid of 0xFFFFFFFF with a cam_path_lid of 0xFFFFFFFF (CAM-2). This means the CAM-2 XPT can be addressed by both addresses so peripheral drivers can determine if this is a CAM or CAM-2 XPT. The XPT shall report in the CAM Path Inquiry CCB that it is a CAM-2 XPT.

Rules for the SIM/HBAs:

The SIM/HBAs shall report in the CAM Path Inquiry CCB that it is a CAM or CAM-2 SIM/HBA.

The SIM/HBAs that support CAM-2 shall report in the CAM-2 Path Inquiry CCB that it supports CAM-2 CCBs and it shall report if it supports CAM CCBs (optional).

The SIM/HBA shall determine through a CAM Path Inquiry CCB that it is a CAM or CAM-2 XPT.

Rules for CAM-2 peripheral drivers:

Peripheral drivers shall determine through the CAM Path Inquiry CCB if the XPT is a CAM or CAM-2 XPT.

Peripheral drivers shall determine through the CAM Path Inquiry CCB if the addressed SIM/HBA is a CAM or CAM-2 SIM/HBA.

A peripheral driver shall support CAM or CAM-2 and optionally both.