```
<title>DVB CA Device</title>
<para>The DVB CA device controls the conditional access hardware. It can be
accessed through
<emphasis role="tt">/dev/dvb/adapter0/ca0</emphasis>. Data types and and ioctl
definitions can be accessed by
including <emphasis role="tt">linux/dvb/ca.h</emphasis> in your application.
</para>

<section id="ca_data_types">
<title>CA Data Types</title>


<section id="ca_slot_info_t">
<title>ca_slot_info_t</title>
 <programlisting>
 /&#x22C6; slot interface types and info &#x22C6;/

 typedef struct ca_slot_info_s {
         int num;               /&#x22C6; slot number &#x22C6;/

         int type;           /&#x22C6; CA interface this slot supports &#x22C6;/
 #define CA_CI            1  /&#x22C6; CI high level interface &#x22C6;/
 #define CA_CI_LINK       2  /&#x22C6; CI link layer level interface &#x22C6;/
 #define CA_CI_PHYS       4  /&#x22C6; CI physical layer level interface
&#x22C6;/
 #define CA_SC          128  /&#x22C6; simple smart card interface &#x22C6;/

         unsigned int flags;
 #define CA_CI_MODULE_PRESENT 1 /&#x22C6; module (or card) inserted &#x22C6;/
 #define CA_CI_MODULE_READY   2
 } ca_slot_info_t;
</programlisting>

</section>
<section id="ca_descr_info_t">
<title>ca_descr_info_t</title>
 <programlisting>
 typedef struct ca_descr_info_s {
         unsigned int num;  /&#x22C6; number of available descramblers (keys)
&#x22C6;/
         unsigned int type; /&#x22C6; type of supported scrambling system
&#x22C6;/
 #define CA_ECD           1
 #define CA_NDS           2
 #define CA_DSS           4
 } ca_descr_info_t;
</programlisting>

</section>
<section id="ca_cap_t">
<title>ca_cap_t</title>
 <programlisting>
 typedef struct ca_cap_s {
         unsigned int slot_num;  /&#x22C6; total number of CA card and module
slots &#x22C6;/
         unsigned int slot_type; /&#x22C6; OR of all supported types &#x22C6;/
```

```
        unsigned int descr_num; /&#x22C6; total number of descrambler slots
(keys) &#x22C6;/
        unsigned int descr_type;/&#x22C6; OR of all supported types &#x22C6;/
 } ca_cap_t;
</programlisting>

</section>
<section id="ca_msg_t">
<title>ca_msg_t</title>
 <programlisting>
 /&#x22C6; a message to/from a CI-CAM &#x22C6;/
 typedef struct ca_msg_s {
        unsigned int index;
        unsigned int type;
        unsigned int length;
        unsigned char msg[256];
 } ca_msg_t;
</programlisting>

</section>
<section id="ca_descr_t">
<title>ca_descr_t</title>
 <programlisting>
 typedef struct ca_descr_s {
        unsigned int index;
        unsigned int parity;
        unsigned char cw[8];
 } ca_descr_t;
</programlisting>
 </section></section>
<section id="ca_function_calls">
<title>CA Function Calls</title>


<section id="ca_fopen">
<title>open()</title>
<para>DESCRIPTION
</para>
<informaltable><tgroup cols="1"><tbody><row><entry
 align="char">
<para>This system call opens a named ca device (e.g. /dev/ost/ca) for subsequent
use.</para>
<para>When an open() call has succeeded, the device will be ready for use.
 The significance of blocking or non-blocking mode is described in the
 documentation for functions where there is a difference. It does not affect the
 semantics of the open() call itself. A device opened in blocking mode can later
 be put into non-blocking mode (and vice versa) using the F_SETFL command
 of the fcntl system call. This is a standard system call, documented in the
Linux
 manual page for fcntl. Only one user can open the CA Device in O_RDWR
 mode. All other attempts to open the device in this mode will fail, and an
error
 code will be returned.</para>
</entry>
 </row></tbody></tgroup></informaltable>
<para>SYNOPSIS
```

```
</para>
<informaltable><tgroup cols="1"><tbody><row><entry
 align="char">
<para>int open(const char &#x22C6;deviceName, int flags);</para>
</entry>
 </row></tbody></tgroup></informaltable>
<para>PARAMETERS
</para>
<informaltable><tgroup cols="2"><tbody><row><entry
 align="char">
<para>const char
 *deviceName</para>
</entry><entry
 align="char">
<para>Name of specific video device.</para>
</entry>
 </row><row><entry
 align="char">
<para>int flags</para>
</entry><entry
 align="char">
<para>A bit-wise OR of the following flags:</para>
</entry>
 </row><row><entry
 align="char">
</entry><entry
 align="char">
<para>O_RDONLY read-only access</para>
</entry>
 </row><row><entry
 align="char">
</entry><entry
 align="char">
<para>O_RDWR read/write access</para>
</entry>
 </row><row><entry
 align="char">
</entry><entry
 align="char">
<para>O_NONBLOCK open in non-blocking mode</para>
</entry>
 </row><row><entry
 align="char">
</entry><entry
 align="char">
<para>(blocking mode is the default)</para>
</entry>
 </row></tbody></tgroup></informaltable>
<para>ERRORS
</para>
<informaltable><tgroup cols="2"><tbody><row><entry
 align="char">
<para>ENODEV</para>
</entry><entry
 align="char">
<para>Device driver not loaded/available.</para>
```

```
</entry>
 </row><row><entry
 align="char">
<para>EINTERNAL</para>
</entry><entry
 align="char">
<para>Internal error.</para>
</entry>
 </row><row><entry
 align="char">
<para>EBUSY</para>
</entry><entry
 align="char">
<para>Device or resource busy.</para>
</entry>
 </row><row><entry
 align="char">
<para>EINVAL</para>
</entry><entry
 align="char">
<para>Invalid argument.</para>
</entry>
 </row></tbody></tgroup></informaltable>

</section>
<section id="ca_fclose">
<title>close()</title>
<para>DESCRIPTION
</para>
<informaltable><tgroup cols="1"><tbody><row><entry
 align="char">
<para>This system call closes a previously opened audio device.</para>
</entry>
 </row></tbody></tgroup></informaltable>
<para>SYNOPSIS
</para>
<informaltable><tgroup cols="1"><tbody><row><entry
 align="char">
<para>int close(int fd);</para>
</entry>
 </row></tbody></tgroup></informaltable>
<para>PARAMETERS
</para>
<informaltable><tgroup cols="2"><tbody><row><entry
 align="char">
<para>int fd</para>
</entry><entry
 align="char">
<para>File descriptor returned by a previous call to open().</para>
</entry>
 </row></tbody></tgroup></informaltable>
<para>ERRORS
</para>
<informaltable><tgroup cols="2"><tbody><row><entry
 align="char">
<para>EBADF</para>
```

```
</entry><entry
 align="char">
<para>fd is not a valid open file descriptor.</para>
</entry>
 </row></tbody></tgroup></informaltable>
 </section>
</section>
```