# Installation Guide
# of
# CAMAC Library and Driver package
# (Version 2.0.29c)
# for
# Linux OS and ISA/PCI-CC7x00

# 目 次

# 第1章　How to install

## 1.1　Package Version

本パッケージには東洋テクニカ社製の “ISAcc7000” , “ISAcc7700” ,“PCIcc7700” を Linux OS 上で使用するためのドライバ及びライブラリが含まれています。皆さんが入手したパッケージの名前は “camac.linux.X.X.XX.Y.tgz” の形となっていると思います (ここで X は数字 Y は英語の小文字 1 文字を表す)。“camac.linux” の後に続く 3 つの数値は開発が行われた Linux の OS バージョンで、最後にくっついている英語の文字がこのパッケージのバージョンにあたります。例えば “camac.linux.2.0.29a.tgz” であれば、Linux 2.0.29 で開発された “a” バージョンのパッケージを表します。もちろんのこと、同じ “a” バージョンのパッケージでも “camac.linux” の後に続く 3 つの数値が異なれば全く違う内容になっています。パッケージの数値と皆さんが使っている Linux のバージョンの数値との関係ですが、始めの 2 つの数値が一致していればおそらく動作すると思われます。(できれば 3 つの数値が一致していることが望ましい。)

## 1.2　Directories

まず入手したパッケージを展開します。このドキュメントを読んでいる方はもうパッケージを展開しているとは思いますが、

```
% tar -tvzf camac.linux.X.X.XX.Y.tgz ↵
```

で展開する事ができます。すると展開を行ったディレクトリ以下に、

```
------ camaclinux.X.X.XX.Y --- src
                              |
                              +--- include
                              |
                              +--- lib
                              |
                              +--- doc
                              |
                              +--- samples --- src
                                              |
                                              +--- bin
```

**List 1.2.1: “Directories tree”**

この様なディレクトリが作成されるはずです。(“camac.X.X.XX.Y” のディレクトリを以下トップディレクトリと呼びます。)

## 1.3 Before making

実際にライブラリーとドライバーを作成する手順は、

1. トップディレクトリにある Makefile の編集
2. トップディレクトリで make clean の実行
3. トップディレクトリで make dep の実行
4. トップディレクトリで make Src の実行
5. トップディレクトリで make install の実行

となっています。Makefile の一部分を以下に示しておきます。

```
# File Name     : Makefile
# Introduction  : Makefile for Crate Controller "ISA/PCI-cc7x00" on Linux OS.
#               : Main version.
# Creation Date : 1997/08/19
# Author        : Yasu.S Yamagata Univ. department of physics.
# Modified by   :
# Version       : 2.0.29a

#
# User Selection.
#
# Crate Controller(Select from ISAcc7000 , ISAcc7700 or PCIcc7700)
CRATETYPE = PCIcc7700
#
CC7x00_MAJOR = "30"
# Base Address of Crate Controller
ADDRESS = "0x0310"
# Irq of Crate Controller.
IRQ     = "10"
# Debug Option
DEBUG      =

#
# Do not edit following lines.
#
SHELL    = /bin/sh
CC7x00_TOP = $(shell pwd)
INCDIR     = $(CC7x00_TOP)/include
```

```
LIBDIR      = $(CC7x00_TOP)/lib
```

**List 1.3.2: “Makefile”**

編集すべき部分は、“CRATETYPE” , “CC7x00_MAJOR” , “ADDRESS” , “IRQ” “DEBUG” の部分です。(数値を指定する部分については “ ” で囲むのを忘れないで下さい。)

1. CRATENAME: この行は、使用するクレートコントローラーの選択部分です。
   必ず “ISAcc7000” “ISAcc7700” “PCIcc7700” の 3 つの中から 1 つを選んで下さい。
2. CC7x00_MAJOR: この行では作成するドライバのメジャーナンバを指定します。
   “/usr/src/linux/include/linux/major.h” を見て使われていないものを選択して割り当てて下さい。
3. ADDRESS: この行ではクレートコントローラーの I/O Port のベースアドレスを指定します。

   ```
   % more /proc/ioports
   ```

   を実行し、空いているポートを選択して割り当てて下さい。“PCIcc7700” を選択した場合でもこの値は必要ですので、必ず指定して下さい。
4. IRQ: この行では使用する IRQ を指定します。

   ```
   % more /proc/ioports
   ```

   を実行し、空いている IRQ を指定して下さい。“PCIcc7700” を選択した場合でもこの値は必要ですので、必ず指定して下さい。ただし、“PCIcc7700” を選択した場合、IRQ に “10” を設定するとシステムがハングアップすることが確認されています。
5. DEBUG: この行では何も指定しないで下さい。

複数のクレートコントローラー用のドライバ及びライブラリを作成する人は、ライブラリーとドライバーを作成する手順の (1) から (5) を繰り返して下さい。

## 1.4 How to make

Makefile の編集が終ったら実際にドライバとライブラリを作成します。トップディレクトリに移動し、

```
% make clean
```

```
% make dep
```

```
% make Src
```

と実行します。これでライブラリ及びドライバがトップディレクトリの下にある “lib” ディレクトリに作成されたはずです。では、実際に作成したドライバ及びライブラリのインストールですが、root 権限を持つユーザで行ないます。

```
% su
```

等を用いて root 権限を持つユーザになった後、

```
% make install
```

を実行すると、“/dev” ディレクトリの中にドライバが、“/usr/lib” の中にライブラリ等がインストールされます。

# 第2章　Sample programs

## 2.1　How to make sample programs

本パッケージには、スイッチレジスタとインターラプトレジスタを用いた、ライブラリー及びドライバーのテスト用サンプルプログラムが附属しています。ドライバー及びライブラリーがインストールされた後、これらのサンプルプログラムを作成することが可能になります。作成する手順は、

1. “camac.linux.X.X.XX.Y/samples/src” にある testlibdrv.h の編集
2. トップディレクトリで make Samples の実行

となっています。
以下に示す “testlibdrv.h” の部分で、使用するスイッチレジスタとインターラプトレジスタのステーションナンバを定義して下さい。

```
/* For single16/24 and block 16/24 */
/* station number of switch register */
#define SW_REG 4

/* For waitlam */
/* station number of interrupt register */
#define INTERRUPT_REG 2
```

**List 2.1.1: “testlibdrv.h”**

“testlibdrv.h” の編集後、トップディレクトリで

```
% make Samples ↵
```

と実行すると、 “camac.linux.X.X.XX.Y/samples/bin” にサンプルプログラムが作成されます。

## 2.2 List of sample programs

### 2.2.1 cratetest

クレートコントローラーまわりのファンクション (オープン、クローズ、リセット等) のテストプログラムです。

### 2.2.2 single16

スイッチレジスタに対し 16 bit データの書き込み/読み込みを繰り返します。書き込まれるデータは 0 から始まって 0xffff までインクリメントされて行きます。

### 2.2.3 single24

スイッチレジスタに対し 24 bit データの書き込み/読み込みを繰り返します。書き込まれるデータは 0 から始まって 0xffffff までインクリメントされて行きます。

### 2.2.4 bloack16

16 bit ブロックアクションのテストです。スイッチレジスタを用います。

### 2.2.5 bloack24

24 bit ブロックアクションのテストです。スイッチレジスタを用います。

### 2.2.6 waitlam

割り込み関係のテストです。インターラプトレジスタのサブアドレス 0 番に適当なパルスをクロックジェネレータ等から入れて下さい。

# 第3章　Library reference

## 3.1 COPEN

| | |
|---|---|
| NAME | COPEN |
| Defined Name | CAM_Open CAMOPN |
| Abstract | Open the CAMAC crate controller. |
| Syntax | int COPEN(void) |
| Arguments | Nothing. |
| Return Value | Successful : 0 (CC_K_READY)<br>False : 1 (CC_K_BUSY) defined by “camlib.h” |

## 3.2 CCLOSE

| | |
|---|---|
| NAME | CCLOSE |
| Defined Name | CAM_Close CAMCLS |
| Abstract | Close the CAMAC crate controller. |
| Syntax | int COPEN(void) |
| Arguments | Nothing. |
| Return Value | Successful : 0<br>False : It returns groval value “errno” defined by “errno.h” |

## 3.3 CRESET

| | |
|---|---|
| NAME | CRESET |
| Defined Name | |
| Abstract | Reset the CAMAC crate controller. |
| Syntax | int CRESET(void) |
| Arguments | Nothing. |
| Return Value | 0<br>In the case of using “ISAcc7000”, this function is null function, so this function always returns 0. |

## 3.4 CSETBR

| | |
|---|---|
| NAME | CSETBR |
| Defined Name | |
| Abstract | Setting current branch number. |
| Syntax | int CSETBR(int No) |
| Arguments | No : Branch's No. |
| Return Value | This function is not supported. So always this function returns 0. |

## 3.5 CSETCR

| | |
|---|---|
| NAME | CSETCR |
| Defined Name | |
| Abstract | Setting current crate number. |
| Syntax | int CSETCR(int No) |
| Arguments | No : crate's No. |
| Return Value | Successful : 0<br>False : EINVAL defined by "errno.h" |

## 3.6 CGENZ

| | |
|---|---|
| NAME | CGENZ |
| Defined Name | |
| Abstract | Initializing the current CAMAC crate. |
| Syntax | int CGENZ(void) |
| Arguments | Nothing. |
| Return Value | 0 |

## 3.7 CGENC

| | |
|---|---|
| NAME | CGENC |
| Defined Name | |
| Abstract | Clearing the dataway of current CAMAC crate. |
| Syntax | int CGENC(void) |
| Arguments | Nothing. |
| Return Value | 0 |

## 3.8 CSETI

| | |
|---|---|
| NAME | CSETI |
| Defined Name | |
| Abstract | Setting the dataway inhibit of current CAMAC crate. |
| Syntax | int CSETI(void) |
| Arguments | Nothing. |
| Return Value | 0 |

## 3.9 CREMI

| | |
|---|---|
| NAME | CREMI |
| Defined Name | |
| Abstract | Resetting the dataway inhibit of current CAMAC crate. |
| Syntax | int CSETI(void) |
| Arguments | Nothing. |
| Return Value | 0 |

## 3.10 CAMAC

| | |
|---|---|
| NAME | CAMAC |
| Defined Name | CAMACL |
| Abstract | Executing single CAMAC operation with 24-bit CAMAC data. |
| Syntax | int CAMAC(int naf, void* data, int* q, int* x) |
| Arguments | naf : NAF data created by NAF() function.<br>data : Pointer of integer data.<br>q : Pointer of integer data. (If it is set 0, there is no Q response.)<br>x : Pointer of integer data. (If it is set 0, there is no X response.) |
| Return Value | 0 |

## 3.11 CAMACW

| | |
|---|---|
| NAME | CAMACW |
| Defined Name | |
| Abstract | Executing single CAMAC operation with 16-bit CAMAC data. |
| Syntax | int CAMACW(int naf, void* data, int* q, int* x) |
| Arguments | naf : NAF data created by NAF() function.<br>data : Pointer of short integer data.<br>q : Pointer of integer data. (If it is set 0, there is no Q response.)<br>x : Pointer of integer data. (If it is set 0, there is no X response.) |
| Return Value | 0 |

## 3.12 CDMAL

| | |
|---|---|
| NAME | CDMAL |
| Defined Name | CamBloLonWAIt CBLWAI |
| Abstract | Executing block CAMAC operation with 24-bit CAMAC data. |
| Syntax | int CDMAL(int mode, int naf, void* buffer,<br>int length, int* ret_len,int err) |
| Arguments | mode : Mode of block action.<br>CC_K_MODES_QSTOP : Q stop mode.<br>CC_K_MODES_IGNRQ : ignore Q mode.<br>CC_K_MODES_REPEAT : Q repeat mode.<br>CC_K_MODES_ASCAN : Address scan mode.<br>buffer : pointer of integer data array.<br>length : Maxmum length of data buffer by counting a integer.<br>ret_len : Length of writting or reading data by counting integer.<br>err : Pointer os integer data. (Q and X response.)<br>If it is set 0 : Q and X If it is set 1 : no Q and X<br>If it is set 2 : Q and no X If it is set 3 : no Q and no X |
| Return Value | 0 |

## 3.13 CDMAW

| | |
|---|---|
| NAME | CDMAW |
| Defined Name | CamBloWorWAIt CBWWAI |
| Abstract | Executing block CAMAC operation with 16-bit CAMAC data. |
| Syntax | int CDMAW(int mode, int naf, void* buffer,<br>int length, int* ret_len,int err) |
| Arguments | mode : Mode of block action.<br>CC_K_MODES_QSTOP : Q stop mode.<br>CC_K_MODES_IGNRQ : ignore Q mode.<br>CC_K_MODES_REPEAT : Q repeat mode.<br>CC_K_MODES_ASCAN : Address scan mode.<br>naf : NAF data created by NAF() function.<br>buffer : Pointer of short integer data array.<br>length : Maxmum length of data buffer by counting a short integer.<br>ret_len : Length of writting or reading data by counting short integer.<br>err : Pointer os integer data. Q and X response.<br>If it is set 0 : Q and X     If it is set 1 : no Q and X<br>If it is set 2 : Q and no X     If it is set 3 : no Q and no X |
| Return Value | 0 |

## 3.14 CENLAM

| | |
|---|---|
| NAME | CENLAM |
| Defined Name | CAM_EnableLAM CELAM |
| Abstract | Enabling interrupt of current crate controller. |
| Syntax | int CENLAM(int LamPattern) |
| Arguments | LamPattern : Integer data. (This is 24-bit data)<br>If bit is 1, enabling LAM.<br>If bit is 0, disbling LAM.<br>ex) At LamPattern is 0x00000013.<br>Enabling interrupt by modules station No 1, 2, and 5. |
| Return Value | 0 |

## 3.15 CDSLAM

| NAME | CENLAM |
|---|---|
| Defined Name | CAM_DisbleLAM CDLAM |
| Abstract | Disabling interrupt of current crate controller. |
| Syntax | int CDSLAM(void) |
| Arguments | Nothing. |
| Return Value | 0 |

## 3.16 CWTLAM

| NAME | CWTLAM |
|---|---|
| Defined Name | CAM_WaitLAM CWLAM |
| Abstract | Waitnig for generating LAM on current crate. |
| Syntax | int CWTLAM(int time) |
| Arguments | time : Integer data. Waitnig time.(m sec) |
| Return Value | 0 : Successful<br>ETIMEOUT : Time out occured.<br>EINVAL : Unexpected LAM occured. defined by “errno.h”<br>In the case of using “PCIcc7700”, this function is not suppoted now. |

## 3.17 NAF

| NAME | |
|---|---|
| Defined Name | NAF |
| Abstract | Generating CAMAC address data by using “n”, “a” and “f”. |
| Syntax | int NAF(int n, int a, int f) |
| Arguments | n : Station number.<br>a : Sub address.<br>f : CAMAC function. |
| Return Value | Compressed address data for this library functions. |

# 第4章　I/O prot map

## 4.1　ISAcc7000 Register map

| Offset | -00h- | DLR |
|---|---|---|
| BIT | Write | Read |
| 15 | W16 : Data Bit 16 | R16 : Data Bit 16 |
| 14 | W15 : Data Bit 15 | R15 : Data Bit 15 |
| 13 | W14 : Data Bit 14 | R14 : Data Bit 14 |
| 12 | W13 : Data Bit 13 | R13 : Data Bit 13 |
| 11 | W12 : Data Bit 12 | R12 : Data Bit 12 |
| 10 | W11 : Data Bit 11 | R11 : Data Bit 11 |
| 09 | W10 : Data Bit 10 | R10 : Data Bit 10 |
| 08 | W09 : Data Bit 09 | R09 : Data Bit 09 |
| 07 | W08 : Data Bit 08 | R08 : Data Bit 08 |
| 06 | W07 : Data Bit 07 | R07 : Data Bit 07 |
| 05 | W06 : Data Bit 06 | R06 : Data Bit 06 |
| 04 | W05 : Data Bit 05 | R05 : Data Bit 05 |
| 03 | W04 : Data Bit 04 | R04 : Data Bit 04 |
| 02 | W03 : Data Bit 03 | R03 : Data Bit 03 |
| 01 | W02 : Data Bit 02 | R02 : Data Bit 02 |
| 00 | W01 : Data Bit 01 | R01 : Data Bit 01 |

| Offset | -01h- | CCR |
|---|---|---|
| 07 | — | — |
| 06 | — | — |
| 05 | — | — |
| 04 | — | — |
| 03 | — | — |
| 02 | — | — |
| 01 | CA2 : Crate Address “2” | — |
| 00 | CA1 : Crate Address “1” | — |

| Offset | -02h- | DHR |
|---|---|---|
| BIT | Write | Read |
| 15 | — | — |
| 14 | — | — |
| 13 | — | — |
| 12 | — | — |
| 11 | — | — |
| 10 | — | — |
| 09 | — | — |
| 08 | — | — |
| 07 | W24 : Data Bit 24 | R24 : Data Bit 24 |
| 06 | W23 : Data Bit 23 | R23 : Data Bit 23 |
| 05 | W22 : Data Bit 22 | R22 : Data Bit 22 |
| 04 | W21 : Data Bit 21 | R21 : Data Bit 21 |
| 03 | W20 : Data Bit 20 | R20 : Data Bit 20 |
| 02 | W19 : Data Bit 19 | R19 : Data Bit 19 |
| 01 | W18 : Data Bit 18 | R18 : Data Bit 18 |
| 01 | W17 : Data Bit 17 | R17 : Data Bit 17 |

| Offset | -04h- | NAFLR/LAMLR |
|---|---|---|
| Bit | Write | Read |
| 15 | — | L16 : LAM Station 16 |
| 14 | — | L15 : LAM Station 15 |
| 13 | — | L14 : LAM Station 14 |
| 12 | — | L13 : LAM Station 13 |
| 11 | A8 : SubAddress "8" | L12 : LAM Station 12 |
| 10 | A4 : SubAddress "4" | L11 : LAM Station 11 |
| 09 | A2 : SubAddress "2" | L10 : LAM Station 10 |
| 08 | A1 : SubAddress "1" | L09 : LAM Station 09 |
| 07 | — | L08 : LAM Station 08 |
| 06 | — | L07 : LAM Station 07 |
| 05 | — | L06 : LAM Station 06 |
| 04 | N16 : Station Number "16" | L05 : LAM Station 05 |
| 03 | N08 : Station Number "08" | L04 : LAM Station 04 |
| 02 | N04 : Station Number "04" | L03 : LAM Station 03 |
| 01 | N02 : Station Number "02" | L02 : LAM Station 02 |
| 00 | N01 : Station Number "01" | L01 : LAM Station 01 |

| Offset | -06h- | CSR/LAMHR/NAFHR |
|---|---|---|
| BIT | Write | Read |
| 15 | LI : LAM Internal | LI : LAM Internal |
| 14 | — | LS : LAM Sum |
| 13 | — | ONL : On Line |
| 12 | — | DNE : Done |
| 11 | EI : Enable Interrupt | EI : Enable Interrupt |
| 10 | I : Inhibit | I : Inhibit |
| 09 | Z : Initialize | NX : NO-X |
| 08 | C : Clear | NQ : NO-Q |
| 07 | — | L24 : LAM Station 24 |
| 06 | — | L23 : LAM Station 23 |
| 05 | — | L22 : LAM Station 22 |
| 04 | F16 : Function code “16” | L21 : LAM Station 21 |
| 03 | F08 : Function code “08” | L20 : LAM Station 20 |
| 02 | F04 : Function code “04” | L19 : LAM Station 19 |
| 01 | F02 : Function code “02” | L18 : LAM Station 18 |
| 00 | F01 : Function code “01” | L17 : LAM Station 17 |

| Offset | -07h- | EXER |
|---|---|---|
| 07 | — | — |
| 06 | — | — |
| 05 | — | — |
| 04 | — | — |
| 03 | — | — |
| 02 | — | — |
| 01 | — | — |
| 00 | — | — |

○ ISAcc7000 クレートコントローラー取扱説明書より抜粋

## 4.2 ISAcc7700 Register map

| Offset | -00h- | CSR |
|---|---|---|
| Bit | Write | Read |
| 15 | — | — |
| 14 | — | — |
| 13 | — | — |
| 12 | — | — |
| 11 | — | — |
| 10 | — | — |
| 09 | — | — |
| 08 | — | — |
| 07 | LI : LAM Internal | LI : LAM Internal |
| 06 | RS : Reset | LS : LAM Sum |
| 05 | — | ONL : On Line |
| 04 | — | DNE : Done |
| 03 | EI : Enable Interrupt | EI : Enable Interrupt |
| 02 | I : InhiBit | I : InhiBit |
| 01 | Z : Initialize | NX : NO-X |
| 00 | C : Clear | NQ : NO-Q |

| Offset | -02h- | CCR/LAMLR |
|---|---|---|
| Bit | Write | Read |
| 15 | — | L16 : LAM Station 16 |
| 14 | — | L15 : LAM Station 15 |
| 13 | — | L14 : LAM Station 14 |
| 12 | — | L13 : LAM Station 13 |
| 11 | — | L12 : LAM Station 12 |
| 10 | — | L11 : LAM Station 11 |
| 09 | — | L10 : LAM Station 10 |
| 08 | — | L09 : LAM Station 09 |
| 07 | — | L08 : LAM Station 08 |
| 06 | — | L07 : LAM Station 07 |
| 05 | — | L06 : LAM Station 06 |
| 04 | — | L05 : LAM Station 05 |
| 03 | — | L04 : LAM Station 04 |
| 02 | CA04 : Crate Address “04” | L03 : LAM Station 03 |
| 01 | CA02 : Crate Address “02” | L02 : LAM Station 02 |
| 00 | CA01 : Crate Address “01” | L01 : LAM Station 01 |

| Offset | -04h- | NAFR/LAMHR |
|---|---|---|
| Bit | Write | Read |
| 15 | — | — |
| 14 | — | — |
| 13 | N16 : Station Number “16” | — |
| 12 | N08 : Station Number “08” | — |
| 11 | N04 : Station Number “04” | — |
| 10 | N02 : Station Number “02” | — |
| 09 | N01 : Station Number “01” | — |
| 08 | A08 : SubAddress “08” | — |
| 07 | A04 : SubAddress “04” | L24 : LAM Station 24 |
| 06 | A02 : SubAddress “02” | L23 : LAM Station 23 |
| 05 | A01 : SubAddress “01” | L22 : LAM Station 22 |
| 04 | F16 : Function code “16” | L21 : LAM Station 21 |
| 03 | F08 : Function code “08” | L20 : LAM Station 20 |
| 02 | F04 : Function code “04” | L19 : LAM Station 19 |
| 01 | F02 : Function code “02” | L18 : LAM Station 18 |
| 00 | F01 : Function code “01” | L17 : LAM Station 17 |

| Offset | -06h- | DLR |
|---|---|---|
| Bit | Write | Read |
| 15 | W16 : Data Bit 16 | R16 : Data Bit 16 |
| 14 | W15 : Data Bit 15 | R15 : Data Bit 15 |
| 13 | W14 : Data Bit 14 | R14 : Data Bit 14 |
| 12 | W13 : Data Bit 13 | R13 : Data Bit 13 |
| 11 | W12 : Data Bit 12 | R12 : Data Bit 12 |
| 10 | W11 : Data Bit 11 | R11 : Data Bit 11 |
| 09 | W10 : Data Bit 10 | R10 : Data Bit 10 |
| 08 | W09 : Data Bit 09 | R09 : Data Bit 09 |
| 07 | W08 : Data Bit 08 | R08 : Data Bit 08 |
| 06 | W07 : Data Bit 07 | R07 : Data Bit 07 |
| 05 | W06 : Data Bit 06 | R06 : Data Bit 06 |
| 04 | W05 : Data Bit 05 | R05 : Data Bit 05 |
| 03 | W04 : Data Bit 04 | R04 : Data Bit 04 |
| 02 | W03 : Data Bit 03 | R03 : Data Bit 03 |
| 01 | W02 : Data Bit 02 | R02 : Data Bit 02 |
| 00 | W01 : Data Bit 01 | R01 : Data Bit 01 |

| Offset | -08h- | DHR |
|---|---|---|
| Bit | Write | Read |
| 15 | — | — |
| 14 | — | — |
| 13 | — | — |
| 12 | — | — |
| 11 | — | — |
| 10 | — | — |
| 09 | — | — |
| 08 | — | — |
| 07 | W24 : Data Bit 24 | R24 : Data Bit 24 |
| 06 | W23 : Data Bit 23 | R23 : Data Bit 23 |
| 05 | W22 : Data Bit 22 | R22 : Data Bit 22 |
| 04 | W21 : Data Bit 21 | R21 : Data Bit 21 |
| 03 | W20 : Data Bit 20 | R20 : Data Bit 20 |
| 02 | W19 : Data Bit 19 | R19 : Data Bit 19 |
| 01 | W18 : Data Bit 18 | R18 : Data Bit 18 |
| 00 | W17 : Data Bit 17 | R17 : Data Bit 17 |

| Offset | -0Ah- | EXER |
|---|---|---|
| Bit | Write | Read |
| 15 | — | — |
| 14 | — | — |
| 13 | — | — |
| 12 | — | — |
| 11 | — | — |
| 10 | — | — |
| 09 | — | — |
| 08 | — | — |
| 07 | — | — |
| 06 | — | — |
| 05 | — | — |
| 04 | — | — |
| 03 | — | — |
| 02 | — | — |
| 01 | — | — |
| 00 | — | — |

○ ISAcc7700 クレートコントローラー取扱説明書より抜粋

## 4.3 PCIcc7700 Register map

| Offset | -00h- | CSR |
|---|---|---|
| BIT | Write | Read |
| 31 | — | — |
| 30 | — | — |
| 29 | — | — |
| 28 | — | — |
| 27 | — | — |
| 26 | — | — |
| 25 | — | — |
| 24 | — | — |
| 23 | — | — |
| 22 | — | — |
| 21 | — | — |
| 20 | — | — |
| 19 | — | — |
| 18 | — | — |
| 17 | — | — |
| 16 | — | — |
| 15 | — | — |
| 14 | — | — |
| 13 | — | — |
| 12 | — | — |
| 11 | — | — |
| 10 | — | — |
| 09 | — | — |
| 08 | — | — |
| 07 | LI : LAM Internal | LI LAM Internal |
| 06 | RS : Reset | LS : LAM Sum |
| 05 | — | ONL : On Line |
| 04 | — | DNE : Done |
| 03 | EI : Enable Interrupt | EI : Enable Interrupt |
| 02 | I : Inhibit | I : Inhibit |
| 01 | Z : Initialize | NX : NO-X |
| 00 | X : Clear | NQ : NO-Q |

| Offset | -04h- | CCR/LAMR/NAFR |
|---|---|---|
| BIT | Write | Read |
| 31 | — | — |
| 30 | — | — |
| 29 | — | — |
| 28 | — | — |
| 27 | — | — |
| 26 | — | — |
| 25 | — | — |
| 24 | — | — |
| 23 | — | LAM Station 24 |
| 22 | — | LAM Station 23 |
| 21 | — | LAM Station 22 |
| 20 | — | LAM Station 21 |
| 19 | — | LAM Station 20 |
| 18 | CA4 : Crate Address "04" | LAM Station 19 |
| 17 | CA2 : Crate Address "02" | LAM Station 18 |
| 16 | CA1 : Crate Address "01" | LAM Station 17 |
| 15 | — | LAM Station 16 |
| 14 | — | LAM Station 15 |
| 13 | N16 : Station Number "16" | LAM Station 14 |
| 12 | N08 : Station Number "08" | LAM Station 13 |
| 11 | N04 : Station Number "04" | LAM Station 12 |
| 10 | N02 : Station Number "02" | LAM Station 11 |
| 09 | N01 : Station Number "01" | LAM Station 10 |
| 08 | A08 : Sub Address "08" | LAM Station 09 |
| 07 | A04 : Sub Address "04" | LAM Station 08 |
| 06 | A02 : Sub Address "02" | LAM Station 07 |
| 05 | A01 : Sub Address "01" | LAM Station 06 |
| 04 | F16 : Function code "16" | LAM Station 05 |
| 03 | F08 : Function code "08" | LAM Station 04 |
| 02 | F04 : Function code "04" | LAM Station 03 |
| 01 | F02 : Function code "02" | LAM Station 02 |
| 00 | F01 : Function code "01" | LAM Station 01 |

| Offset | -08h- | DR |
|---|---|---|
| BIT | Write | Read |
| 31 | — | — |
| 30 | — | — |
| 29 | — | — |
| 28 | — | — |
| 27 | — | — |
| 26 | — | — |
| 25 | — | — |
| 24 | — | — |
| 23 | W24 : Data Bit 24 | R24 : Data Bit 24 |
| 22 | W23 : Data Bit 23 | R23 : Data Bit 23 |
| 21 | W22 : Data Bit 22 | R22 : Data Bit 22 |
| 20 | W21 : Data Bit 21 | R21 : Data Bit 21 |
| 19 | W20 : Data Bit 20 | R20 : Data Bit 20 |
| 18 | W19 : Data Bit 19 | R19 : Data Bit 19 |
| 17 | W18 : Data Bit 18 | R18 : Data Bit 18 |
| 16 | W17 : Data Bit 17 | R17 : Data Bit 17 |
| 15 | W16 : Data Bit 16 | R16 : Data Bit 16 |
| 14 | W15 : Data Bit 15 | R15 : Data Bit 15 |
| 13 | W14 : Data Bit 14 | R14 : Data Bit 14 |
| 12 | W13 : Data Bit 13 | R13 : Data Bit 13 |
| 11 | W12 : Data Bit 12 | R12 : Data Bit 12 |
| 10 | W11 : Data Bit 11 | R11 : Data Bit 11 |
| 09 | W10 : Data Bit 10 | R10 : Data Bit 10 |
| 08 | W09 : Data Bit 09 | R09 : Data Bit 09 |
| 07 | W08 : Data Bit 08 | R08 : Data Bit 08 |
| 06 | W07 : Data Bit 07 | R07 : Data Bit 07 |
| 05 | W06 : Data Bit 06 | R06 : Data Bit 06 |
| 04 | W05 : Data Bit 05 | R05 : Data Bit 05 |
| 03 | W04 : Data Bit 04 | R04 : Data Bit 04 |
| 02 | W03 : Data Bit 03 | R03 : Data Bit 03 |
| 01 | W02 : Data Bit 02 | R02 : Data Bit 02 |
| 00 | W01 : Data Bit 01 | R01 : Data Bit 01 |

| Offset | -0Ch- | EXER |
|---|---|---|
| BIT | Write | Read |
| 31 | — | — |
| 30 | — | — |
| 29 | — | — |
| 28 | — | — |
| 27 | — | — |
| 26 | — | — |
| 25 | — | — |
| 24 | — | — |
| 23 | — | — |
| 22 | — | — |
| 21 | — | — |
| 20 | — | — |
| 19 | — | — |
| 18 | — | — |
| 17 | — | — |
| 16 | — | — |
| 15 | — | — |
| 14 | — | — |
| 13 | — | — |
| 12 | — | — |
| 11 | — | — |
| 10 | — | — |
| 09 | — | — |
| 08 | — | — |
| 07 | — | — |
| 06 | — | — |
| 05 | — | — |
| 04 | — | — |
| 03 | — | — |
| 02 | — | — |
| 01 | — | — |
| 00 | — | — |

○ PCIcc7700 クレートコントローラー取扱説明書より抜粋

# 第5章　Memo and Bugs

## 5.1　Memo

### 5.1.1　Commands

1. /bin/mknod : スペシャルファイルを作成する。
2. /sbin/insmod : ローダブルモジュールのロード。

   ex) ****.o があるディレクトリで、

   ```
   % insmod ****  ↵
   ```

   と実行するとモジュール “****.o” のロードを行なう。
3. /sbin/rmmod : ローダブルモジュールのアンロード。

   ex) insmod でロードしたモジュール “****.o” のアンロード。

   ```
   % rmmod ****  ↵
   ```
4. make Src : ドライバとライブラリの作成。
5. make Samples : サンプルの作成。
6. make dep : make 用環境ファイルの作成。
7. make clean : ディレクトリの清掃
8. make force_clean : ディレクトリの強制清掃
9. make install : ライブラリ、ドライバのインストール
10. make uninstall : ライブラリ、ドライバのアンインストール
11. make force_uninstall : ライブラリ、ドライバの強制アンインストール
12. make load : ドライバのロード
13. make unload : ドライバのアンロード

### 5.1.2　Files

1. /proc/devices : 現在ロードされているデバイスの一覧。
2. /proc/interrupts : IRQ の一覧。
3. /proc/ioports : I/O Port の一覧。
4. /proc/pci : PCI デバイスの一覧。

5. /proc/version : Linux のバージョン。
6. /var/log/syslog : Linux の kernel が吐き出すメッセージ。root 権限が無いと見ることが出来ない。

## 5.2 Known Bugs

### 5.2.1 Common

1. とにかくまだあちこちにいっぱいバグが有るはず。

### 5.2.2 ISAcc7000

1. LAM が立っている間、ドライバーの割り込みルーチンが何回も呼ばれ続ける。解決策として、割り込みルーチンが呼ばれた時点でクレートコントローラーを割り込み不可状態に変更している。

### 5.2.3 ISAcc7000

1. 今はまだ見つかっていないがどこかにあるはず。

### 5.2.4 PCIcc7700

1. ドライバーをロードするときに IRQ 10 を選択してコンパイルされていると、システムがハングアップする。開発環境特有の問題かどうかは不明。
2. ドライバーの割り込み関数がまだ完成していない。