

# PCI-2702C Board Support Package Installation on RedHawk

## Release Notes Revision B

September 9, 2022



# 1. はじめに

本書は、Concurrent Real Time Inc(CCRT)の RedHawk 上で動作する、インターフェース社製 PCI- 2702C PCI ボードサポートパッケージ 用リリースノートです。

## 2. インストールのための条件

PCI- 2702C BSP をインストールするためには、以下の製品がインストールされている必要があります。

- PCI- 2702C ボード
- RedHawk 6.x 以上
- Extmem version 8.3 以上

PCI-2702Cは、PCIバスに準拠した、TTLシュミット入力(DC+5V)64点デジタル信号 入力と、TTLオープンコレクタ64点のデジタル信号出力を持つデジタル入出力製品です。

## 3. インストール方法

**PCI-2702C BSP** は、IRQ 共有するように設計されています。もしこのデバイスの IRQ が、別のデバイスによって共有されている場合に、このドライバの性能は損なわれる場合があります。そのため、可能な限り、このボードはその IRQ が他の装置と共有されていないPCIスロットの中に実装する事が奨励されます。“lspci -v”コマンドをシステムで種々の装置の IRQ を確認するために使用することができます。

**PCI-2702C BSP** は、CDROM/DVD 上の RPM/DEB フォーマットで供給され、別途 extmem デバイスドライバがインストールされている必要があります。

以下に、インストールの手順を示します。:

x86\_64 アーキテクチャの場合

=== root ユーザで実行してください===

```
# mount /dev/cdrom /mnt あるいは mount /dev/dvd /mnt
```

```
# cd /mnt
```

もし、extmemを同時にインストールする場合には、以下のコマンドを入力してください

```
# rpm -ivh bin-extmem-X.Y_RHx.y-z.x86_64.rpm
```

**PCI2702C BSP 実行パッケージのインストール**

```
# rpm -ivh bin-pci2702c -X.Y_RHx.y-z.x86_64.rpm
```

もし必要であれば、続けて開発パッケージのインストールを行ってください

```
# rpm -ivh dev-pci2702c -X.Y_RHx.y-z.x86_64.rpm
```

```
# umount /mnt
```

amd64 アーキテクチャの場合

=== root ユーザで実行してください===

```
# mount /dev/cdrom /mnt あるいは mount /dev/dvd /mnt
```

```
# cd /mnt
```

もし、extmemを同時にインストールする場合には、以下のコマンドを入力してください

```
# apt install ./bin-extmem-rhx.y_X.Y_amd64.deb
```

**PCI2702C BSP 実行パッケージのインストール**

```
# apt install ./bin-pci2702c -rhx.y_X.Y_amd64.deb
```

もし必要であれば、続けて開発パッケージのインストールを行ってください

```
# apt install ./dev-pci2702c -rhx.y_X.Y_amd64.deb
```

```
# umount /mnt
```

arm64 アーキテクチャの場合

=== root ユーザで実行してください===

```
# mount /dev/cdrom /mnt あるいは mount /dev/dvd /mnt
```

```
# cd /mnt
```

もし、extmemを同時にインストールする場合には、以下のコマンドを入力してください

```
# apt install ./bin-extmem-rhx.y_X.Y_arm64.deb
```

#### **PCI2702C BSP 実行パッケージのインストール**

```
# apt install ./bin-pci2702c-rhx.y_X.Y_arm64.deb
```

もし必要であれば、続けて開発パッケージのインストールを行ってください

```
# apt install ./dev-pci2702c-rhx.y_X.Y_arm64.deb
```

```
# umount /mnt
```

(**x.y** は RedHawk のバージョン番号であり、6.x,7.x または 8.x で、**X.Y** は、BSP のバージョン、**z** は、BSP のリリース番号を示し、予告なく変更することがあります。)

**PCI-2702C BSP** パッケージは **/usr/local/CNC/drivers/extmem/interface/pci2702c** ディレクトリにインストールされ、必要な場所に展開されます。

## **4. アンインストール方法**

**PCI-2702C BSP** パッケージは、以下のコマンドでアンインストールします。この作業により **/usr/local/CNC/drivers/extmem/interface/pci2702c** ディレクトリは削除されます。

x86\_64 アーキテクチャの場合

```
=== root ユーザで実行してください===
```

開発パッケージをインストールしていた場合には、

```
# rpm -e dev-pci2702c -X.Y_RHx.y-z.x86_64 (開発パッケージの削除)
```

```
# rpm -e bin-pci2702c -X.Y_RHx.y-z.x86_64 (実行パッケージの削除)
```

実行パッケージのみをインストールしていた場合には、

```
# rpm -e bin-pci2702c -X.Y_RHx.y-z.x86_64 (実行パッケージの削除)
```

amd64 アーキテクチャの場合

```
=== root ユーザで実行してください===
```

開発パッケージをインストールしていた場合には、

```
# apt purge dev-pci2702c-rhx.y (開発パッケージの削除)
```

```
# apt purge bin-pci2702c-rhx.y (実行パッケージの削除)
```

実行パッケージのみをインストールしていた場合には、

```
# apt purge bin-pci2702c-rhx.y (実行パッケージの削除)
```

arm64 アーキテクチャの場合

```
=== root ユーザで実行してください===
```

開発パッケージをインストールしていた場合には、

```
# apt purge dev-pci2702c-rhx.y (開発パッケージの削除)
```

```
# apt purge bin-pci2702c-rhx.y (実行パッケージの削除)
```

実行パッケージのみをインストールしていた場合には、

```
# apt purge bin-pci2702c-rhx.y (実行パッケージの削除)
```

## 5. ライブラリマニュアル

ライブラリマニュアルは、オンラインで提供されます。

# man pci2702c

pci2702c(3)

pci2702c(3)

### NAME

pci2702c - external memory device access library

### SYNOPSIS

[ ボードの詳細は、各マニュアルを見てください ]

### DESCRIPTION

pci2702c は、external memory ドライバを利用した pci2702c ボードアクセスライブラリです。

```
#include <sys/pci2702c.h>
```

```
gcc [options ...] file -lpci2702c -lxtmem ...
```

\*\*\*\*\*

### PCI2702C

\*\*\*\*\*

#### 割り込みハンドラの登録

```
int pci2702c_setup_signal
```

```
(  
    int fd,  
    void (*interrupt_hadler)( int, siginfo_t *, void *),  
    int mask  
);
```

戻り値

エラーなら-1 成功なら 0

引数

fd ファイルディスクリプタ番号

void (\*interrupt\_hadler)( int, siginfo\_t \*, void \*) 割り込みハンドラ

mask 割り込みを許可するマスク値

#### デバイスの非初期化处理

```
int pci2702c_reset(int fd);
```

```
int pci2702c_uninit(int fd);
```

戻り値  
エラーなら-1 成功なら 0

引数

fd ファイルディスクリプタ番号

2つの関数は同じ処理、全ての制御レジスタに 0 値を設定する。

#### デバイスの初期化处理

```
int pci2702c_init
```

```
(  
    int fd,  
    int option  
);
```

```

    戻り値
        エラーなら-1 成功なら 0

    引数
        fd ファイルディスクリプタ番号
        option 1を指定すると以下の情報が表示される
        BAR0 I/O Region addr 0x00004480 offset 0x00000000 16 bytes

Switch 1

pci2702c の出力を発生させる
int pci2702c_raise_signal
(
    int fd,
    int ack,
    int out1,
    int out2
);
    戻り値
        エラーなら-1 成功なら 0

    引数
        fd ファイルディスクリプタ番号
        ack,out1,out2 割り込みの種類 以下のいずれかを指定する
        ack1
            PCI2702C_PULS_ACK1_NOACTION なにもしない
            PCI2702C_PULS_ACK1_LOW_HIGH ACK1 端子をクリア
        Low->High
            PCI2702C_PULS_ACK1_HIGH_LOW ACK1 端子をクリア
        High->Low
            out1
                PCI2702C_PULS_OUT1_NOACTION なにもしない
                PCI2702C_PULS_OUT1_LEVEL_HIGH High レベル出力
                PCI2702C_PULS_OUT1_LEVEL_LOW Low レベル出力
                PCI2702C_PULS_OUT1_PULSE_LOW Low パルスを出力
            out2
                PCI2702C_PULS_OUT2_NOACTION なにもしない
                PCI2702C_PULS_OUT2_LEVEL_HIGH High レベル出力
                PCI2702C_PULS_OUT2_LEVEL_LOW Low レベル出力
                PCI2702C_PULS_OUT2_PULSE_LOW Low パルスを出力

割り込みサービス関数 割り込んだ際の割り込み要因レジスタ(オフセット 0x0c)
の値を戻す
int pci2702c_intr_service
(
    int fd,
    unsigned int *iflag,
    int *pending
);

    戻り値
        エラーなら-1 成功なら 0

    引数
        fd ファイルディスクリプタ番号
        iflag 値を戻す変数
        pending 保留されている割り込みの数を戻す変数
        割り込みを禁止する

int pci2702c_disable_intrrupt

```

```

(
    int fd,
    int mask
);
戻り値
    エラーなら-1 成功なら 0
引数
    fd ファイルディスクリプタ番号
    mask 割り込みを禁止するビットマスク 以下のいずれかを指定する
        PCI2702C_IMASK_STB1      STB1 からの入力信号
        PCI2702C_IMASK_IRIN1     IR.IN1 からの入力信号
        PCI2702C_IMASK_IRIN2     IR.IN2 からの入力信号
        PCI2702C_IMASK_TIMER     タイマー割り込み
        PCI2702C_IMASK_RESET     リセット割り込み

割り込みを許可する
int pci2702c_enable_intrrupt
(
    int fd,
    int mask
);
戻り値
    エラーなら-1 成功なら 0
引数
    fd ファイルディスクリプタ番号
    mask 割り込みを許可するビットマスク 以下のいずれかを指定する
        PCI2702C_IMASK_STB1      STB1 からの入力信号
        PCI2702C_IMASK_IRIN1     IR.IN1 からの入力信号
        PCI2702C_IMASK_IRIN2     IR.IN2 からの入力信号
        PCI2702C_IMASK_TIMER     タイマー割り込み
        PCI2702C_IMASK_RESET     リセット割り込み
        PCI2702C_IMASK_ALL       上記のすべて

インターバルタイマーをセットする
int pci2702c_set_interval_timer(int fd,unsigned int base,unsigned int
div);
戻り値
    エラーなら-1 成功なら 0
引数
    fd ファイルディスクリプタ番号
    base ベースクロック値 以下のいずれかを指定する
        PCI2702C_TIMER_BASE_STOP      停止
        PCI2702C_TIMER_BASE_010USEC   10 マイクロ秒
        PCI2702C_TIMER_BASE_100USEC   100 マイクロ秒
        PCI2702C_TIMER_BASE_001MSEC   1 ミリ秒
        PCI2702C_TIMER_BASE_010MSEC   10 ミリ秒
        PCI2702C_TIMER_BASE_100MSEC   100 ミリ秒
    div ベースクロックを分周する値 カウントダウンし 0 の時割り込
みが発生する
        最大15分周しかできない

インターバルタイマーの現在値を読み出す
int pci2702c_get_interval_timer(int fd,unsigned int *count);
戻り値
    エラーなら-1 成功なら 0

```

#### 引数

fd ファイルディスクリプタ番号  
count 値を読み出す変数へのポインタ

汎用関数 オフセット値を指定してレジスタの値を読み出す

```
int pci2702c_get_ioport(int fd,int offset,unsigned int *value);
```

戻り値

エラーなら-1 成功なら 0

#### 引数

fd ファイルディスクリプタ番号  
offset レジスタオフセット  
value 値を読み出す変数へのポインタ

汎用関数 オフセット値を指定してレジスタに値を書き出す

```
int pci2702c_set_ioport(int fd,int offset,unsigned int *value);
```

戻り値

エラーなら-1 成功なら 0

#### 引数

fd ファイルディスクリプタ番号  
offset レジスタオフセット  
value 値を出す変数へのポインタ

チャンネルを指定して入力データを読み出す

```
int pci2702c_read_data(int fd,int ch,unsigned char *data);
```

戻り値

エラーなら-1 成功なら 0

#### 引数

fd ファイルディスクリプタ番号  
ch チャンネル  
data 値を出す変数へのポインタ

すべてのチャンネルの入力データを読み出す

```
int pci2702c_read_data_all(int fd,unsigned char *data);
```

戻り値

エラーなら-1 成功なら 0

#### 引数

fd ファイルディスクリプタ番号  
data 値を出す配列変数へのポインタ

DIP スイッチの読み込み

```
int pci2702c_get_sw(int fd,unsigned long int *data);
```

戻り値

エラーなら-1 成功なら 0

#### 引数

fd ファイルディスクリプタ番号  
data 出力変数へのポインタ

#### SEE ALSO

/usr/local/CNC/drivers/extmem/interface/pci2702c 下のプログラム

#### AUTHORS

Copyright (C) 1995-2016 Concurrent Real Time Inc.

28 Apr 2016

pci2702c(3)