# AIO-160802L Board Support Package Installation on RedHawk

Release Notes  Revision B

August 29, 2022


concurrent
REAL-TIME

# 1. はじめに

本書は、Concurrent Real Time Inc(CCRT)の RedHawk 上で動作する、コンテック社製 AIO- 160802L PCI Express 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

AIO- 160802L 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事が必要です。

- AIO-160802L ボード
- RedHawk 6.x 以上
- Extmem version 6.7 以上

AIO-160802Lは、高精度アナログ入出力、デジタル入出力、カウンタを搭載したPCI Expressバス準拠のマルチファンクションボードです。

# 3. インストール方法

***AIO-160802L 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。

***AIO-160802L 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ことが必要です。

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**_RH***x.y-z***.x86_64.rpm

**AIO-160802L BSP 実行パッケージのインストール**

# rpm -ivh bin-aio160802l-**X.Y**_RH***x.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- aio160802l-**X.Y**_RH***x.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

**AIO-160802L BSP 実行パッケージのインストール**

# apt install ./bin-aio160802l-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-aio160802l-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

**AIO-160802L BSP 実行パッケージのインストール**
# apt install ./bin-aio160802l-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-aio160802l-rhx.y_X.Y_arm64.deb
# umount /mnt

(***x.y*** は RedHawk のバージョン番号であり、6.0,6.3,6.5,7.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。)

***AIO-160802L BSP*** パッケージは ***/usr/local/CNC/drivers/extmem/interface/aio160802l*** 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

***AIO-160802L 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/ aio160802L*** 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- aio160802l-**X.Y**_RH***x.y-z***.x86_64 *(開発パッケージの削除)*
# rpm -e bin-aio160802l-**X.Y**_RH***x.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-aio160802l-**X.Y**_RH***x.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-aio160802l-rhx.y *(開発パッケージの削除)*
# apt purge bin-aio160802l-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-aio160802l-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-aio160802l-rhx.y *(開発パッケージの削除)*
# apt purge bin-aio160802l-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-aio160802l-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 aio160802L

```
aio160802l(3)                                                    aio160802l(3)

NAME
       aio160802l - 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
       aio160802l は、external memory ドライバを利用した AIO160802L ボードアクセスライブラリです。

       #include <sys/aio160802l.h>
       gcc [options ...] file -laio160802l -lextmem ...

       ********************************************************************************
       AIO160802L
       ********************************************************************************

       マルチファンクションボード AIO160802L をアクセスする関数群
       API のパラメータ変数がポインタでない場合は、unsigned long int に定義されていても、有効な値は 32 ビットである。

       デバイスの非初期化処理
       int aio160802l_reset(int fd,int port);
       port は、以下のビットオアまたは、全てを示す AIO160802L_ALL_PORT である
            AIO160802L_AI_PORT
            AIO160802L_AO_PORT
            AIO160802L_DI_PORT
            AIO160802L_DO_PORT
            AIO160802L_COUNTER_PORT
            AIO160802L_MEMORY_PORT
            AIO160802L_ECU_PORT

       int aio160802l_uninit(int fd); 注意：aio160802l_uninit()関数は aio160802l_reset()を呼び出す。
       デバイスの初期化処理と、割り込みサービスの登録
       int aio160802l_init(int fd,int option);
       割り込みサービスの登録を行い、ボードを初期化する。
       割り込み時のレジスタの値は、aio160802l_intr_service()API で得られ、割り込みフラグクリアは、extmem デバイスドライバによって終了している。
       注意：DO は割り込まない。
       option 1 を指定すると以下の情報が表示される
              BAR0 I/O Region addr 0x0000ccc0 offset 0x00000000 64 bytes
       注意：aio160802l_init()関数は aio160802l_reset()を呼び出す。

       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 aio160802l_get_ioport_long(int fd,int offset,unsigned long int *value);
int aio160802l_get_ioport_short(int fd,int offset,unsigned long int *value);
aio160802l_get_ioport_long()は、32bit、aio160802l_get_ioport_short()が16bitである。

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 aio160802l_set_ioport_long(int fd,int offset,unsigned long int *value);
int aio160802l_set_ioport_short(int fd,int offset,unsigned long int *value);
aio160802l_set_ioport_long()は、32bit、aio160802l_set_ioport_short()が16bitである。

割り込みハンドラの登録と、割り込みの許可
int aio160802l_setup_signal(int fd,void (*interrupt_hadler)(int,siginfo_t *,void *),unsigned long int aimask,unsigned long int aomask,unsigned long int dimask,unsigned long int comask,unsigned long int memask);
各マスク値は、1である場合に許可、0である場合に不許可である(実際の書き出しには、NOTの値を使用する)
この関数の呼び出しによって、ボードのIRQ信号によって、シグナルハンドラが起動されるようになる。
また、本関数は、aio160802l_enable_intrrupt()を呼び出す。

割り込みを許可する
int aio160802l_enable_intrrupt(int fd,unsigned long int aimask,unsigned long int aomask,unsigned long int dimask,unsigned long int comask,unsigned long int memask);
各マスク値は、1である場合に許可、0である場合に不許可である(実際の書き出しには、NOTの値を使用する)

割り込みを禁止する
int aio160802l_disable_intrrupt (int fd,unsigned long int aimask,unsigned long int aomask,unsigned long int dimask,unsigned long int comask,unsigned long int memask);
各マスク値は、1である場合に不許可、0である場合に許可である(実際の書き出にその値を使用する)

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の値を戻す
int aio160802l_intr_service(int fd,unsigned long int *iflag, unsigned long int *aiflag,unsigned long int *aoflag,unsigned long int *diflag,unsigned long int *coflag,unsigned long int *meflag,unsigned long int *pending);
通常本関数は、aio160802l_setup_signal()関数で登録したシグナルハンドラ内で呼び出される。
ファイルディスクリプタ(fd)以外の変数は、割り込み時のレジスタ値である。
iflag:ECU機能確認ポート0x38の値
aiflag:AI 関連のフラグ
aoflag:AO 関連のフラグ
diflag:DI 関連のフラグ
coflag:COUNTER 関連のフラグ
meflag:MEMORY 関連のフラグ
pending:処理がペンディングされている数

イベントコントローラ入出力信号の結線を行う
int aio160802l_set_signal_assign(int fd,unsigned long int destnation_signal,unsigned long int source_signal);

イベントコントローラ入出力信号の結線を得る
int aio160802l_get_signal_assign(int fd,unsigned long int destnation_signal,unsigned long int source_signal,unsigned long int *destnation_signal_ret,unsigned long int *source_signal_ret);
設定できる信号は以下の通りだが、下記組み合わせが存在し、不可能な組み合わせではエラー(-1)になる。
destnation_signal

| | |
|---|---|
| DST_SIGNAL_AI_STORE_ENABLE_TRIGGER | AI 格納許可トリガ[0x00] |
| DST_SIGNAL_AI_STORE_DISABL_TRIGGER | AI 格納不許可トリガ[0x02] |
| DST_SIGNAL_SAMPLING_CLOCK | AI サンプリングクロック[0x04] |
| DST_SIGNAL_AO_UPDATE_ENABLE_TRIGGER | AO 更新許可トリガ[0x20] |
| DST_SIGNAL_AO_UPDATE_DISABLE_TRIGGER | AO 更新不許可トリガ[0x22] |
| DST_SIGNAL_UPDATE_CLOCK | AO 更新クロック[0x24] |
| DST_SIGNAL_CNTEXTSTATUS00 | CNT 外部ステータス[0x74] |

```
        DST_SIGNAL_COUNT_START_TRIGGER0          CNT 開始トリガ[0x80]
        DST_SIGNAL_COUNT_STOP_TRIGGER0           CNT 停止トリガ[0x82]
        DST_SIGNAL_COUNT_UP0                     CNT アップカウント[0x86]
    source_signal
        SRC_SIGNAL_NO_CONNECTION                 Non Connection[0x000]
        SRC_SIGNAL_AI_INTERNAL_CLOCK             AI 内部サンプリングクロック[0x004]
        SRC_SIGNAL_AI_BEFORE_TRRIGER_NUM_END     AI ビフォートリガサンプリング回数終了[0x011]
        SRC_SIGNAL_AO_INTERNAL_CLOCK             AO 内部更新クロック[0x042]
        SRC_SIGNAL_AO_BEFORE_TRIGGER_NUM_END     AO ビフォートリガ更新回数終了[0x050]
        SRC_SIGNAL_DI_AI_EXTCLK_START_EDGE       AI 外部開始エッジ[0x090]
        SRC_SIGNAL_DI_AI_EXTCLK_STOP_EGDE        AI 外部停止エッジ[0x091]
        SRC_SIGNAL_DI_AI_EXTCLK                  AI 外部クロック[0x092]
        SRC_SIGNAL_DI_AO_EXTCLK_START_EDGE       AO 外部開始エッジ[0x093]
        SRC_SIGNAL_DI_AO_EXTCLK_STOP_EDGE        AO 外部停止エッジ[0x094]
        SRC_SIGNAL_DI_AO_EXTCLK                  AO 外部クロック[0x095]
        SRC_SIGNAL_DI_CNT_EXTUCLK1               CNT 外部アップカウントクロック[0x096]
        SRC_SIGNAL_COUNT_INTERNAL0               CNT 内部サンプリングクロック[0x110]
        SRC_SIGNAL_COUNT_COUNTUP0                CNT アップカウント一致 [0x120]
        SRC_SIGNAL_COUNT_BUSY0                   CNT ビジィ[0x131]
        SRC_SIGNAL_MEMORY_AO_DATA_EMPTY          AO 更新データエンプティ[0x160]
        SRC_SIGNAL_ECU_GENERAL_COMMAND0          汎用コマンド [0x180]
```

| | | ディスティネーションシグナル | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | AI | | | AO | | | CNT | | | | |
| ソースシグナル | | 格納許可トリガ 0x00 | 格納不許可トリガ 0x02 | サンプリングクロック 0x04 | 更新許可トリガ 0x20 | 更新不許可トリガ 0x22 | 更新クロック 0x24 | 外部ステータス 0x74 | 開始トリガ 0x80 | 停止トリガ 0x82 | アップカウント 0x86 | |
| Non Connection | 0x000 | ◎ | ◎ | ◎ | ◎ | ◎ | ◎ | ◎ | ◎ | ◎ | ◎ | |
| AI 内部サンプリングクロック | 0x004 | | | ◎ | | | | | | | | |
| AI ビフォートリガサンプリング回数終了 | 0x011 | | ◎ | | | | | | | | | |
| AO 内部更新クロック | 0x042 | | | | | | ◎ | | | | | |
| AO ビフォートリガ更新回数終了 | 0x050 | | | | | ◎ | | | | | | |
| AI 外部開始エッジ | 0x090 | ◎ | | | | | | | | | | |
| AI 外部停止エッジ | 0x091 | | ◎ | | | | | | | | | |
| AI 外部クロック | 0x092 | | | ◎ | | | | | | | | |
| AO 外部開始エッジ | 0x093 | | | | ◎ | | | | | | | |
| AO 外部停止エッジ | 0x094 | | | | | ◎ | | | | | | |
| AO 外部クロック | 0x095 | | | | | | ◎ | | | | | |

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| CNT 外部アップカウントクロック | 0x096 | | | | | | | | | ◎ | |
| CNT 内部サンプリングクロック | 0x110 | | | | | | | | | ◎ | |
| CNT アップカウント一致 | 0x120 | | | | | | | | | ◎ | |
| CNT ビジィ | 0x131 | | | | | | ◎ | | | | |
| AO 更新データエンプティ | 0x160 | | | | | | ◎ | | | | |
| 汎用コマンド | 0x180 | ◎ | ◎ | | ◎ | ◎ | | | ◎ | ◎ | |

イベントコントローラ(0x38)に、AI 異常停止設定コマンド(0x10)を出力後、ECU 設定データポート(0x3C)に value を出力する。
value は、AIO160802L_ECU_GENERAL0_COMMAND_AI_CLK_ERROR(0x00000001)が AI CLK Error で、AIO160802L_ECU_GENERAL0_COMMAND_AI_OVERFLOW(0x00010000)が AI Over Flow である。

int aio160802l_ao_abort_settings(int fd,unsigned long int value);
イベントコントローラ(0x38)に、AO 異常停止設定コマンド(0x11)を出力後、ECU 設定データポート(0x3C)に value を出力する。
value は、AIO160802L_ECU_GENERAL0_COMMAND_AO_CLK_ERROR(0x00000001) が AO CLK Error,AIO160802L_ECU_GENERAL0_COMMAND_AO_EMPTY(0x00010000) が、AO Empty で、AIO160802L_ECU_GENERAL0_COMMAND_AO_EXT_STOP(0x01000000)が AO Ext Stop である。

int aio160802l_ai_setcommand(int fd,unsigned long int command,unsigned long int value)
アナログ入力機能設定コマンド
アナログ入力コマンドポート(0x30)に、command を(32bit)出力する。
パラメータを伴うコマンドの場合には、 アナログ入力設定データポート(0x34)に、value(32bit)を出力する。
command,value は以下の通り

| command | : value |
|---|---|
| AIO160802L_AI_RESET_COMMAND | : AI 初期化コマンド(0x10000000) |
| | value=0 を設定 |
| AIO160802L_AI_GATE_OPEN_COMMAND | : AI 内部ゲートオープンコマンド(0x10000001) |
| | value=0 を設定 |
| AIO160802L_AI_ABORT_COMMAND | : AI 強制停止コマンド(0x10000002) |
| | value=0 を設定 |
| AIO160802L_AI_INTCLK_COMMAND | : AI 用内部クロック設定(0x10000003) |
| | value=内部クロックのパルス周期 = 周期(ms)/25 -1 |
| AIO160802L_AI_CHNUM_COMMAND | : AI 格納チャネル数設定(0x10000005) |
| | value=チャンネル数-1(最大 7) |
| AIO160802L_AI_CHSEQ_COMMAND | : AI チャネル設定(0x1000000C) |
| | value=AIO160802L_AI_CHSEQ_COMMAND_MULTI または AIO160802L_AI_CHSEQ_COMMAND_SINGLE |
| AIO160802L_AI_BEFTRG_COMMAND | : AI ビフォートリガサンプリング回数設定(0x10000009) |
| | value = ビフォートリガサンプリング回数-1 |

int aio160802l_ao_setcommand(int fd,unsigned long int command,unsigned long int value)
アナログ出力コマンドポート(0x30)に、command を(32bit)出力する。
パラメータを伴うコマンドの場合には、 アナログ入力設定データポート(0x34)に、value(32bit)を出力する。

| | |
|---|---|
| AIO160802L_AO_RESET_COMMAND | : AO 初期化コマンド(0x20000000) |
| | value=0 を設定 |
| AIO160802L_AO_GATE_OPEN_COMMAND | : AO 内部ゲートオープンコマンド(0x20000001) |
| | value=0 を設定 |
| AIO160802L_AO_ABORT_COMMAND | : AO 強制停止コマンド(0x20000002) |
| | value=0 を設定 |
| AIO160802L_AO_INTCLK_COMMAND | : AO 用内部クロック設定(0x20000003) |

value=内部クロックのパルス周期 = 周期(ms)/25 -1
AIO160802L_AO_CHNUM_COMMAND : AO 更新チャネル数設定(0x20000005)
value=チャンネル数-1(最大 1)
AIO160802L_AO_CHSEQ_COMMAND : AO 出力切り替えコマンド(0x2000000C)
value=AIO160802L_AO_CHSEQ_COMMAND_SINGLE または AIO160802L_AO_CHSEQ_COMMAND_MULTI
AIO160802L_AO_BEFTRG_COMMAND : AO ビフォートリガ更新回数設定(0x20000009)
value = ビフォートリガ更新回数-1

int aio160802l_memory_setcommand(int fd,unsigned long int command,unsigned long int value)
メモリコマンドポート(0x30)に、command を(32bit)出力する。
AIO160802L_MEMORY_AO_MEM_COMPARETE_TYPE_COMMAND の場合には、 アナログ入力設定データポート(0x34)に、value(32bit)を出力する。
command は以下の通り

command : value
AIO160802L_MEMORY_RESET_COMMAND : MEM 初期化コマンド(0x60000000)
value=0 を設定
AIO160802L_MEMORY_AI_MEM_CLEAR_COMMAND : AI MEM クリア(0x60000007)
value=0 を設定
AIO160802L_MEMORY_AO_MEM_CLEAR_COMMAND : AO MEM クリア(0x60000008)
value=0 を設定
AIO160802L_MEMORY_AI_MEM_COMPARETE_TYPE_COMMAND : AI 用 MEM 比較タイプ設定(0x60000010)
value=0 を設定
このコマンドでは、バッファメモリに格納される任意のデータ数で
フラグをセットする条件のうち比較タイプを設定します。
このコマンドでは、"AI 比較データ数"フラグを生成するためにあります。
A/D コンバータからアナログ入力データが入力された場合に
"AI 比較データ数ホールド"フラグがセットされます。
AI 機能のデータ入力ポートからアナログ入力データを読み込みによって、
データ数が減り、設定したデータ数を通過した場合は、このフラグはセットされません。
このコマンドの次に AIO160802L_MEMORY_AI_MEM_COMPARETE_DATA_COMMAND(AI MEM 比較データ設定)が必要です。
リセットは ECU の機能にあります。

AIO160802L_MEMORY_AO_MEM_COMPARETE_TYPE_COMMAND : AO 用 MEM 比較タイプ設定(0x60000020)
value=1(下方一致) または value=2(設定転送数毎)
このコマンドでは、バッファメモリに格納される任意のデータ数/ポインタで
フラグをセットする条件のうち比較タイプを設定します。
このコマンドでは、"AO 比較データ数ホールド"フラグもしくは"AO 転送数"フラグを
生成するためにあります。
"下方一致" の場合:
アナログ出力データが D/A コンバータへ出力された場合に"AO 比較データ数ホールド"フラグがセットされます。
AO 機能のアナログ出力ポートからアナログ出力データを追加の書き込みによって、
データ数が増え、設定したデータ数を通過した場合は、このフラグはセットされません。
"設定転送数毎"の場合:
ポインタの移動量が設定した値に等しくなった場合に “AO 転送数” フラグがセットされます。
このコマンドの次に AIO160802L_MEMORY_AO_MEM_COMPARETE_DATA_COMMAND(AO MEM 比較データ設定)が必要です。

int aio160802l_memory_getcommand(int fd,unsigned long int command,unsigned long int *value) メモリコマンドポート(0x30)に、command を(32bit)出力し、メモリ設定データポート(0x34)から、value(32bit)を入力する。

AIO160802L_MEMORY_AI_MEM_FIFOCOUNT_COMMAND:AI FIFO カウンタ確認コマンド(0x6000000B)
value に AI FIFO カウンタの値
AIO160802L_MEMORY_AO_MEM_FIFOCOUNT_COMMAND:AO FIFO カウンタ確認コマンド(0x6000000C)
value に AO FIFO カウンタの値

int aio160802l_memory_setcommand2(int fd,unsigned long int command,unsigned long int value1,unsigned long int value2)
メモリコマンドポート(0x30)に、command を(32bit)出力し、メモリ設定データポート LO(0x34)に、value1(16bit)を出力後、メモリ設定データポート HI(0x36)に、value2(16bit)を出力する。

ただし、AIO160802L_MEMORY_AO_MEM_COMPARETE_DATA_COMMAND の場合には、メモリ設定データポート(0x34)に、value1(32bit)を出力後、メモリ設定データポート(0x34)に、value2(32bit)を出力する。

command,value1,value2 は以下の通り

AIO160802L_MEMORY_AI_MEM_COMPARETE_DATA_COMMAND:AI 用 MEM 比較データ設定(0x60000011)
value1:比較番号
0:比較データ
上記以外無効
value2:設定データ=(比較データ-1)

AIO160802L_MEMORY_AO_MEM_COMPARETE_DATA_COMMAND : AO 用 MEM 比較データ設定(0x60000021)
value1:比較番号
0:データ数
1:転送数
value2:設定データ=(比較データ数+1)

int aio160802l_di_setcommand2(int fd,int command,int value1,int value2);

デジタル入力コマンドポート(0x30)に、command を(32bit)出力し、デジタル入力設定データポート L0(0x34)に、value1(16bit)を出力後、デジタル入力設定データポート HI(0x36)に、value2(16bit)を出力する。

command,value1,value2 は以下の通り

AIO160802L_DI_RESET_COMMAND: DI 初期化コマンド(0x30000000)
value1=0,value2=0 を設定

AIO160802L_DI_EDGE_DETECT_COMMAND: DI エッジ検出設定(0x30000001)
value1=以下の外部から入力されるディジタル入力タイプを選択
AIO160802L_DI_EDGE_TYPE_AI(0x1)
AIO160802L_DI_EDGE_TYPE_AO(0x2)
AIO160802L_DI_EDGE_TYPE_CNT(0x3)
value2=ディジタル入力の以下の検出ビットを
AIO160802L_DI_EDGE_TYPE_NONE
AIO160802L_DI_EDGE_TYPE_RISE
AIO160802L_DI_EDGE_TYPE_FALL
AIO160802L_DI_EDGE_TYPE_BOTH
下記のポート分シフトした値をビットオアした値を設定
AIO160802L_DI_EDGE_BIT00_SHIFT
AIO160802L_DI_EDGE_BIT01_SHIFT
AIO160802L_DI_EDGE_BIT02_SHIFT
例えば、(AIO160802L_DI_EDGE_TYPE_RISE<<AIO160802L_DI_EDGE_BIT00_SHIFT)|...

AIO160802L_DI_DIGITAL_FILTER_COMMAND: DI Digital Filter 設定(0x30000002)
value1=以下のディジタルフィルタを入れる信号名を選択
AIO160802L_DI_DIGITAL_FILTER_DI DI Digital Filter
AIO160802L_DI_DIGITAL_FILTER_AI AI Digital Filter
AIO160802L_DI_DIGITAL_FILTER_AO AO Digital Filter
AIO160802L_DI_DIGITAL_FILTER_CNT CNTDigital Filter
value2=以下のディジタルフィルタ時間を選択
AIO160802L_DI_DIGITAL_FILTER_OFF Digital Filter なし
AIO160802L_DI_DIGITAL_FILTER_1U Digital Filter 1 マイクロ秒

int aio160802l_do_setcommand(int fd,int command);

デジタル出力コマンドポート(0x30)に、AIO160802L_DO_RESET_COMMAND を(32bit)出力する

command は以下の通り

AIO160802L_DO_RESET_COMMAND DO 初期化コマンド (0x40000000)
本ボードは設定が固定になっています。設定の変更はできません。

int aio160802l_cnt_setcommand(int fd,int command,int value);

カウンタコマンドポート(0x30)に、command を(32bit)出力し、カウンタ設定データポート(0x34)に、value(32bit)を出力する。

AIO160802L_COUNTER_RESET_COMMAND: CNT 初期化コマンド(0x50000000)

value=0 を設定
AIO160802L_COUNTER_GATE_OPEN_COMMAND:CNT 内部ゲートオープンコマンド(0x50000001)
value=0 or 1：内部ゲートオープンするかしないかを設定
AIO160802L_COUNTER_ABORT_COMMAND: CNT 強制停止コマンド(0x50000002)
value=0 or 1：強制停止するかしないかを設定
int aio160802l_get_counter(int fd,int *value);
カウンタコマンドポート(0x30)に、AIO160802L_COUNTER_INTCLK_COMMAND(0x50000003)を(32bit)出力し、カウンタ設定データポート L0(0x34)に、内部クロックの値を読み出し*value に設定する。
int aio160802l_set_counter(int fd,int ch,int value);
カウンタコマンドポート(0x30)に、AIO160802L_COUNTER_INTCLK_COMMAND(0x50000003)を(32bit)出力し、カウンタ設定データポート L0(0x34)に、 value の値を設定する。
int aio160802l_set_counter_comparete(int fd,int value);
カウンタコマンドポート(0x30)に、AIO160802L_COUNTER_COMPARETE_COMMAND(0x50000007)を(32bit)出力し、カウンタ設定データポート(0x34)に、value(32bit)を出力し、比較データを設定する。
int aio160802l_ai_read(int fd,unsigned long int *data,int n);
AD 変換器から n データを読み出す
内部で、aio160802l_ai_start()を最初に呼び出す
注意：この関数で無限サンプリングを実現することはできません。
無限サンプリングを行うためのレジスタ設定例は aio160802l_samples/intr.c を参考にしてください。
int aio160802l_ai_read_data(int fd,unsigned long int *data);
AD 変換器から 1 データだけ読み出す
int aio160802l_ai_read_csr(int fd,unsigned short int *data);
AD 変換器の CSR(0x04)から 1 データだけ読み出す
int aio160802l_ai_calibration(int fd,unsigned int value);
AI のキャリブレーションを行う
value:固定値のため以下の値を設定する
AIO160802L_AI_CAL_RANGE_M10VtoP10V -10 to +10
返り値:-2:タイムアウト(5 秒)
-1:それ以外 errno 参照
AO をスタートする
(1)アナログ出力コマンドポート(0x30)に、AIO160802L_AO_GATE_OPEN_COMMAND(0x20000001)を(32bit)出力する。
(2)ECU コマンドポート(0x38)に、AIO160802L_ECU_GENERAL0_COMMAND(0x00000005)を出力し、AO をスタートする。
(3)ECU コマンドポート(0x38)に、AIO160802L_ECU_AO_FLAGRESET_IRQMASK_COMMAND(0x20000000)を出力し、
アナログ出力フラグ割り込みマスクポート(0x3C)をスキャンし、AIO160802L_AO_INTR_MASK_MOTION_END(0x80000000)ビットが真になる間、フラグスキャンループを行う。
int aio160802l_ao_calibration(int fd,unsigned int range,unsigned int ch)
AO のキャリブレーションを行う
以下の値を設定する
range
AIO160802L_AO_CAL_RANGE_M10VtoP10V -10 to +10
ch:
AIO160802L_AO_CAL_CH0
AIO160802L_AO_CAL_CH1
返り値:-2:タイムアウト(5 秒)
-1:それ以外 errno 参照 int aio160802l_di_setmask(int fd,int value);
DI のマスク(0x14)を設定する
int aio160802l_di_getvalue(int fd,int *value);
DI の値(0x10)を読み出す
int aio160802l_do_setmask(int fd,int value);
DO のマスク(0x1C)を設定する
int aio160802l_do_setvalue(int fd,int value);
DO(0x18)に値を設定する

SEE ALSO

/usr/local/CNC/drivers/extmem/contec/aio160802l 下のプログラム

AUTHORS