

PL-UDD00

取扱説明書

PL-UDD00 は(株)デジタル製パネルコンピュータ（以下 PL-3700 と称します）用の入出力各 16 点のフォトアイソレーション型 DIO ユニットです。入力 16 点中 2 点は、割り込みを発生することが可能です。PL-UDD00 は PL-3700 の拡張 BUS に装着して使用します。

対応機種：PL-3700 シリーズ

下記のほか、本書に記載の商品名は、各社の商標・登録商標です。

Pro-face：(株)デジタル

MS-DOS：米国 Microsoft 社



警告 安全に関する使用上の注意

- ・ PL-3700 への取り付け時は感電の危険性がありますので、PL-3700 に電源が供給されていないことを必ず確認して取り付けてください。
- ・ PL-UDD00 は改造しないでください。火災、感電の恐れがあります。

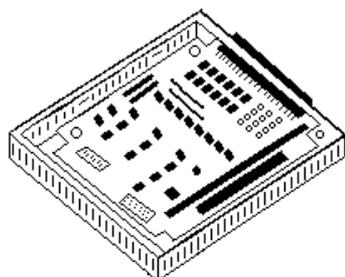
故障しないために

- ・ PL-UDD00 は精密機器ですので、衝撃を与えないでください。
- ・ PL-UDD00 に水など液状のものや金属が付着しないようにしてください。故障や感電の原因になります。
- ・ 直射日光に当たる場所や高温の場所、ほこりの多い場所、振動の加わる場所での保管および使用は避けてください。
- ・ 薬品が気化し、発散している空気や薬品が付着する場所での保管および使用は避けてください。
- ・ 修理や改造を行わないでください。
- ・ ファイル破損を防ぐため、必ず OS を終了してからコンピュータの電源を切るようにしてください。

梱包内容

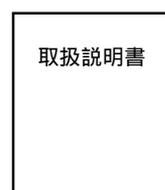
梱包箱には、以下のものが入っています。ご使用前に必ず確認してください。

PL-UDD00 本体

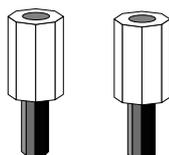


取扱説明書 1枚(本書)

PL-UDD00 取扱説明書



スペーサ 2本



品質や梱包などには出荷時に際し、万全を期しておりますが、万一破損や部品不足、その他お気付きの点がありましたら、直ちに販売店までご連絡くださいますようお願いいたします。

1 ハードウェア仕様

機能仕様

入力点数	16点(8点1コモン)
入力電圧	DC +12V ~ +24V
ON/OFF電圧	ON:+10V(最低)、OFF:+8V(最高)
逆電圧保護	あり
過電圧保護	なし
絶縁方法	フォトカプラ(NEC PS2801)
入力フィルタ	MC14490 チャタリング除去回路
入力割り込み	2点(IN0、IN8) 立ち上がり、立ち下がりエッジ切り替え可能 IRQ9、10、11、15(デフォルト)より選択
出力点数	16点(8点1コモン)
定格回路電圧	DC+5V / +24V
定格出力電流 (ON時)	200mA(最大)8点同時動作時
ターンON時間	5 μ sec(平均)、10 μ sec(最大)、I=20mA
ターンOFF時間	0.5msec(平均)、1msec(最大)

設定環境仕様

使用周囲温度	0 ~ 40
保存周囲温度	-10 ~ 60
耐ノイズ性 (ノイズシュミレータによる)	ノイズ電圧：1000Vp-p パルス幅：50ns、500ns、1 μ s
静電耐圧	4KV



- ・ 上記環境仕様はPL-3700 に組み込んだ状態での仕様です。

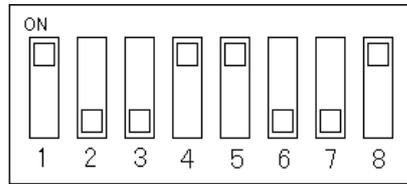
入出力コネクタ仕様

- ・ ユニット側使用コネクタ FCN-365P040-AU(オス)
- ・ ケーブル用コネクタ FCN-361J040-AU(メス半田付け)
FCN-360C040-D(カバー)
FCN-363J040(メス圧着タイプ)
すべて富士通(株)製

A1		B1	
ピン番号	信号名	ピン番号	信号名
A1	OUT0	B1	OUT1
A2	OUT2	B2	OUT3
A3	OUT4	B3	OUT5
A4	OUT6	B4	OUT7
A5	COM0	B5	COM0
A6	OUT8	B6	OUT9
A7	OUT10	B7	OUT11
A8	OUT12	B8	OUT13
A9	OUT14	B9	OUT15
A10	COM1	B10	COM1
A11	IN0	B11	IN1
A12	IN2	B12	IN3
A13	IN4	B13	IN5
A14	IN6	B14	IN7
A15	COM2	B15	COM2
A16	IN8	B16	IN9
A17	IN10	B17	IN11
A18	IN12	B18	IN13
A19	IN14	B19	IN15
A20	COM3	B20	COM3

2 ハードウェア設定

I/Oベースアドレス設定



ディップスイッチ SW1

(ON:0 / OFF:1)

スイッチ番号	アドレス	デフォルト	設定
1	A10	ON	3
2	A9	OFF	
3	A8	OFF	
4	A7	ON	330h
5	A6	ON	
6	A5	OFF	
7	A4	OFF	
8	A3	ON	0

入力チャタリング除去回路の時定数の設定 <ショートプラグ J1 >

ピン番号	内容
1-2	400 μsec以下除去(デフォルト)
2-3	4msec以下除去

割り込みレベル設定 <ショートプラグ J2 >

ピン番号	内容
1-2	IRQ15 (デフォルト)
3-4	IRQ11
5-6	IRQ10
7-8	IRQ9

割り込みエッジ設定 <ショートプラグ J3 >

ピン番号	設定	内容		
		入力	入力回路	入力ポート
1-2	ショート	IN8	L H	1 0
	オープン	IN8	H L	0 1
3-4	ショート	IN0	L H	1 0
	オープン	IN0	H L	0 1

I/O アドレス

< I/O ベースアドレスからのオフセット一覧 >

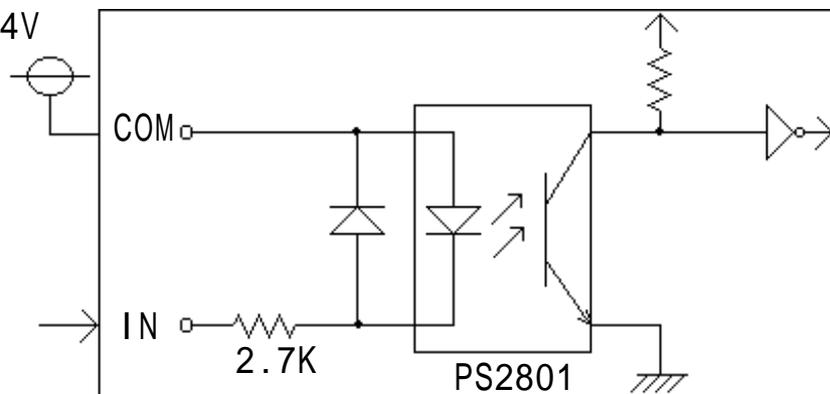
I/Oアドレス	名称	内容
BASE+0 D0 } D7	INPUT LO IN0 } IN7	(READ)入力ポート下位バイト
BASE+1 D0 } D7	INPUT HI IN8 } IN15	(READ)入力ポート上位バイト
BASE+2 D0 } D7	OUTPUT LO OUT0 } OUT7	(READ/WRITE)出力ポート下位バイト
BASE+3 D0 } D7	OUTPUT HI OUT8 } OUT15	(READ/WRITE)出力ポート上位バイト
BASE+4 D0 D1 D2 } D7	INTERRUPT FLAG IN0 IN8 } 未使用	(READ)割り込みフラグ 0で割り込み無し 1で割り込み有り
BASE+4 D0 D1 D2 } D7	INTERRUPT CLEAR IN0 IN8 } 未使用	(WRITE)割り込みクリア 0を書き込むことで 割り込み状態を解除します。
BASE+6 D0 D1 D2 } D7	INTERRUPT MASK IN0 IN8 } 未使用	(READ/WRITE)割り込みマスク 0で割り込み可能 1で割り込み不可(初期値)

4 入出力回路

入力回路

外部

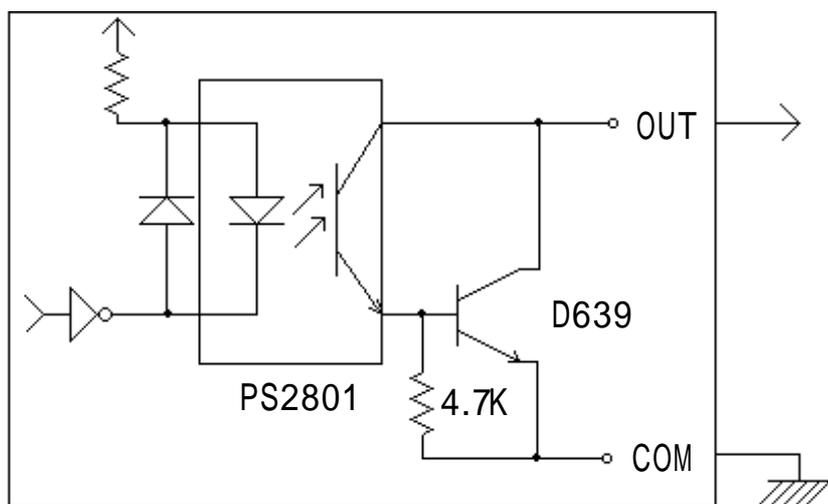
12 ~ 24V



PL-UDD00内部

* IN0 ~ IN15は、LOWで ” 1 ” です。

出力回路



PL-UDD00内部

* OUT0 ~ OUT15は、 ” 1 ” でLOWを出力します。

5 ユニットの取り付け

以下の方法でユニットを PL-3700 へ取り付けてください。



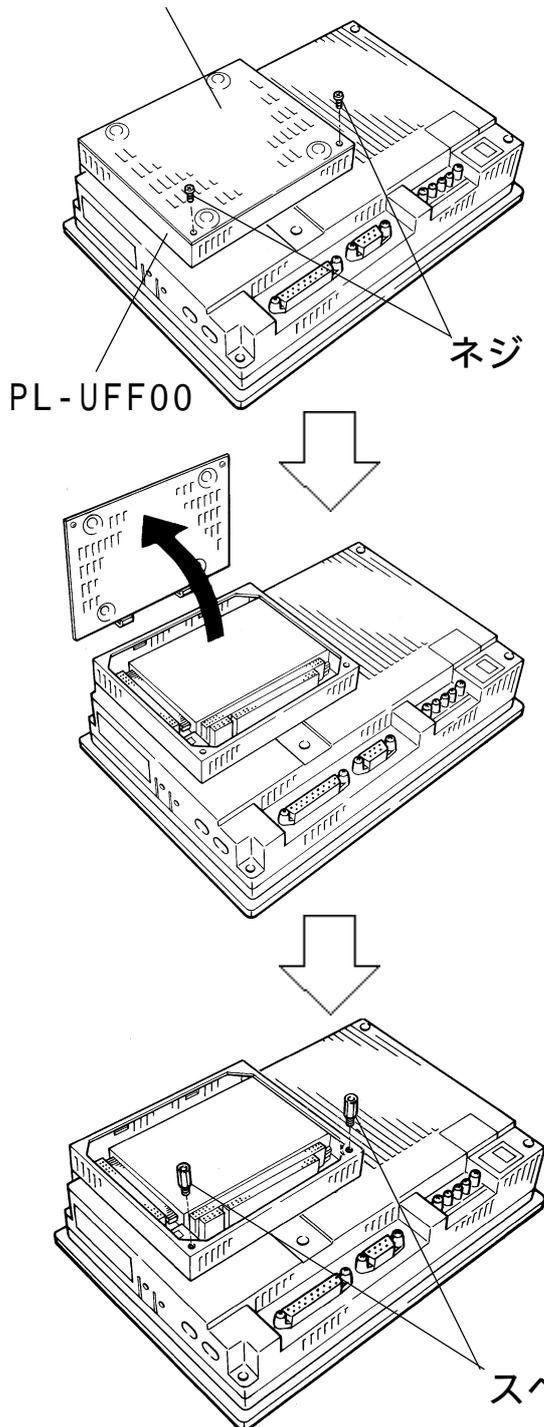
- ・ オプションユニットは運転時には、3ユニット、開発、デバッグ時には4ユニットまで接続可能です。



警告

- ・ 感電の恐れがありますので、必ずPL-3700の電源を切ってから作業を行ってください。

リアメンテナンスハッチ



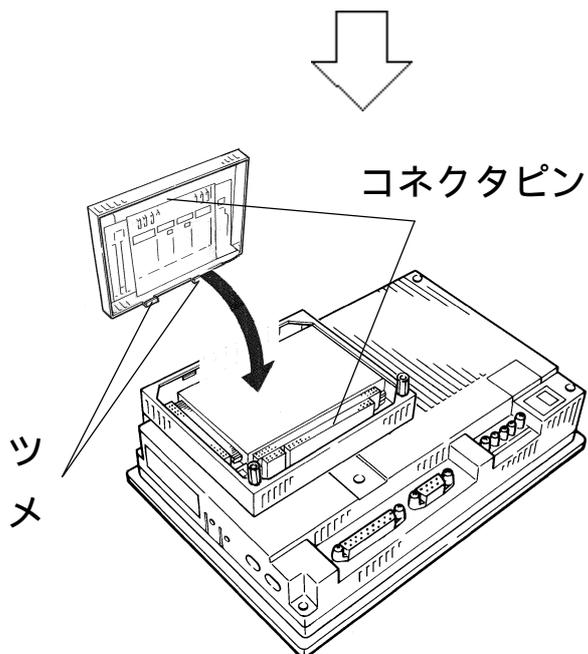
PL-3700 背面のメンテナンスハッチのネジ2カ所を外します。

重要 ・ フラッシュファイルディスクユニットと併用する場合には、フラッシュファイルディスクユニット(PL-UFF00)を必ず第1ユニットとしてPL本体に直接取り付けてください。PL-UDD00を第1ユニットする事はできません。

リアメンテナンスハッチを下から持ち上げながら、矢印方向へずらして取り外します。

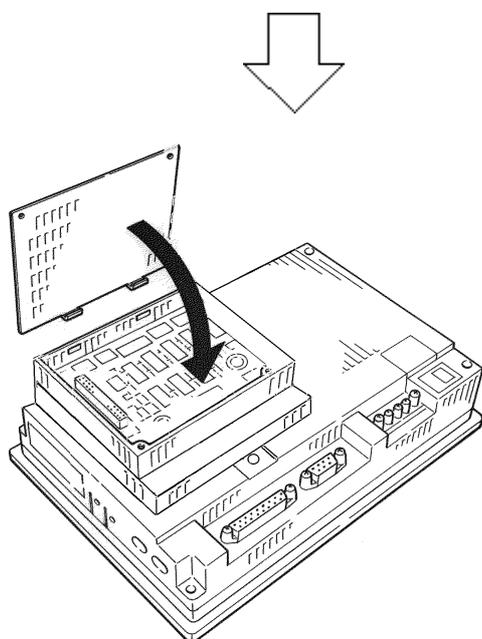
重要 ・ ツメが曲がらないように注意してください。

リアメンテナンスハッチを取り外したネジ穴2カ所に、付属のスペーサを止めます。

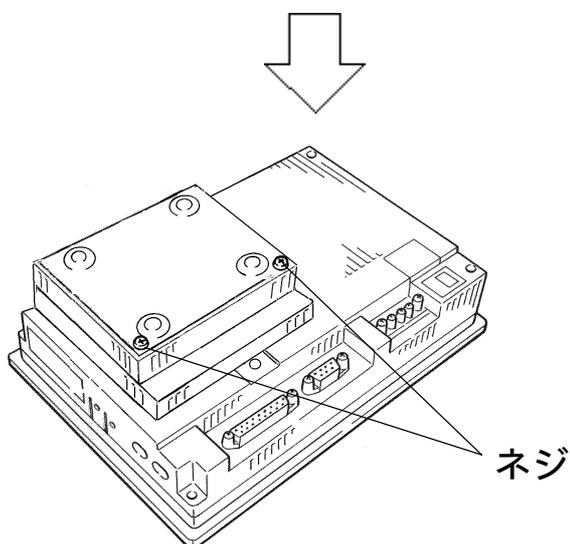


ユニットのツメの部分で PL-3700 本体の穴に引っかけて、コネクタピンが曲がらないように押し込みます。

重要・ 誤った接続で電源を投入すると、機器の破損の恐れがあります。



リアメンテナンスハッチのツメの部分でユニットの穴に引っかけてフタをします。



ユニットにリアメンテナンスハッチのフタをネジ 2 カ所で止めます。

6 サンプルプログラム

本サンプルプログラムはアプリケーション開発のためのプログラム方法を理解するためのものであり、動作の保証は致しかねます。

```
//*****
// PL-3700 DIO SAMPLE PROGRAM *
// din - dout loop back *
// *
// Copyright (c) 1998 *
// Digital Electronics Corporation *
//*****
#include <stdio.h>
#include <dos.h>
#include <signal.h>
#include <conio.h>
#include <stdlib.h>
#include <process.h>

#define BASE_PORT 0x330 /* base port */
#define INPUT_PORT_1 BASE_PORT /* input port1 */
#define OUTPUT_PORT_1 BASE_PORT+2 /* output port1*/
#define INT_FLAG_PORT BASE_PORT+4 /* interrupt flag port */
#define INT_PORT_CLR 0x00 /* interrupt port clear */
#define INT_MASK_PORT BASE_PORT+6 /* interrupt mask port */
#define INT_FUNC_OFF 0x3 /* interrupt function off */

#define MASTER_PIC_CMD_PORT 0x020 /* master interrupt controller command */
#define SLAVE_PIC_CMD_PORT 0x0a0 /* slave interrupt controller command*/
#define MASTER_PIC_MSK_PORT 0x021 /* master interrupt controller mask */
#define SLAVE_PIC_MSK_PORT 0x0a1 /* slave interrupt controller mask */
#define INT_NO 0x77 /* interrupt no. */
#define PIC_EOI_CMD 0x20 /* EOI command */
#define IRQ_15_MASK 0x7f /* PCI IRQ 15 mask */
#define INT_STATUS_ON 1 /* interrupt status on */
#define INT_STATUS_CLR 0 /* interrupt status off */

#define TIMER_BIOS_CALL 0x15 /* system BIOS timer function */
#define WAIT_FLAG_FUNC 0x83 /* set flag when time up */
#define SET_TIMER 0x00 /* set timer */
#define TIMER_VAL 500000L /* 500ms */
#define TIMER_SET_OK 0 /* time set OK status */
#define NOT_TIME_UP 0 /* time up status */

#define DATA_CLR 0 /* clear data to 0 */
#define OK 1 /* set 1 to OK status */
#define ERR 0 /* set 0 to error status */

int int_flag;
long save_vect;
int old_mask;

unsigned char __far *timeup_flag_ptr;
unsigned char __far timeup_flag;
```

```

//*****
// close function
//*****

void close_func(void)
{
    _disable();                /* disable interrupt */
    _outp(INT_MASK_PORT,INT_FUNC_OFF); /* disable DIO board's interrupt */
    _outp(INT_FLAG_PORT,INT_PORT_CLR); /* clear interrupt port */
    _outpw(OUTPUT_PORT_1,DATA_CLR); /* clear dout port */
    _outp(SLAVE_PIC_CMD_PORT,PIC_EOI_CMD); /* issue EOI */
    _outp(MASTER_PIC_CMD_PORT,PIC_EOI_CMD); /* issue EOI */
    _outp(SLAVE_PIC_MSK_PORT,old_mask); /* set original mask */
    _dos_setvect((unsigned) INT_NO,(void far*)save_vect); /* set original vect */
    _enable();                /* enable interrupt */
}

//*****
// Ctrl-C stop handler
//*****

void ctrl_c_handler(int sig) /* ctrl-c stop */
{
    signal(SIGINT,SIG_IGN);
    printf("CTRL-C is pressed to stop\r\n");
    close_func();
    exit(0);
}

//*****
// interrupt handler
//*****

void interrupt far int_func(void)
{
    _outp(INT_FLAG_PORT,INT_PORT_CLR); /* clear interrupt port */
    _outp(SLAVE_PIC_CMD_PORT,PIC_EOI_CMD); /* issue EOI */
    _outp(MASTER_PIC_CMD_PORT,PIC_EOI_CMD); /* issue EOI */
    int_flag = INT_STATUS_ON; /* set interrupt flag to 1 */
}

//*****
// time delay
//*****
int time_wait(long time_val)
{
    short time_val_cx,time_val_dx;
    union _REGS inregs,outregs;
    struct _SREGS segregs;

    time_val_cx = time_val >> 16; /* set time value's high to cx */
    time_val_dx = time_val & 0xffff; /* set time value's low to dx */
}

```

```

    inregs.h.ah = WAIT_FLAG_FUNC;          /* set flag when time up */
    inregs.h.al = SET_TIMER;              /* set timer function */
    inregs.x.bx = _FP_OFF(timeup_flag_ptr); /* time up flag offset */
    inregs.x.cx = time_val_cx;           /* set timer value */
    inregs.x.dx = time_val_dx;           /* set timer value */
    segregs.es = _FP_SEG(timeup_flag_ptr); /* time up flag segment */
    _int86x( TIMER_BIOS_CALL, &inregs, &outregs, &segregs );
                                          /* call timer BIOS */
    if(outregs.x.cflag == TIMER_SET_OK)   /* check status */
        return(OK);
    else return(ERR);
}

//*****
// initialize
//*****

void init(void)
{
    timeup_flag = DATA_CLR;             /* clear time up flag */
    timeup_flag_ptr = &timeup_flag;     /* set pointer to time up flag */

    _disable();                          /* disable interrupt */

    _outp(INT_MASK_PORT, INT_FUNC_OFF);  /* disable DIO board's interrupt */
    _outpw(OUTPUT_PORT_1, DATA_CLR);    /* clear dout port */
    save_vect = (long)_dos_getvect((unsigned) INT_NO);
                                          /* save current vect */
    _dos_setvect((unsigned) INT_NO, int_func);
                                          /* set new interrupt vect */

    old_mask = inp(SLAVE_PIC_MSK_PORT);   /* save old mask */
    _outp(SLAVE_PIC_MSK_PORT, old_mask & IRQ_15_MASK);
                                          /* set mask */
    _outp(SLAVE_PIC_CMD_PORT, PIC_EOI_CMD); /* issue EOI */
    _outp(MASTER_PIC_CMD_PORT, PIC_EOI_CMD); /* issue EOI */
    _outp(INT_FLAG_PORT, INT_PORT_CLR);  /* clear interrupt port */
    _outp(INT_MASK_PORT, INT_PORT_CLR);  /* enable DIO board's interrupt */

    _enable();                            /* enable interrupt */
}

//*****
// dio test
//*****

void dio_test(void)
{
    unsigned short out_data;
        out_data = 1;
    while(1)
    {
        printf("press any key\r\n");
    }
}

```

```

    getch();                                /* wait for keyin */

    if(time_wait(TIMER_VAL) != OK)          /* check time set */
    printf("time delay set error\n");

    _outpw(OUTPUT_PORT_1,out_data);        /* compare dout data */
    if(out_data != _inpw(OUTPUT_PORT_1))
        printf("dout data error\n");

    while((timeup_flag == ERR)&&(!int_flag)); /* wait time up or interrupt */

    if(int_flag == INT_STATUS_ON)          /* show message when interrupt */
    {
        while(timeup_flag == NOT_TIME_UP);
        printf("interrupt occurred\n");
    }

    if(out_data != _inpw(INPUT_PORT_1))
        printf("din data error\n");        /* compare din data */

    out_data = out_data << 1;              /* change to next data */
    if(out_data == 0x0000)
        out_data = 0x0001;

    timeup_flag = NOT_TIME_UP;
    int_flag = INT_STATUS_CLR;             /* interrupt flag clear */
}

//*****
// main
//*****
void main(void)
{
    printf("press CTRL-C to stop program\n");
    if(signal(SIGINT,ctrl_c_handler) == SIG_ERR)
        abort();                           /* set CTRL-C handler */

    init();                                 /* initialize hardware setting */

    dio_test();                             /* do test */

    close_func();                           /* restore hardware setting */
}

```

株式会社 デジタル
 〒 559-0031
 大阪市住之江区南港東8-2-52
 T E L (06)6613-1101 (代)
 F A X (06)6613-5888