PL-UDD00
Operation Instructions

The PL-UDD00 is a photo-isolation type DIO unit, utilizing 16 Input/Output points. This unit is designed for use with Digital's Panel Computer, the PL 3700, hereafter referred to as the PL. Two (2) of this unit's 16 input points can be used for interrupt. The PL-UDD00 connects to the PL's expansion bus.

Corresponding Product: PL-3700 Series Units

Warning - Safety Precautions

- Be sure to check that the PL unit's power is disconnected before installing the PL-UDD00 in the PL in order to prevent electrical shock.
- Do not attempt to modify or open the PL-UDD00, due to the danger of shock or fire.

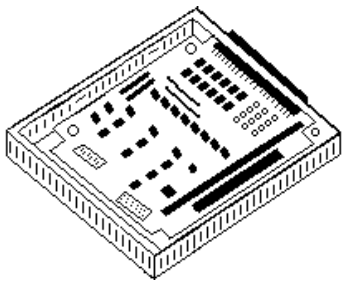
To Prevent Accidents

- Since the PL-UDD00 is a precision instrument, be sure it is neither hit by nor pressed strongly against another object.
- Be sure water, liquids or metal particles are not allowed to enter the unit. Any of these may cause either a breakdown or an electrical shock.
- Do not place or store this unit in a location where there is direct sunlight, excessive heat, dust or vibration.
- Do not store or operate this unit near chemicals, or where there are chemical fumes.
- To prevent damage to file data, be sure to shut down the unit's OS before turning OFF the main power.

Package Contents

Please check that the following items are all included in your package.

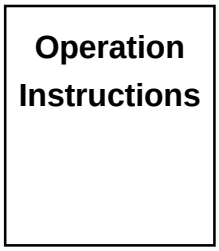
■ **PL-UDD00** **(1)**



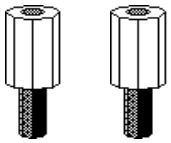
■ **Operation Instructions**

(PL-UDD00)

(English and Japanese)



■ **Spacers** **(2)**



Digital has taken the utmost care to insure the quality of this product when it was shipped, however, should, for any reason, problems or damage have occurred during shipping, please contact your Digital representative immediately for service.

1 Hardware Specifications

■ Functional Specifications

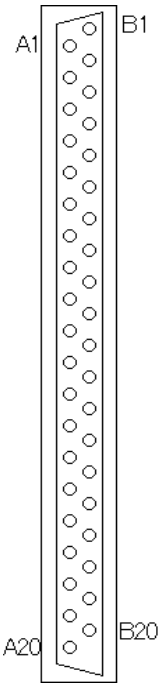
No. of Input Points	16 (8 pins use 1 common)
Input Voltage	DC +12V to +24V
ON/OFF Voltage	ON: +10V (min.), OFF: +8V(max.)
Reverse Voltage Protection	Used
Overvoltage Protection	None
Insulation Method	Photocoupler (NEC PS2801)
Input Filter	MC 14490 Chattering Removal Circuit
Input Interrupt	2 points (IN0, IN8) Arise, Arise Edge Switch is possible Select from IRQ9, 10, 11, 15 (default)
No. of Output Points	16 (1 in 8 is common)
Standard Circuit Voltage	DC +5V / +24V
Standard Output Current (when ON)	200mA (max.), Simultaneous use of 8 points
At power ON	5 μ sec (max.), 10 μ sec (max.), I = 20mA
At power OFF	0.5msec (ave.), 1msec (max.)

■ Environmental Specifications

Ambient Temperature	0 to 40°C
Storage Temperature	-10 to 60°C
Noise Resistance (via noise emulator)	Noise Voltage: 1000Vp-p
	Pulse Width: 50ns, 500ns, 1 μ s
Static Electricity Resistance	4KV

■ Interface Specifications

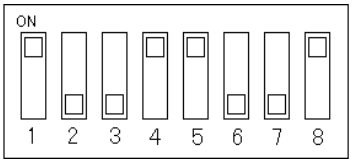
Connector Type FCN-365P040-AU(Male)
Cable Connector FCN-361J040-AU(Female - soldered)
FCN-360C040-D(Cover)
FCN-363J040(Female - crimp contact)
(All items made by Fujitsu Corp.)



Pin No.	Signal Name	Pin No.	Signal Name
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 Hardware Settings

■ I/O Base Address Settings



DIP Switch SW1

(ON:0 / OFF: 1)

Switch No.	Address	Default Settings		
1	A10	ON	330h	3
2	A9	OFF		
3	A8	OFF		
4	A7	ON		3
5	A6	ON		
6	A5	OFF		
7	A4	OFF		
8	A3	ON		0

■ Input Chattering Removal Constant Value Circuit **<Short Plug J1>**

Pin No.	Data
1-2	Removes for up to 400 μsec (default)
2-3	Removes for up to 4msec

■ Interrupt Level **<Short Plug J2>**

Pin No.	Data
1-2	IRQ15 (default)
3-4	IRQ11
5-6	IRQ10
7-8	IRQ9

■ Interrupt Edge **<Short Plug J3>**

Pin No.	Setting	Input	Input Circuit	Input Port
1-2	Short	IN8	L → H	1 → 0
	Open	IN8	H → L	0 → 1
3-4	Short	IN0	L → H	1 → 0
	Open	IN0	H → L	0 → 1

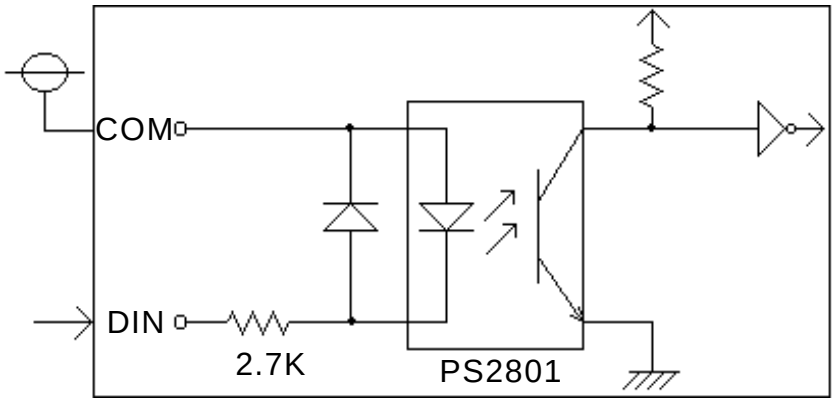
3 I/O Addresses

■ I/O Address Offset List

I/O Address	Name	Data
Base +0 D0 to D7	Input LO IN0 to IN7	(READ) Input Port - LO byte
Base +1 D0 to D7	Input HI IN8 to IN15	(READ) Input Port - HI byte
Base +2 D0 to D7	Output LO OUT0 to OUT7	(READ/WRITE) Output Port LO byte
Base +3 D0 to D7	Output HI OUT8 to OUT15	(READ/WRITE) Output Port HI byte
Base +4 D0 D1 D2 to D7	Interrupt Flag IN0 IN8 N/A	(READ) Interrupt Flag 0 = Interrupt not used 1 = Interrupt used
Base +4 D0 to D7	Interrupt Clear N/A	(WRITE) Interrupt Clear Writing to this port effectively releases the Interrupt state.
Base +6 D0 D1 D2 to D7	Interrupt Mask IN0 IN8 N/A	(READ/WRITE) Interrupt Mask 0 = Interrupt is possible 1 = Interrupt is not possible

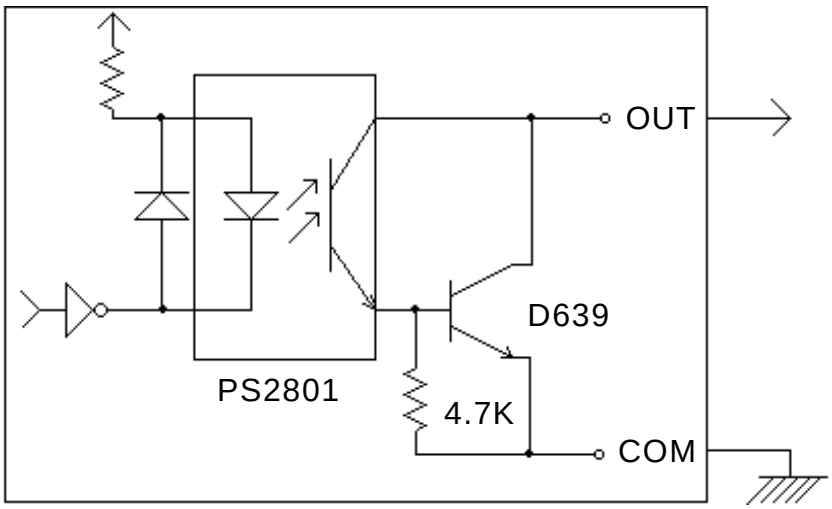
4 Input/Output Circuits

■ PL-UDD00's Internal Input Circuit



* During LOW, IN0 to IN15 are "1".

■ PL-UDD00's Internal Output Circuit



* During "1", OUT0 to OUT15 will output LOW.

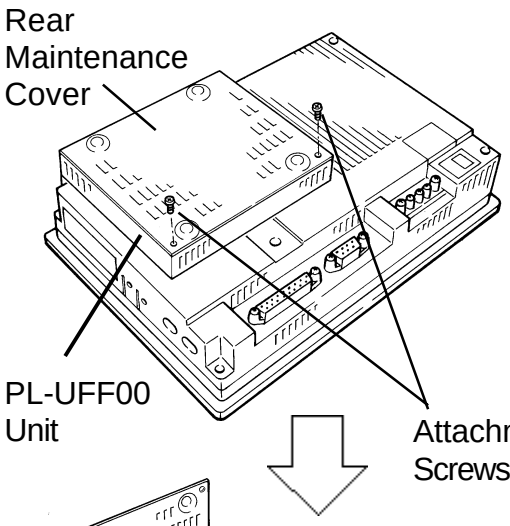
5 Installation

Use the following steps to install the PL-UDD00 in the PL.

Note: • **Normally, up to 3 option units can be attached, one on top of the other, to the PL. For development and debugging purposes, however, up to 4 units can be used.**

WARNING

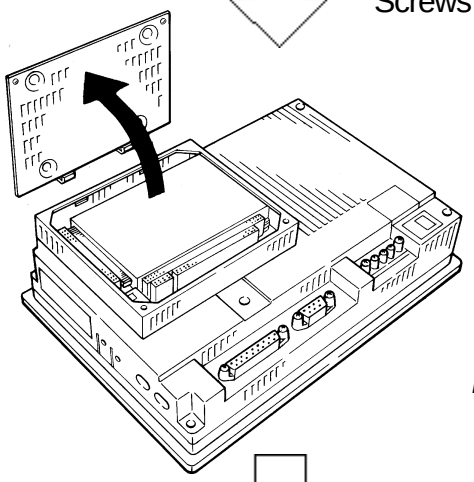
WARNING: Shock Danger! Be sure to unplug the PL unit prior to installing the PL-UDD00



1) Unscrew the PL's Rear Maintenance Cover Attachment Screws.

Important

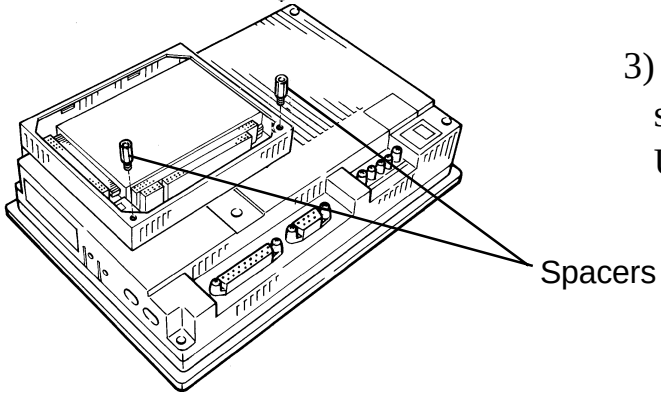
- ***Be sure to first attach the PL-UFF00 unit to the PL-3700. This unit should always be attached as the first level option unit. The PL-UDD00 then attaches on top of the PL-UFF00.***



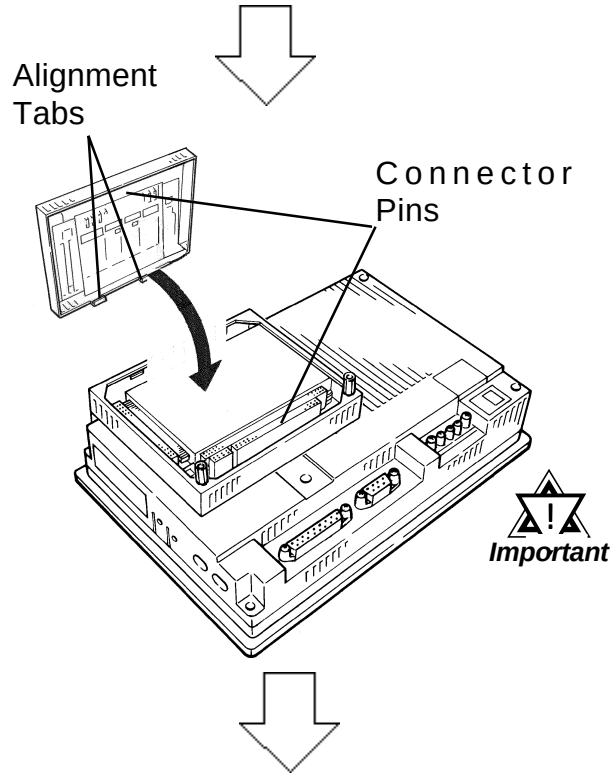
2) Lift up the cover from the bottom edge, as shown by the arrow, and remove it.

Important

- ***Be careful not to bend the cover's alignment tabs.***



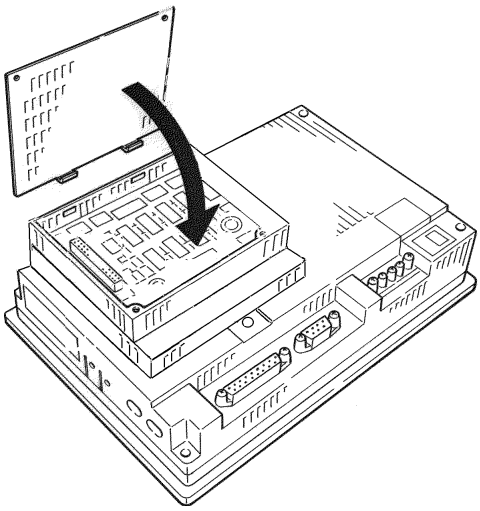
3) Insert (screw in) the two (2) spacers included in the PL-UDD00's package.



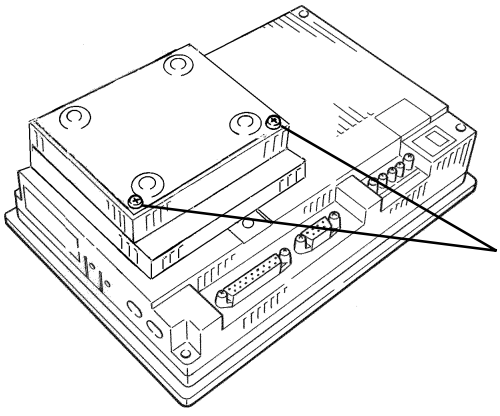
4) Using the PL-UDD00's alignment tabs as guides, insert the unit into the PL. Be sure not to bend the unit's connector pins.

Important

- ***If the pins are not aligned correctly, when the PL's power is turned ON, the PL and the unit may be damaged.***



5) Using the Rear Maintenance Cover's alignment tabs as guides, attach the cover to the back of the PL-UDD00.



6) Use the cover's Attachment Screws to secure the cover in place. (This will secure the PL-UDD00 in position)

5 Sample Program

The following sample program is intended for application development information purposes. Please refer to it when creating programs for the PL, and use it to check the operation of various features.

```
//**************************************
//   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 oringinal 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\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\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))
                                                 /* compare din data */
                printf("din data error\n");

        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)
                                                 /* set CTRL-C handler */
        abort();

    init();                                      /* initialize hardware setting */

    dio_test();                                  /* do test */

    close_func();                                /* restore hardware setting */

}
```