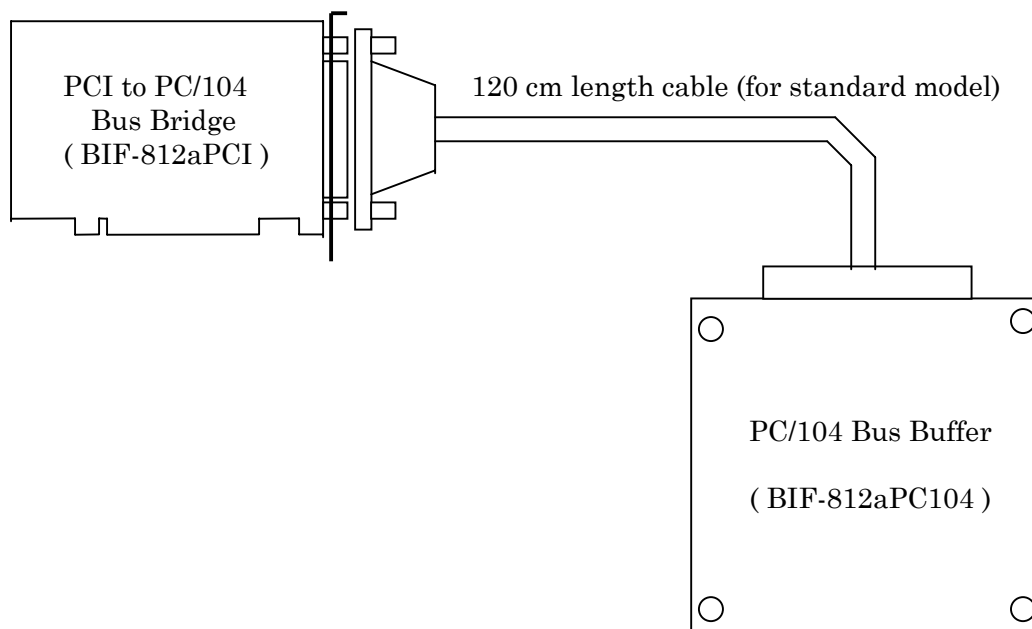


Real Solution for FA & LA



PCI to PC/104 Bus Bridge Kit
(Bus Interface Boards + Cable) **BIF-812aK**

User's Manual

MICRO SCIENCE., Co Ltd

2-37-12, Nishiogikita, Suginami-ku,
Tokyo, 167-0042, JAPAN

Phone: +81-3-3396-8362 (Japanese only)

Fax: +81-3-3301-5593 (also English)

Aug 21, 2002

Table of Contents

| | |
|---|----|
| Caution | 3 |
| Legal Notice | 3 |
| Software License Agreement | 4 |
| Customer Support Policy | 5 |
| Price List and Ordering Information | 6 |
| | |
| Section 1. Introduction | |
| 1-1. Guide this Manual | 7 |
| 1-2. Functional Specification | 7 |
| 1-3. Functional Description | 9 |
| 1-4. Profile of them | 11 |
| 1-5. Settings on the Board | 13 |
| 1-6. Connector Pin Assignment | 14 |
| 1-7. Installing BIF-812PCI | 16 |
| | |
| Section 2. General Programming | |
| 2-1. General Programming information | 19 |
| 2-2. I/O Register Memory Map | 20 |
| 2-3. Reset, get ID and Number | 21 |
| 2-4. Interrupt Control | 22 |
| | |
| Section 3. General Purpose Software | |
| 3-1. Installing the Software | 25 |
| 3-2. Getting the Resource information | 26 |
| | |
| Section 4. General Purpose Function Library | |
| 4-1. Specification | 29 |
| 4-2. Setting the System | 29 |
| 4-3. Cording the applications | 30 |
| 4-4. Functions | 31 |
| | |
| Section 5. Maintenance and Appendix | |
| 5-1. Trouble Shootings | 37 |
| Q & A form | 38 |

Caution

Unpacking

This package contain a BIF-812aPCI board, BIF-812aPC104 board, Cable, and a User's Manual, and 4 pieces of 15mm standoff. Upon receipt the package, visually inspect the board for missing or damaged materials. This product was shipped in perfect condition as it was new. Examine the package for physical damage. In the event of damage, save all packing materials and notify your courier to validate shipping claims.

Anti-static discharge

Both board contains components that are susceptible to static discharge, and should be handled with appropriate caution. The anti-static packing material protects components from being damaged by static discharge. Should both board need to be returned for repair at a later date, it can be safely done by packing it in the original materials.

Warranty

MICRO SCIENCE warrants that this product was manufactured free of defect in materials or workmanship under normal use and service as described in this User's Manual. Obligations under this warranty are limited to replacing or repairing at MICRO SCIENCE's option. Any said of products, at MICRO SCIENCE's factory or facility, should have to be prepaid transportation charges, and which are after examination disclosed to the satisfaction of MICRO SCIENCE to be thus defective, for a period within one year shipment. These provisions do not extend the original warranty period of any product which has either been repaired or replaced by MICRO SCIENCE. This warranty does not contain a guarantee, either expressed or implied, of merchantability or fitness for particular purpose. This warranty shall not apply to any such products which have been repaired or altered except by MICRO SCIENCE or which have been subject to misuse, negligence, or accident.

MICRO SCIENCE assumes no liability for damages or loss consequent to use of this product. This product is not designed for a level of reliability for use in life support or critical applications. It is customer's sole responsibility to determine if this product is suitable for the application.

Disclaimer

The information contained in this document has been carefully examined and is believed to be entirely accurate. However, MICRO SCIENCE assumes no responsibility for errors or omissions. MICRO SCIENCE reserves the right to make changes to this manual without prior notification in accordance with the purpose of product support and or improvement.

Proprietary Notice

This entire document is not to be photocopied, transcribed, recorded, communicated, or reproduced by any means without permission of MICRO SCIENCE. The foregoing does not apply to vendor proprietary rights or rights under the patents of third parties. Use of this product does not convey a license either expressed or implied.

Applicable Laws

Any claim relating to MICRO SCIENCE's products shall be governed by the internal laws of Japan.

Copyright 2002 by
MICRO SCIENCE, Co. LTD

Software License Agreement

This Agreement constitutes the license between MICRO SCIENCE, Co and the purchaser of MICRO SCIENCE products.

Definitions

In this Agreement, a "FILE" shall mean a contiguous collection of machine-readable symbols, bytes, characters, or codes which may be used by the CPU on the user's computer or processing equipment.

A "PROGRAM" is a file or related group of files which may be loaded and processed on the user's computer or processing equipment to perform the functions.

A "SOFTWARE" shall mean one or more FILES or PROGRAMS.

Usage

The files that supplied to MICRO SCIENCE customers are solely for the purpose of illustrating how MICRO SCIENCE products operate or to allow modifications of this software by the customer for a particular application. In the case of binary executable only files, the user may not disassemble, reverse compile, reverse engineer or otherwise remove the proprietary status of these programs.

The programs and files on MICRO SCIENCE-supplied media are copyrighted by and the proprietary of MICRO SCIENCE, Co. All supplied programs and files except for executable-only files may be reproduced, transferred, copied, printed out, displayed, modified, or used in the customer's application as long as they control MICRO SCIENCE products

Binary executable –only files of customer's application which used MICRO SCIENCE software may be distributed to anyone without restriction.

All supplied source files and the source files of customer's application which used MICRO SCIENCE software may not be disclosed, copied, reproduced, or transferred to third parties without written permission.

Disclaimer

Although the MICRO SCIENCE software have been carefully checked and executed, errors are possible. However, MICRO SCIENCE assumes no responsibility for damages or loss arising from loading and running the software which are originally supplied, or has been modified for a particular purpose.

This software is not designed for a level of reliability for use life support, or critical applications.

It is customer's sole responsibility to determine if this software is suitable for the application.

MICRO SCIENCE retains the right to modify this software without prior notification in accordance with the purpose of product support and improvement.

Customer Product Support Policy

MICRO SCIENCE will answer the written questions including FAX, or Email in Japanese or English from the registered user about this product.
Send us the question form in this manual filled with the information.

We do not answer on phone with any language but Japanese.
Although MICRO SCIENCE may offer advice, we will not design the user's application.

Price List (# Aug, 2002)

| Item | Unit Price | Description |
|-----------|------------|---|
| BIF-812aK | \$ 360 US | PCI to PC/104 Bus Bridge Interface Kit |
| VSM-520SW | \$ 24 US | AC-adaptor (AC100v to 240v input DC5v/2A output) |

The product consists of a BIF-812aPCI board, a BIF-812aPC104 board, Cable, 4 pieces of standoff, printed User's Manual, and the sample software disk.

Special Discount

At repeat or volume order, special discount is available without the printed User's Manual and software disk.

Please add (E) next to follow the product name as BIF-812aK(E), and the unit price is reduced \$10.00 for each.

The User's Manual in PDF format is also available from MICRO SCIENCE's web site www.microscience.co.jp/eng/.

Section 1. Introduction

1-1. Guide this Manual

This Manual contains a complete set of hardware and programming information for the BIF-812aK, including configuration, installation, and I/O connection.

Section 1 contains the outline of functional descriptions and detail specifications, the installation, and setup procedure for the board.

Section 2 contains the programming information and the register specifications. Section 3 contains the software information. Section 4 contains the specification and description for I/O library and driver. Section 5 contains the trouble-shootings, and repair. The last page is the request form for the Q and A.

1-2. Functional Specification

BIF-812aK consist of BIF-812aPCI, BIF-812aPC104, and DX100D-120.

PCI interface board (BIF-812aPCI)

| | |
|-------------------------|---|
| Bus bridge function | 256 byte address I/O access from PCI to PC/104 bus (Lower 16 byte must be reserved for own board control) |
| PC/104 bus drive | Sink current = 48mA (at Vol = 0.5v) Source current = -15mA (at Voh = 2.0v) Input current = -0.1mA(driven by low state) = 0.02mA(driven by High state) |
| Interrupt to PCI bus | 1 level (OR-function with multiple interrupt form PC/104 bus) |
| I/O library and driver | for WINDOWS 2000, XP, 98, ME (without Interrupt function because of WDM driver.) This I/O library has the messaging function that send the message when detect the change of programmed input bits of the port. It may be instead of the interrupt. |
| Physical, Environmental | Dimensions : 174.4 x 98.4 mm (Short size of PCI board) Operating Temperature Range 0 to +45 C Storage Temperature Range -10 to +85 C Relative Humidity 80% (Non-condensing) Power Supply, Consumption +5v 0.5 A (from PC) |

PC/104 interface board (BIF-812aPC104)

| | |
|-------------------------|---|
| Basic Function | PC/104 bus buffer |
| PC/104 bus drive | Sink current = 48mA (at Vol = 0.5v) Source current = -15mA (at Voh = 2.0v) Input current = -0.1mA(driven by low state) = 0.02mA(driven by High state) |
| Physical, Environmental | Dimensions : 141 x 137 mm Operating Temperature Range 0 to +55 C Storage Temperature Range -10 to +85 C Relative Humidity 80% (Non-condensing) Power Supply, Consumption +5v 0.6 A (to be supplied) |

Cable (DX100D-120)

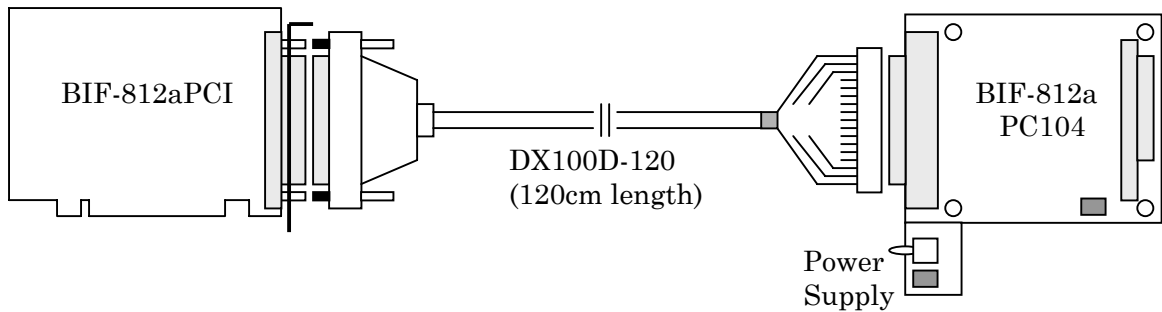
| | |
|----------------------------------|--|
| Interconnection | between BIF-812aPCI board and BIF-812aPC104 board. |
| Connectors / made by HIROSE / | Plug : DX40-100P-(03) Shell : DX100-CVI |
| Length | 120cm for standard model. |

1-3. Functional Description

BIF-a812K is designed for multiple I/O boards access on PC/104 bus from PCI bus. It consists of a “BIF-812aPCI” board as a PCI interface, a “BIF-812aPC104” board as a PC/104 bus buffer, and a Cable for inter-connection between the boards. BIF-812aPCI installed in the IBM compatible WINDOWS PC must be get 256 byte I/O resources by the plug-and-play operation, and generate PC/104 bus compatible signals to interface “BIF-812aPC104”. General purpose I/O library and WDM-type device driver for Windows2000/XP/98/ME is also available come with the product.

BIF-812aPC104 inter-connected with a BIF-812aPCI is a PC/104 bus buffer and the back plane to access the multiple PC/104 I/O boards stacking up with. PC/104 bus drive-ability that is specified with the sink current and source current is up to 30 TTL current source load. See section 1-2 for details. This PCI to PC/104 bus-bridge system can access to pp00 to ppFFH of PC/104 bus I/O address. Where pp is the specified value with the on-board switch SW1 and SW2 of BIF-812aPC104. Note, pp00 to pp0FH must be reserved for BIF-812aK system control including interrupt sub-system, and this system dose not support Memory access, Bus master, and DMA function on PC/104 bus.

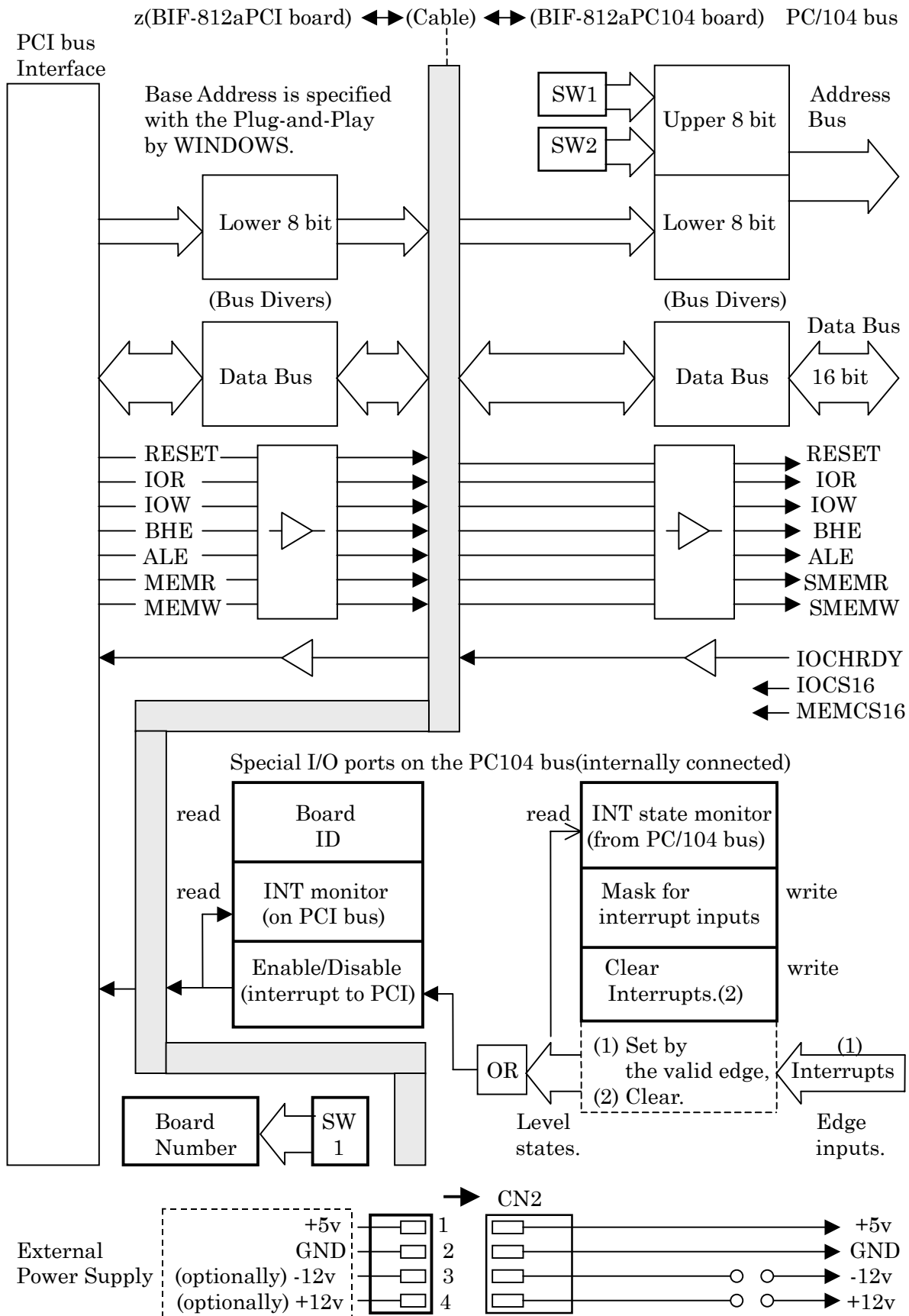
Figure 1-3A. System Configuration



consist of the PCI to PC/104 Bus bridge system KIT “BIF-812aK”

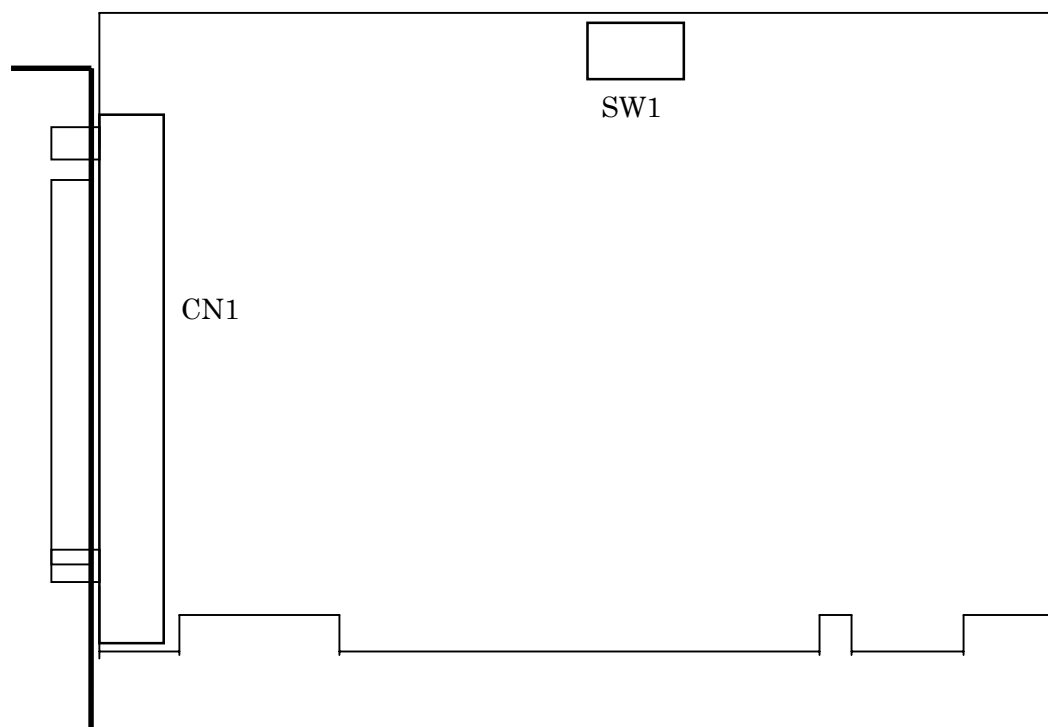
| BIF-812aPCI | DX100D-120 | BIF-812aPC/104 |
|---|---|--|
| PCI board for IBM compatible WINDOWS PC. General purpose I/O access library for PC/104 bus for WINDOWS2000/XP/98/ME is available come with the KIT. | 120cm length sealed cable for inter connection between BIF-812aPCI and BIF-812aPC104 board. | PC/104 Bus buffer and Back-plane for stacking up inter-connect with PC/104 I/O boards. |
| | Expanding to xxx cm is available in option. | I/O address between pp10 to ppFFH is available for any PC/104 boards. |

Figure 1-3B. Functional Block Diagram.



1-4. Profiles of them

Figure 1-4A. Layout of “BIF-812aPCI” board.



- # CN1: Connector for inter connection with “BIF-812aPC104” board.
- # SW1: Switch for board-number .<read-able> / See section 2-3. /

Figure 1-4B “DX100D-120” cable.

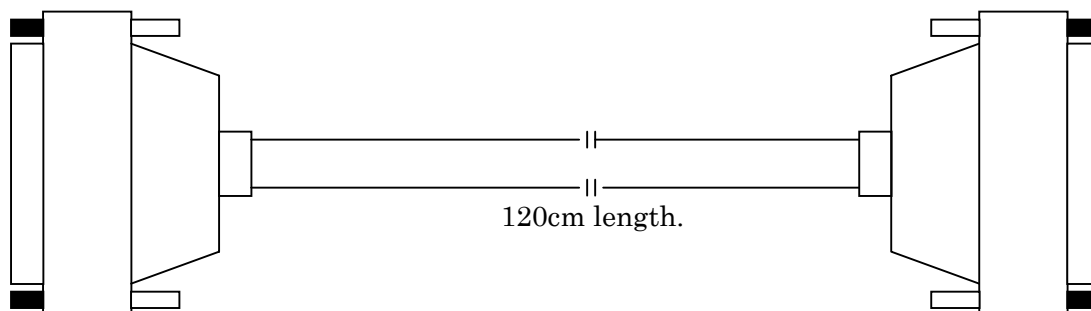
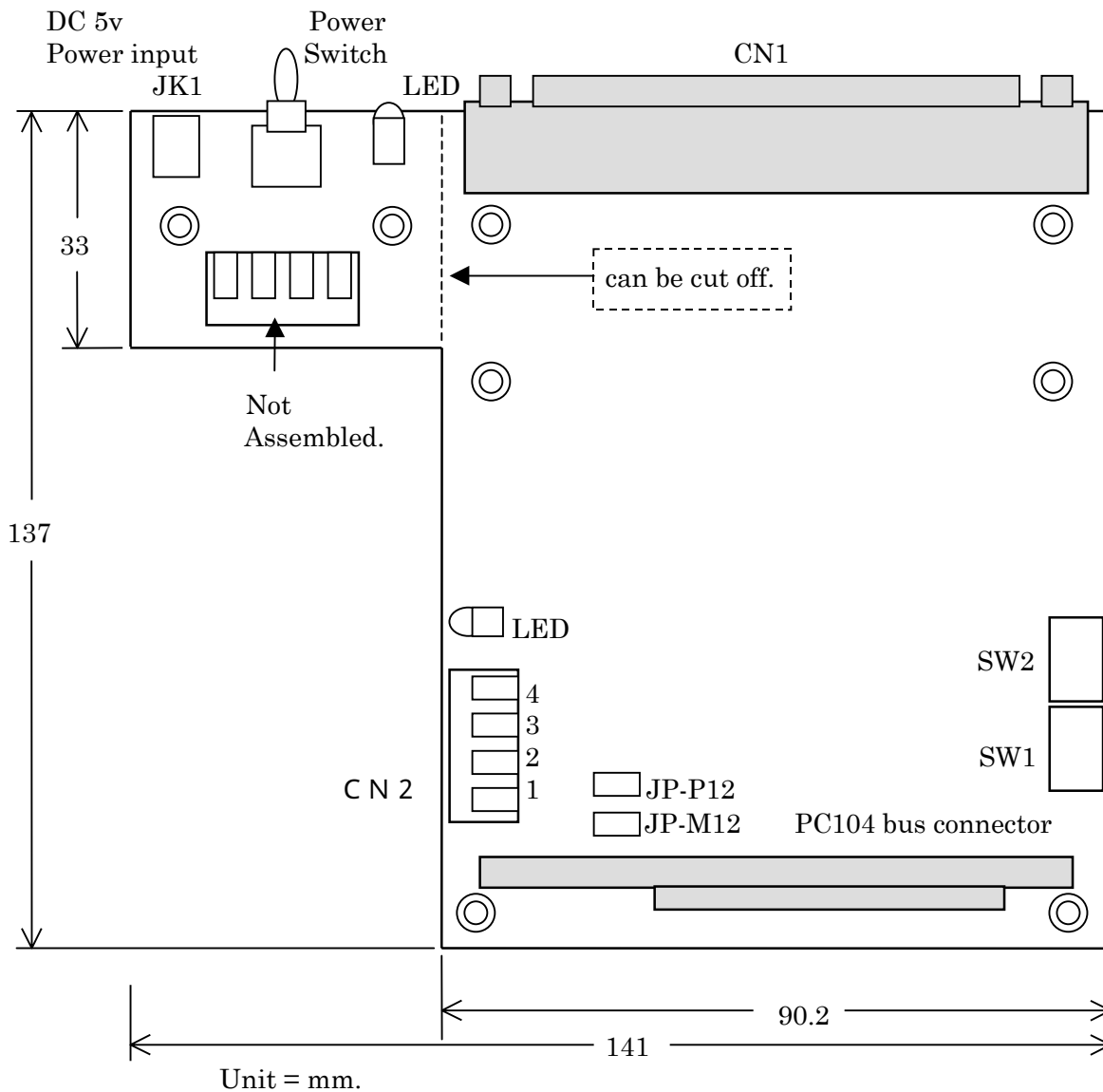


Figure 1-4C. Layout of “BIF-812aPC104” board.



CN1: Connector for inter-connection with “BIF-812aPCT” board.

SW1, SW2 : Switch for specify the upper byte I/O address of PC/104 bus.

#LED: Indicator for +5v power supply.

<Power Supply> DC 5v power supply is acceptable either “JK1” or “CN2”.

JK1: Receptacle for DC5v(max.2A) Power Supply. /* center “+” */
Optional AC adapter “VSM-520SW” is suitable with this way.

#CN2: (sub) Power receptacle.

DC power supply is acceptable
by “CN2” instead of “JK1”

(Pin assignment) Pin4: +5v

Pin3: GND

Pin2: -12v (option)

Pin1: +12v (option)

#JP-P12: Jumper for connect the “Pin-4” of CN2 to +12v line of PC/104 bus.

#JP-M12: Jumper for connect the “Pin-3” of CN2 to -12v line of PC/104 bus.

1-5. Settings on the board

1-5-1. Board Number of “BIF-812aPCI”

Set SW1 of “BIF-812aPCI” board among 0H to FH as a board number.

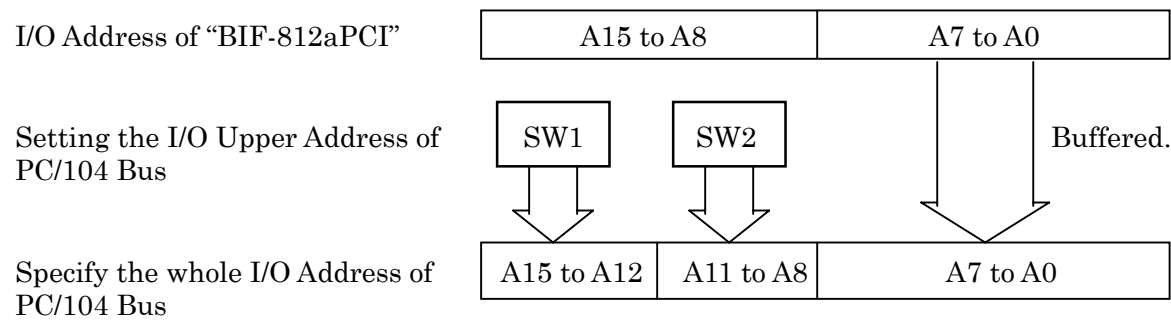
It is useful to identify from a party of few “BIF-812aPCI” boards.

1-5-2. I/O Base Address of PC/104 Bus

In the “BIF-812aK” system, any I/O board installed in PC/104 bus appears as a party of registers within the 256-byte block of PCI bus.
The upper byte address of PC/104 bus is fixed by the switches of SW1 and SW2 on the “BIF-812aPC104” board as like “pp”.
Other hand,
I/O Address of “BIF-812aPCI” shall be set a block from xx00H to xxFFH of PCI bus by Pug-and-Play operation.

Therefore I/O access of xx00H to xxFFH on the PCI bus cause I/O access of pp00H to ppFFH on the PC/104 bus.
Note, any I/O address of the board on the PC/104 bus in the “BIF-812aK” system must be located within pp10H to ppFFH, because pp00H to pp0FH must be reserved for the special function registers as described in section 2-2 to 2-5.

Figure 1-5. I/O address configuration.



1-6. Connector Pin Assignment

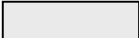
Table 1-6A. Inter-connection cable Connector (CN1) pin assignment

| IN/OUT | Function | sign | pin number | | sign | /Function/ |
|--------|------------------------|-------|------------|-----|------|--------------|
| IN/OUT | DATA Bus | D0 | 1 | 0 0 | 51 | GND /common/ |
| IN/OUT | DATA Bus | D1 | 2 | 0 0 | 52 | GND /common/ |
| IN/OUT | DATA Bus | D2 | 3 | 0 0 | 53 | GND /common/ |
| IN/OUT | DATA Bus | D3 | 4 | 0 0 | 54 | GND /common/ |
| IN/OUT | DATA Bus | D4 | 5 | 0 0 | 55 | GND /common/ |
| IN/OUT | DATA Bus | D5 | 6 | 0 0 | 56 | GND /common/ |
| IN/OUT | DATA Bus | D6 | 7 | 0 0 | 57 | GND /common/ |
| IN/OUT | DATA Bus | D7 | 8 | 0 0 | 58 | GND /common/ |
| IN/OUT | DATA Bus | D8 | 9 | 0 0 | 59 | GND /common/ |
| IN/OUT | DATA Bus | D9 | 10 | 0 0 | 60 | GND /common/ |
| IN/OUT | DATA Bus | D10 | 11 | 0 0 | 61 | GND /common/ |
| IN/OUT | DATA Bus | D11 | 12 | 0 0 | 62 | GND /common/ |
| IN/OUT | DATA Bus | D12 | 13 | 0 0 | 63 | GND /common/ |
| IN/OUT | DATA Bus | D13 | 14 | 0 0 | 64 | GND /common/ |
| IN/OUT | DATA Bus | D14 | 15 | 0 0 | 65 | GND /common/ |
| IN/OUT | DATA Bus | D15 | 16 | 0 0 | 66 | GND /common/ |
| OUT | Address Bus | A0 | 17 | 0 0 | 67 | GND /common/ |
| OUT | Address Bus | A1 | 18 | 0 0 | 68 | GND /common/ |
| OUT | Address Bus | A2 | 19 | 0 0 | 69 | GND /common/ |
| OUT | Address Bus | A3 | 20 | 0 0 | 70 | GND /common/ |
| OUT | Address Bus | A4 | 21 | 0 0 | 71 | GND /common/ |
| OUT | Address Bus | A5 | 22 | 0 0 | 72 | GND /common/ |
| OUT | Address Bus | A6 | 23 | 0 0 | 73 | GND /common/ |
| OUT | Address Bus | A7 | 24 | 0 0 | 74 | GND /common/ |
| OUT | Address Bus | A8 | 25 | 0 0 | 75 | GND /common/ |
| OUT | Address Bus | A9 | 26 | 0 0 | 76 | GND /common/ |
| OUT | Address Bus | A10 | 27 | 0 0 | 77 | GND /common/ |
| OUT | Address Bus | A11 | 28 | 0 0 | 78 | GND /common/ |
| OUT | Address Bus | A12 | 29 | 0 0 | 79 | GND /common/ |
| OUT | Address Bus | A13 | 30 | 0 0 | 80 | GND /common/ |
| OUT | Address Bus | A14 | 31 | 0 0 | 81 | GND /common/ |
| OUT | Address Bus | A15 | 32 | 0 0 | 82 | GND /common/ |
| OUT | Direction. | DIR | 33 | 0 0 | 83 | GND /common/ |
| OUT | Bus Clock | BCLK | 34 | 0 0 | 84 | GND /common/ |
| OUT | (internally activated) | | 35 | 0 0 | 85 | GND /common/ |
| OUT | (internally activated) | | 36 | 0 0 | 86 | GND /common/ |
| IN | I/O Channel Ready | CHRDY | 37 | 0 0 | 87 | GND /common/ |
| IN | (internally activated) | | 38 | 0 0 | 88 | GND /common/ |
| IN | Interrupt | INT | 39 | 0 0 | 89 | GND /common/ |
| IN | (internally activated) | | 40 | 0 0 | 90 | GND /common/ |
| OUT | RESET | RST | 41 | 0 0 | 91 | GND /common/ |
| OUT | (internally activated) | | 42 | 0 0 | 92 | GND /common/ |
| OUT | Byte High Enable | BHE | 43 | 0 0 | 93 | GND /common/ |
| OUT | Address Latch Enable | ALE | 44 | 0 0 | 94 | GND /common/ |
| OUT | I/O Read | IOR | 45 | 0 0 | 95 | GND /common/ |
| OUT | I/O Write | IOW | 46 | 0 0 | 96 | GND /common/ |
| OUT | Memory Read | MEMR | 47 | 0 0 | 97 | GND /common/ |
| OUT | Memory Write | MEMW | 48 | 0 0 | 98 | GND /common/ |
| - - | (Not used) | | 49 | 0 0 | 99 | GND /common/ |
| OUT | (internally activated) | | 50 | 0 0 | 100 | GND /common/ |

Table 1-6B. PC/104 Bus Signal Assignments

| Pin Number | J1/P1 Row-A | J1/P2 Row-B | J2/P1 Row-C | J2/P2 Row-D |
|------------|-------------|-------------|-------------|-------------|
| 0 | -- | -- | GND | GND |
| 1 | IOCHCHK | GND | SBHE* | MEMCS16* |
| 2 | SD7 | RESETDRV | LA23 | IOCS16* |
| 3 | SD6 | +5v | LA22 | IRQ10 |
| 4 | SD5 | IRQ9 | LA21 | IRQ11 |
| 5 | SD4 | -5v | LA20 | IRQ12 |
| 6 | SD3 | DRQ2 | LA19 | IRQ15 |
| 7 | SD2 | -12v | LA18 | IRQ14 |
| 8 | SD1 | ENDXFR* | LA17 | DACK0* |
| 9 | SD0 | +12v | MEMR* | DRQ0 |
| 10 | IOCHRDY | (KEY) :#2 | MEMW* | DACK5* |
| 11 | AEN | SMEMW* | SD8 | DRQ5 |
| 12 | SA19 | SMEMR* | SD9 | DACK6* |
| 13 | SA18 | IOW* | SD10 | DRQ6 |
| 14 | SA17 | IOR* | SD11 | DACK7* |
| 15 | SA16 | DACK3* | SD12 | DRQ7 |
| 16 | SA15 | DRQ3 | SD13 | +5v |
| 17 | SA14 | DACK1* | SD14 | MASTER* |
| 18 | SA13 | DRQ1 | SD15 | GND |
| 19 | SA12 | REFRESH* | (KEY) :#2 | GND |
| 20 | SA11 | SYSCLK | -- | -- |
| 21 | SA10 | IRQ7 | -- | -- |
| 22 | SA9 | IRQ6 | -- | -- |
| 23 | SA8 | IRQ5 | -- | -- |
| 24 | SA7 | IRQ4 | -- | -- |
| 25 | SA6 | IRQ3 | -- | -- |
| 26 | SA5 | DACK2* | -- | -- |
| 27 | SA4 | TC | -- | -- |
| 28 | SA3 | BALE | -- | -- |
| 29 | SA2 | +5v | -- | -- |
| 30 | SA1 | OSC | -- | -- |
| 31 | SA0 | GND | -- | -- |
| 32 | GND | GND | -- | -- |

<NOTES> #1: Rows C and D are not required on 8-bit modules.
#2: Pin number 10 of Row-B and 19 of Row-C are the Key locations.
#3: Signal timing and function are as specified in ISA specification.
#4: Signal source/sink current are up to -15 /48mA for BIF-812aPC104.

 These signals are not available with BIF-812aPC104.

1-7. Installing BIF-812aPCI

BIF-812aPCI must be installed in the PC with plug-and-play process by WINDOWS, before configure and run with the application.

Plug-and-play shall be executed when the first time of PC Start with BIF-812aPCI.

Table 1-7. Information for PCI configuration in BIF-812aPCI on-board- ROM .

| Term | Data | Description |
|---------------------|-------|--|
| Vender ID | 13FDH | For the vender of the PCI interface device. |
| Device ID | 0301H | For the PCI interface device. |
| Subsystem Vender ID | 13FDH | For the vender of the PCI board. (MICRO SCIENCE) |
| Subsystem ID | 0301H | For the PCI board. (BIF-812aPCI) |
| Class Code | 0680H | For the field. |
| Request Resource | | 256 byte of I/O and Memory address block. |

Using Interrupt ?

Interrupt resources are not requested by the default information of the BIF-812aPCI. The basic function library for BIF-812aPCI and BIF-812aPC104 for WINDOWS come with this product does not support the interrupt operation because of using the WDM type driver.

Instead of the real-time interrupt operation, the library provide the messaging function when detect the change state of the specified bit of the specified input-port, and interrupt states of PC/104 bus are assigned to the special input-port.

Where the interrupt states are set by the rising edge of each enabled interrupt signal on the PC/104 bus, and cleared by the write command to the special output-port.

See section 2 for details.

How to get the interrupt resource of PCI.

You can also get the interrupt resource of PCI for the BIF-812aPCI with the follows, then using it with your own drivers.

- (1) Install the BIF-812aPCI with the default condition as described in the next page.
- (2) Research if any interrupt resources are vacancy by “Device Manager” of WINDOWS.
- (3) If the interrupt resource is vacancy, you can rewrite to change the information-ROM of the BIF-812aPCI to request the resources by the utility software “cf9050” come with this product. See section 3-1.
- (4) When the next Plug-and-Play in the power-on process, BIF-812aPCI shall be got a interrupt resource.

<Caution>

Once interrupt resources are requested in case of all the resources are already reserved cause Plug-and-Play shall be incomplete, and the BIF-812aPCI shall not be recognized by WINDOWS.

Board Install procedure for WINDOWS2000/XP/98/ME

- (1) Set BIF-812aPCI board in the PCI slot of the PC.
- (2) Power-on with the switch.
- (3) Plug-and-Play shall be started in the setup process of WINDOWS as a wizard.
- (4) Set the CD-ROM marked 2002/02 or later come with the product, then answer the folder “¥win2K” to the question for where is the driver, or where is “dms_pci.sys” ?
The device driver shall be installed into the appropriate folder (¥WINDOWS¥SYSTEM32¥DRIVERS),
and BIF-812aPCI recognized by WINDOWS.

“dms_pci.sys” is the WDM type device driver for general purpose I/O board made by MICRO SCIENCE.

It works for up to 8 PCI boards in the PC,
BIF-812aPCI is one of them.

The function library “H812_2k.dll” shall be also installed into appropriate folder (¥WINDOWS¥SYSTEM32).

- (5) Refer to the recognition for BIF-812aPCI by “Device Manager” along the WINDOWS menu as follows.
< Device Manager > ----- < MSCIENCE > ---- < BIF-812PCI > --
--- < Properties > ----- < Resources >

Section 2. General Programming

2-1. General Programming Information

Handling

Any PC/104 I/O board stacking up with BIF-812aPC104 appears to the host PCI bus CPU as a block of hardware registers mapped within the I/O address space. These registers control the operation of any PC/104 I/O board as long as they are accessed using 16bit I/O addressing with each data transfers, and include lowest 16 byte special function registers for Reset BIF-812aK system, recognize BIF-812aPC104 board, and interrupt control for PCI bus.

Operation

BIF-812aK works as an expansion system with the PC/104 bus compatible I/O boards for WINDOWS PC.

Section 2 specify the special function ports for control the system including Reset, Interrupt control, and state monitor.

(section 2-2)

Register Memory Map.(on PCI bus)

(section 2-3)

Reset and recognize the BIF-812aK system.

(section 2-4)

Interrupt control to PCI bus including the Edge to State conversion from PC/104 bus, and the state monitor.

2-2. I/O Register Memory Map

Table 2-2 shows the I/O Register Memory Map of BIF-812aK system.

See section 1-5-2 for the detail.

Table 2-2. I/O Register Assignment

| I/O Address | IN/OUT | Description |
|------------------------------------|--------|---|
| <BASE> + FFH to <BASE> + 10H | IN | For any register of any I/O board in this BIF-812aK system. (Allocation area for PC/104 application I/O boards.) |
| | OUT | |
| <BASE> + EH | IN | Reset BIF-812aK control system and get ID |
| | OUT | Reserved. |
| <BASE> + CH | IN | Read BIF-812 system Number. (Read SW1 of BIF-812aPCI) |
| | OUT | Reserved. |
| <BASE> + AH | IN | Reserved. |
| | OUT | Reserved. |
| <BASE> + 8H | IN | Reserved. |
| | OUT | Reserved. |
| <BASE> + 6H | IN | Read back the Control for Interrupt accept from PC/104 bus. |
| | OUT | Control for Interrupt accept from PC/104 bus. |
| <BASE> + 4H | IN | Read Status. (Interrupt request state from PC/104 bus) |
| | OUT | Clear Status. (Interrupt request state from PC/104 bus) |
| <BASE> + 2H | IN | Read Interrupt state of PCI bus. |
| | OUT | Control (Enable/Disable) for Interrupt to PCI bus. |
| <BASE> + 0H | IN | Reserved. |
| | OUT | Reserved. |

<Note-1> (BASE) is the I/O base address of the BIF-812aPCI board on PCI bus that consist of a 256 byte block allocated by Plug-and-Play operation. Other hand, upper address byte on PC/104 bus of BIF-812aPC104 board is programmed by the switches SW1 and SW2 as like “pp”. Therefore, Read or Write between (BASE)+00H to (BASE)+FFH from PCI bus cause Read or Write between pp00H to ppFFH to PC/104 bus.

<Note-2> From (BASE)+00H to (BASE)+0FH must be allocated as the special function Word Registers for this BIF-812aK system.

<Note-3> Any PC/104 application I/O board installed in this BIF-812aK system must be allocated between pp10H to ppFFH with own decoding circuit of the board.

2-3. Reset, Get ID, and Number

```
rst = inpw (BASE+0xE) ; /* Reset BIF-812aK system, and get ID */
```

Read (BASE+EH) word Register cause the BIF-812aK system reset.
The registers on the PC/104 application board shall not be cleared, but all special function registers of the BIF-812aK system must be initialized,

Where “rst” is the ID that depend on the board, “1FH” for BIF-812aPC104.

Table 2-3A. Read (BASE+EH) word Register Bit-Field.

| Bit | Description |
|-----------------|------------------------------------|
| B15 to B8 | Not-used. |
| B7 to B0 | “1FH” is the ID for BIF-812aPC104. |

```
brd = inpw (BASE+0xC) ; /* Read BIF-812aK system Number */
```

Read (BASE+CH) word Register provide the value of the “SW1” of the BIF-812aPCI board that is useful for identify the particular BIF-812aK system.

The function library for WINDOWS come with the product does not care this value, because that support only one BIF-812aK system.

Table 2-3B. Read (BASE+CH) word Register Bit Field.

| Bit | Description |
|------------------|---|
| B15 to B12 | Not-used. |
| B11 to B8 | Value of the “SW”1 of the BIF-812aPCI. (as a BIF-812aK system Number) |
| B7 to B0 | Not-used. |

2-4. Interrupt Control

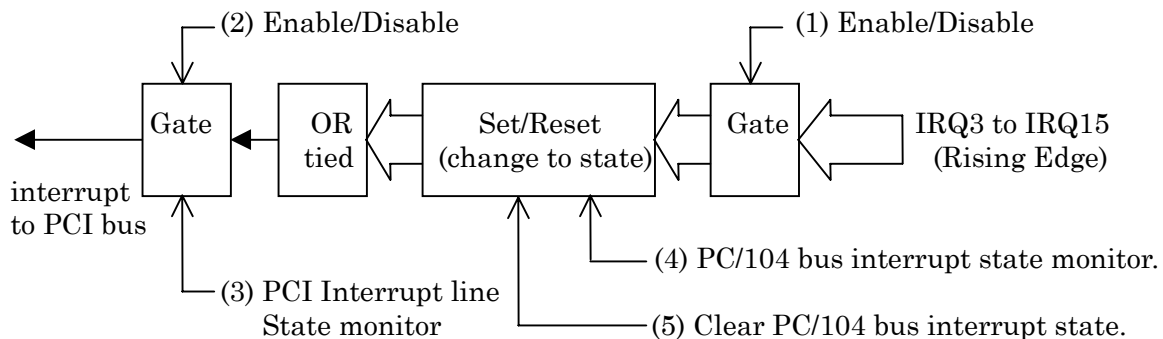
Although the library and WDM type driver come with BIF-812aK does not work with the interrupt, the hardware support the interrupt operation.

Instead of the real-time interrupt operation, the library provide the messaging function when detect the change state of the specified bit of the specified input-port, and interrupt states of PC/104 bus are assigned to the Read (BASE+4H) word register.

You can also program the application software with your own driver and library that support the interrupt operation after get the resource.

See section 1-7 how to get the resource. The programming is follows.

Figure 2-4. Interrupt Configuration



(1) Select the interrupt from PC/104 bus with the gate.

Outpw (BASE+0x6, ints) ; /* Gate for interrupts from PC/104 bus */

Enable the interrupts of PC/104 bus with Write (BASE+6H) word Register, and they operate as only one interrupt level on the PCI bus with their OR tied mode.

You can also read-back the register with Read (BASE+6H) word Register command.

Table 2-4A. Write or Read (BASE+6H) word Resister Bit-Field.

| Bit | Term | "=1" specifies | "=0" specifies | On Reset |
|-----|---------------------------|----------------|----------------|----------|
| B15 | IRQ15 | Enable | Disable | 0 |
| B14 | IRQ14 | Enable | Disable | 0 |
| B13 | Not used (fixed to "0") | | | 0 |
| B12 | IRQ12 | Enable | Disable | 0 |
| B11 | IRQ11 | Enable | Disable | 0 |
| B10 | IRQ10 | Enable | Disable | 0 |
| B9 | IRQ9 | Enable | Disable | 0 |
| B8 | Not used (fixed to "0") | | | 0 |
| B7 | IRQ7 | Enable | Disable | 0 |
| B6 | IRQ6 | Enable | Disable | 0 |
| B5 | IRQ5 | Enable | Disable | 0 |
| B4 | IRQ4 | Enable | Disable | 0 |
| B3 | IRQ3 | Enable | Disable | 0 |
| B2 | Not used (fixed to "0") | | | 0 |
| B1 | Not used (fixed to "0") | | | 0 |
| B0 | Not used (fixed to "0") | | | 0 |

(2) Enable/Disable the interrupt to PCI bus.

Outpw (BASE+0x2, irq) ; /* Gate for interrupt to PCI bus */

Write (BASE+2H) word Register specifies either enable or disable to force the interrupt to PCI bus when the OR-state of the enabled interrupts of PC/104 bus is set to ON(=True)

Table 2-4B. Write (BASE+2H) word Register Bit-Field.

| Bit | Term | "=1" specifies | "=0" specifies | On Reset |
|-----------|------------------------------------|----------------|----------------|----------|
| B15 to B1 | Not used | | | 0 |
| B0 | Interrupt force control to PCI bus | Enable | Disable | 0 |

(3) Interrupt line state monitor on PCI bus.

Ints = inpw (BASE+0x2) ; /* interrupt line on PCI bus */

Read (BASE+2H) word Register shows interrupt line state on PCI bus.

Table 2-4C. Read (BASE+2H) word Register Bit-Field.

| Bit | Term | "=1" specifies | "=0" specifies | On Reset |
|-----------|-----------------------------------|----------------|----------------|----------|
| B15 to B1 | Not used | | | 0 |
| B0 | Interrupt line monitor on PCI bus | ON (Active) | OFF | 0 |

(5) From PC/104 bus / Interrupt states monitor.

Sts = inpw (BASE+0x4) ; /* Interrupt states */

Read (BASE+4H) word Resister shows the interrupt states that should be set by the rising edge of the interrupt input from the PC/104 bus.

You can program the watching method to accept the message when the library detect the change states of this register.
See sample programs for details.

Table 2-4D. Read (BASE+4H) word Resister Bit-Field.

| Bit | Term | "=1" specifies | "=0" specifies | On Reset |
|-----|---------------------------|----------------|----------------|----------|
| B15 | Interrupt state by IRQ15 | Set | Cleared | 0 |
| B14 | Interrupt state by IRQ14 | Set | Cleared | 0 |
| B13 | Not used (fixed to "0") | | | 0 |
| B12 | Interrupt state by IRQ12 | Set | Cleared | 0 |
| B11 | Interrupt state by IRQ11 | Set | Cleared | 0 |
| B10 | Interrupt state by IRQ10 | Set | Cleared | 0 |
| B9 | Interrupt state by IRQ9 | Set | Cleared | 0 |
| B8 | Not used (fixed to "0") | | | 0 |
| B7 | Interrupt state by IRQ7 | Set | Cleared | 0 |
| B6 | Interrupt state by IRQ6 | Set | Cleared | 0 |
| B5 | Interrupt state by IRQ5 | Set | Cleared | 0 |
| B4 | Interrupt state by IRQ4 | Set | Cleared | 0 |
| B3 | Interrupt state by IRQ3 | Set | Cleared | 0 |
| B2 | Not used (fixed to "0") | | | 0 |
| B1 | Not used (fixed to "0") | | | 0 |
| B0 | Not used (fixed to "0") | | | 0 |

(6) From PC/104 bus / Interrupt states Clear.

Outpw (BASE+0x4, clr) ; /* Clear interrupt states */

Write (BASE+4H) word Resister specifies individually.
that clear or not clear the interrupt state

Table 2-4E. Write (BASE+4H) word Resister Bit-Field.

| Bit | Term | "=1" specifies | "=0" specifies | On Reset |
|-----|---------------------------|----------------|----------------|----------|
| B15 | Interrupt state by IRQ15 | Do clear | Non effect | 0 |
| B14 | Interrupt state by IRQ14 | Do clear | Non effect | 0 |
| B13 | Not used (fixed to "0") | | | 0 |
| B12 | Interrupt state by IRQ12 | Do clear | Non effect | 0 |
| B11 | Interrupt state by IRQ11 | Do clear | Non effect | 0 |
| B10 | Interrupt state by IRQ10 | Do clear | Non effect | 0 |
| B9 | Interrupt state by IRQ9 | Do clear | Non effect | 0 |
| B8 | Not used (fixed to "0") | | | 0 |
| B7 | Interrupt state by IRQ7 | Do clear | Non effect | 0 |
| B6 | Interrupt state by IRQ6 | Do clear | Non effect | 0 |
| B5 | Interrupt state by IRQ5 | Do clear | Non effect | 0 |
| B4 | Interrupt state by IRQ4 | Do clear | Non effect | 0 |
| B3 | Interrupt state by IRQ3 | Do clear | Non effect | 0 |
| B2 | Not used (fixed to "0") | | | 0 |
| B1 | Not used (fixed to "0") | | | 0 |
| B0 | Not used (fixed to "0") | | | 0 |

Section 3. General Purpose Software

3-1. Installing the Software

General purpose function library and WDM-type device driver must be installed into appropriate folders automatically when installing “BIF-812aPCI” board.

Then you had better to install additional software including the sample source, and utilities as follows.

Procedure with the CD-ROM

Open DOS-window in the PC, then type as follows. (Where from “D” drive.)

```
C:¥WINDOWS > CD¥ {Enter}
C:¥ > CD D:¥INSTALL¥PCI¥BIF812{Enter}
C:¥ > D:INSTALL D: C: {Enter}
```

Procedure with the Floppy Disk

Open DOS-window in the PC, then type as follows. (Where from “A” drive.)

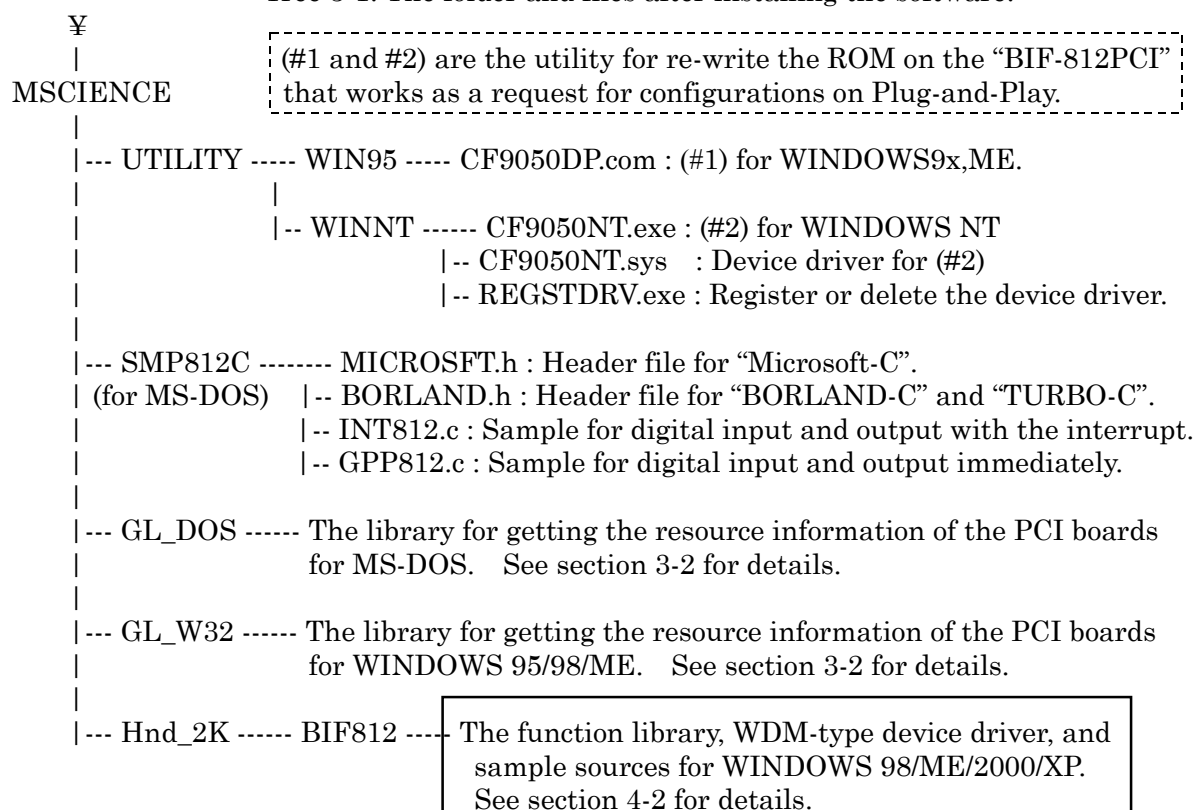
```
C:¥WINDOWS > CD¥ {Enter}
C:¥ > INSTALL A: C: {Enter}
```

The install program shall run, and ask you that if install or not for the group of files with the folder individually.

Be sure install the sample sources.

Where sample sources are including in the “Hnd_2K” folder.

Tree 3-1. The folder and files after installing the software.



3-2. Getting the resource information (not required with WDM driver)

These are not required for the function library and WDM-type device driver for WINDOWS 98/ME/2000/XP that come with this product. Exactly we recommend that. See section 4 for details.

Other hand, If you will access to “BIF-812aK” system with your own driver, the general purpose library for getting information of the resources will help you. See follows.

BIF-812aPCI works with on-board PCI interface device “PLX-9052”.

“PLX-9052” hold the resource information that assigned by the Plug-and-Play operation as described in Table 3-2.

Table 3-2. Configuration registers in the PLX-9052 as a PCI interface

| Register Address | Term | Data |
|------------------|-------------------------|------------------------|
| 00H --- --- | | |
| 18H | Base address register 2 | for memory resource. |
| --- | | |
| 1CH | Base address register 3 | for I/O resource. |
| --- | | |
| 3CH | Interrupt line | for interrupt resource |
| --- --- 3FH | | |

“MS_PCI.dll” for WINDOWS 95/98/ME/NT

This is the library for getting the resource information of BIF-812aPCI for WINDOWS 95/98/ME/NT that works with the driver.

See in the “GL_W32” folder under “MSCIENCE” folder.
The general purpose functions are follows.

(1) Detecting the PCI boards.

| | |
|-------------|---|
| Function | Int GetPciDevice (WORD VendeID, WORD DeviceID, WORD nNum, WORD Flag, word *magic) |
| Parameters | VendeID: Vender ID, Device ID: Device ID, nNum: Board number for detecting (0,1,2,3,...), Flag: 0 (constant), *magic: Pointer for the location data of the PCI board. |
| Returns | 0: success, -1: Fault |
| Description | The location pointer of the information for the PCI board shall be given as a “magic”. Vender ID = 13FDH for MICRO SCIENCE, Device ID = 0301H for BIF-812aPCI, nNum = the board number (first one is “0”) for multiple same board. |

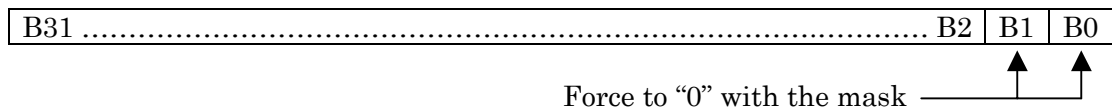
(2) Getting the pointer for the resource information (**as a double-word**)

| | |
|------------|--|
| Function | Int ReadPciDword (WORD magic, WORD reg, DWORD *data) |
| Parameters | magic: Location data of the PCI board. reg : Address of the resource information register. (See table 3-2.) *data : The pointer of the resource information. |
| Returns | 0: success, -1: Fault |

As the base address data of I/O and memory consist of double word, “**ReadPciDword**” function is recommended to get them.

Where in case of I/O base address data, lowest 2 bit should be masked to have the real I/O base address. See figure 3-2.

Figure 3-2. Base I/O address data

(3) Getting the pointer for the resource information (**as a word**)

| | |
|------------|--|
| Function | Int ReadPciWord (WORD magic, WORD reg, WORD *data) |
| Parameters | magic: Location data of the PCI board. reg : Address of the resource information register. (See table 3-2.) *data : The pointer of the resource information. |
| Returns | 0: success, -1: Fault |

(4) Getting the pointer for the resource information (**as a byte**)

| | |
|------------|--|
| Function | Int ReadPciByte (WORD magic, WORD reg, BYTE *data) |
| Parameters | magic: Location data of the PCI board. reg : Address of the resource information register. (See table 3-2.) *data : The pointer of the resource information. |
| Returns | 0: success, -1: Fault |

As the interrupt line data is between 1 to 15 if it has, and consist of a byte, “**ReadPciByte**” function is recommended to get it.

“MSPCIDx.lib” for MS-DOS

This is the library for getting the resource information of BIF-812aPCI for MS-DOS. These functions are as same as “MS_PCI” for WINDOWS, refer to the description of that.

See in the “GL_DOS” folder under “MSCIENCE” folder.
 “MSPCIDM.lib” is for Microsoft-C, and
 “MSPCIDT.lib” is for Turbo-C and Borland-C.
 The general purpose functions are follows.

(1) Detecting the PCI boards.

| | |
|-------------|---|
| Function | Int_far GetPciDevice (WORD VendeID, WORD DeviceID, WORD nNum, WORD Flag, word_far *magic) |
| Parameters | VendeID: Vender ID, Device ID: Device ID, nNum: Board number for detecting (0,1,2,3,...), Flag: 0 (constant), *magic: Pointer for the location data of the PCI board. |
| Returns | 0: success, -1: Fault |
| Description | The location pointer of the information for the PCI board shall be given as a “magic”. Vender ID = 13FDH for MICRO SCIENCE, Device ID = 0301H for BIF-812aPCI, nNum = the board number (first one is “0”) for multiple same board. |

(2) Getting the pointer for the resource information (**as a double-word**)

| | |
|------------|--|
| Function | Int ReadPciDword (WORD magic, WORD reg, DWORD _far *data) |
| Parameters | magic: Location data of the PCI board. reg : Address of the resource information register. (See table 3-2.) *data : The pointer of the resource information. |
| Returns | 0: success, -1: Fault |

(3) Getting the pointer for the resource information (**as a word**)

| | |
|------------|--|
| Function | Int ReadPciWord (WORD magic, WORD reg, WORD _far *data) |
| Parameters | magic: Location data of the PCI board. reg : Address of the resource information register. (See table 3-2.) *data : The pointer of the resource information. |
| Returns | 0: success, -1: Fault |

(4) Getting the pointer for the resource information (**as a byte**)

| | |
|------------|--|
| Function | Int ReadPciByte (WORD magic, WORD reg, BYTE _far *data) |
| Parameters | magic: Location data of the PCI board. reg : Address of the resource information register. (See table 3-2.) *data : The pointer of the resource information. |
| Returns | 0: success, -1: Fault |

Section 4. General Purpose Function Library

“H812_2K.dll” is the library to access the “BIF-812aK” system with the applications written by “Visual-C”, “Visual-C++”, or “Visual-Basic”.

4-1. Specification

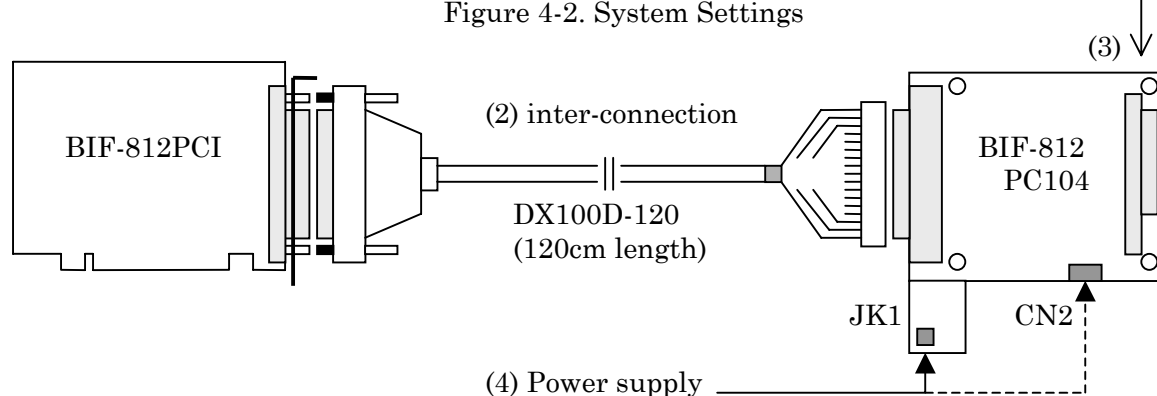
| Term | Description |
|----------------|--|
| Host PC | IBM compatible PC with 16MB or more memory. |
| OS, Compiler | WINDOWS 2000, XP, 98, ME, 32-bit coding. |
| Device driver | WDM-type (installed automatically at the BIF-812aPCI installing) |
| Sample sources | Visual-C(6.0), Visual-C++(6.0), Delphi(3.0), Visual-Basic(6.0) |
| Functions | I/O Read, Write, I/O string Read, Write. (256-byte range) Messaging when detecting the change of specified read register bit. |

<Note> Driver, DLL, and samples are made in Japan with the far-east environment of PC, and compilers.

4-2. Setting the system

- (1) Install “BIF-812aPCI” board into PC as described in section 1-7.
Where the function library and WDM-type device driver are automatically installed into appropriate folders at that same time.
See section 1-7 for details.
- (2) Connect “BIF-812aPCI” and “BIF-812aPC104” with the cable.
- (3) Connect any PC/104 bus compatible application boards by stacking up with “BIF-812aPC104”.
Upper byte of the I/O address of the application boards must be set equal to the value of “SW1” and “SW2” on the “BIF-812aPC104” board.
- (4) Supply required power to the PC/104 system with the connector JK1 or CN2.
- (5) Power on PC and start operations.

Figure 4-2. System Settings



4-3. Cording the applications

Sample sources written by “Visual-C”, “Visual-C++”, and “Visual-Basic” that work with the library are available in the folder “¥MSCIENCE¥Hnd_2K¥BIF812”.

They are made in Japan with the far-east environment of PC, and compilers.

Table 4-3A. Sample sources

| for | File name | Description |
|--------------------|-------------|---|
| Visual-C (6.0) | Sample.c | General purpose I/O functions not depend with any particular board, and sending message with detecting the change by the watching function. |
| Visual-C++ (6.0) | SampDlg.cpp | |
| Visual-Basic (6.0) | Form1.frm | |
| Borland-C (5.0) | Sample.c | |
| Delphi(3.0) | Unit1.pas | |

The profile of the functions are described in Table 4-3B, and details are in section 4-5.

Table 4-3B. Functions

| Name of the function | Description |
|-----------------------------|--|
| (1) Bif812_Open_Driver | Open library with initialize the conditions. |
| (2) Bif812_Start_Driver | Start. |
| (3) Bif812_Stop_Driver | Stop. |
| (4) Bif812_Close_Driver | Close library. |
| (5) Bif812_ReadByte | Read any register as a byte. |
| (6) Bif812_ReadWord | Read any register as a word. |
| (7) Bif812_ReadByteBlock | Read any register as a byte-block. (as a string input) |
| (8) Bif812_ReadWordBlock | Read any register as a word-block. (as a string input) |
| (9) Bif812_WriteByte | Write any register as a byte. |
| (10) Bif812_WriteWord | Write any register as a word. |
| (11) Bif812_WriteByteBlock | Write any register as a byte-block. (as a string output) |
| (12) Bif812_WriteWordBlock | Write any register as a word-block. (as a string output) |
| (13) Bif812_WatchPoint | Register the new watching term. |
| (14) Bif812_StartWatch | Start the watching. |
| (15) Bif812_StopWatch | Stop the watching. |
| (16) Bif812_GetFlags | Get the flag as detecting the change by the watching. |
| (17) Bif812_ClearFlags | Clear flag. |
| (18) Bif812_TransmitMessage | Specify for sending message. |
| (19) Bif812_RelWatch | Delete the watching term. |
| (20) Bif812_Get_Libver | Get the information of the version of this library. |

4-4. Functions

<1> Open Driver

| | |
|--------------|---|
| Function | int Bif812_Open_Driver (void) |
| Parameters | -- |
| Returns | = 0 with success, others with failure (see Table 4-4) |
| Descriptions | Initialize the driver and library, and recognize the “BIF-812aPCI” board. |

<2> Start Driver

| | |
|--------------|---|
| Function | int Bif812_Start_Driver (int board_no, int base_no, int *resource_no) |
| Parameters | board_no : “SW1” of the “BIF-812aPCI”. (default = 0, see section 2-3) base_no : Base address register number = 3 (constant, see section 3-2) *resource_no : assigned resource number as the ID for the other functions. |
| Returns | = 0 with success, others with failure (see Table 4-4) |
| Descriptions | Open and initialize the driver and library, and recognize the “BIF-812aPCI” board. |

The function <1> and <2> should be contiguously called at the entrance of the application.

<3> Stop Driver

| | |
|--------------|---|
| Function | int Bif812_Stop_Driver (int resource_no) |
| Parameters | resource_no : The value got by the <2> Start Driver function. |
| Returns | = 0 with success, others with failure (see Table 4-4) |
| Descriptions | Release the resources. |

<4> Close Driver

| | |
|--------------|---|
| Function | int Bif812_Close_Driver (void) |
| Parameters | -- |
| Returns | = 0 with success, others with failure (see Table 4-4) |
| Descriptions | Close the driver. |

The function <3> and <4> should be contiguously called at the exit of the application.

<5> Read any register as a byte

| | |
|--------------|---|
| Function | int Bif812_ReadByte (int resource_no, int address) |
| Parameters | resource_no : The value got by the <2> Start Driver function. address : Lower byte of the address for the register. (between 00H to FFH) |
| Returns | = “The value” with success, others with failure (see Table 4-4) |
| Descriptions | Read any register on the PC/104 bus as a byte. |

<6> Read any register as a word

| | |
|--------------|---|
| Function | int Bif812_ReadWord (int resource_no, int address) |
| Parameters | resource_no : The value got by the <2> Start Driver function. address : Lower byte of the address for the register. (between 00H to FEH) |
| Returns | = “The value” with success, others with failure (see Table 4-4) |
| Descriptions | Read any register on the PC/104 bus as a word. |

<7> Read any register as a byte-block

| | |
|--------------|--|
| Function | int Bif812_ReadByteBlock (int resource_no, int address, int count, UCHAR *pbuf) |
| Parameters | resource_no : The value got by the <2> Start Driver function. address : Lower byte of the address for the register. (between 00H to FFH) count : Number of data to read repeatedly by string input command. * pbuf : The pointer of the buffer area for the input data. |
| Returns | = 0 with success, others with failure (see Table 4-4) |
| Descriptions | Read any register on the PC/104 bus as a byte-block. |

<8> Read any register as a word-block

| | |
|--------------|--|
| Function | int Bif812_ReadWordBlock (int resource_no, int address, int count, USHORT *pbuf) |
| Parameters | resource_no : The value got by the <2> Start Driver function. address : Lower byte of the address for the register. (between 00H to FEH) count : Number of data to read repeatedly by string input command. * pbuf : The pointer of the buffer area for the input data. |
| Returns | = 0 with success, others with failure (see Table 4-4) |
| Descriptions | Read any register on the PC/104 bus as a word-block. |

<9> Write any register as a byte

| | |
|--------------|---|
| Function | int Bif812_WriteByte (int resource_no, int address, UCHAR data) |
| Parameters | resource_no : The value got by the <2> Start Driver function. address : Lower byte of the address for the register. (between 00H to FFH) data : output data. (1 byte) |
| Returns | = 0 with success, others with failure (see Table 4-4) |
| Descriptions | Write any register on the PC/104 bus as a byte. |

<10> Write any register as a word

| | |
|--------------|---|
| Function | int Bif812_WriteWord (int resource_no, int address, USHORT data) |
| Parameters | resource_no : The value got by the <2> Start Driver function. address : Lower byte of the address for the register. (between 00H to FEH) data : output data. (1 word) |
| Returns | = 0 with success, others with failure (see Table 4-4) |
| Descriptions | Write any register on the PC/104 bus as a word. |

<11> Write any register as a byte-block

| | |
|--------------|---|
| Function | int Bif812_Write ByteBlock (int resource_no, int address, int count, UCHAR *pbuf) |
| Parameters | resource_no : The value got by the <2> Start Driver function. address : Lower byte of the address for the register. (between 00H to FFH) count : Number of data to write repeatedly by string output command. * pbuf : The pointer of the buffer area for the output data. |
| Returns | = 0 with success, others with failure (see Table 4-4) |
| Descriptions | Write any register on the PC/104 bus as a byte-block. |

<12> Write any register as a word-block

| | |
|--------------|---|
| Function | int Bif812_Write WordBlock (int resource_no, int address, int count, USHORT *pbuf) |
| Parameters | resource_no : The value got by the <2> Start Driver function. address : Lower byte of the address for the register. (between 00H to FEH) count : Number of data to write repeatedly by string output command. * pbuf : The pointer of the buffer area for the output data. |
| Returns | = 0 with success, others with failure (see Table 4-4) |
| Descriptions | Write any register on the PC/104 bus as a word-block. |

Watching operation

The specified input register with the mask shall be watched for detecting the change state as they compared with the references by the sled.
The condition as detecting the change state is available as the following three selections.

- (1) If (inport(w_addr) & w_mask = w_data1) then
- (2) If (inport(w_addr) & w_mask! = w_data1) then
- (3) If (inport(w_addr) & w_mask = w_data1) then
If (inport(w_addr) & w_mask = w_data2) then

Where “w_addr” is the input register address for watched and compared, “mask” is the data for masking the input data before compare with the references,
“w_data1” and “w_data2” are the reference.
See function (13) to (19) for details.

Messaging operation

Sending message is also available when the changing state is detected on the specified input register with the mask by watching.
See function (18) for details.

<11> Register the new watching term.

| | |
|--------------|--|
| Function | int Bif812_WatchPoint (WORD ope, WORD addr, WORD length, WORD mask, WORD data_1, WARD data_2, WORD *watch_id) |
| Parameters | ope : Condition for detecting. / =1 for equal , =2 for not-equal, =3 for edge. / addr : Lower byte of address for the input register that should be watched. length : Input data bits length. / =1 for byte, =2 for word. / mask : masking the input data before compare with the references. data_1 : reference for equal, or for compare 1 st for detecting the edge. data_2: reference for not-equal, or for compare 2 nd for detecting the edge. watch_id : registered watching term's ID. |
| Returns | = 0 with success, others with failure (see Table 4-4) |
| Descriptions | Specify the input register and the condition for compare by the watching. Maximum 8 watching terms are able to registered. <Note> Detecting “edge” consist of two compare process. Compare input data with “data_1” and when detect their “equal”, then compare with “data_2”, and detecting their “equal” is get the “edge”. Where input data is continuously updated before each compare process. |

<14 > Start watching.

| | |
|--------------|--|
| Function | int Bif812_StartWatch (WORD watch_id) |
| Parameters | watch_id : watching term's ID (assigned by the previous function <13 >.) |
| Returns | = 0 with success, others with failure (see Table 4-4) |
| Descriptions | Start watching for the specified term by the sled. The sled shall set the flag when detecting the change as specified in previous function <13 >. |

<15 > Stop watching.

| | |
|--------------|---|
| Function | int Bif812_StopWatch (WORD watch_id) |
| Parameters | watch_id : watching term's ID (assigned by the function <13 >.) |
| Returns | = 0 with success, others with failure (see Table 4-4) |
| Descriptions | Stop watching for the specified term by the sled. |

<16 > Get the flags.

| | |
|--------------|---|
| Function | int Bif812_GetFlags (WORD watch_id, WORD *flags) |
| Parameters | watch_id : watching term's ID (assigned by the function <13 >.) flags : Set by detecting the change. (=1 for set, =0 for cleared or default) |
| Returns | = 0 with success, others with failure (see Table 4-4) |
| Descriptions | Get the flag as detecting the change by the specified watching term. |

<17 > Clear flags.

| | |
|--------------|---|
| Function | int Bif812_ClearFlags (WORD watch_id) |
| Parameters | watch_id : watching term's ID (assigned by the function <13 >.) |
| Returns | = 0 with success, others with failure. (see Table 4-4) |
| Descriptions | Clear the relative flag of the watching term. |

<18 > Specify for sending message.

| | |
|--------------|---|
| Function | int Bif812_TransmitMessage (WORD watch_id, HWND hWnd) |
| Parameters | watch_id : watching term's ID (assigned by the function <13 >.) hWnd : destination Handle of the window. |
| Returns | = 0 with success, others with failure (see Table 4-4) |
| Descriptions | The sled shall send the message to the specified window at detecting the change by the watching. (also set the flag.) The message consist of "watch_id" and "the input data of just then". |

<19> Delete watching term.

| | |
|--------------|---|
| Function | int Bif812_Relwatch (WORD watch_id) |
| Parameters | watch_id : watching term's ID (assigned by the function <13 >.) |
| Returns | = 0 with success, others with failure. (see Table 4-4) |
| Descriptions | Delete the watching term. |

<20> Get information for the version.

| | |
|--------------|---|
| Function | int Bif812_Get_Libver (int ver) |
| Parameters | ver : point to the information for return. = 0 for "major version number" + "minor version number" = 1 for "major version number" only. = 2 for "minor version number" only. |
| Returns | = plus value as a information for the version number with success, = minus value with failure. (see Table 4-4) |
| Descriptions | For example, This function get "0x101" as "the version = 1.01" with "ver = 0" |

Table 4-4. Returns with the failure.

| Return | Description. |
|--------|--|
| - 1 | Not found the driver. (DMS_PCL.sys) |
| - 2 | Not found the board. (BIF-812K hardware system) |
| - 3 | Not initialized. |
| - 4 | Not appropriate value of "board_no". (See function <2> Start Driver for details.) |
| - 5 | Not appropriate value of the parameter. |
| - 6 | Not started the driver. |
| - 7 | |
| - 8 | |
| - 9 | |
| -10 | Not appropriate "watch_id". |
| -11 | Not acceptable to register another watching term. (maximum 8 terms.) |
| -12 | Not registered "watch_id". |
| -13 | Already running to watch. |
| -14 | Not running to watch. |
| | |

Section 5. Maintenance and Appendix

5-1. Trouble Shootings

Reconfirm.

The BIF-812aK supplied by MICRO SCIENCE is fully tested.

If it doesn't work on your PC, reconfirm following issues.

- (1) Check the assigned I/O BASE address of "BIF-812aPCI" on the PCI-bus by the "Device-Manager" of WINDOWS.
- (2) Check the I/O BASE address on the PC/104-bus specified by "BIF-812aPC104" on-board switch SW1 and SW2. Then check the I/O base address of the adapted PC/104 board if it is adjustable. By the way, on the IBM PC/AT compatible system, the I/O address must be mapped between "0H" to "3FFH" or the image of this range except for the occupied address by the other devices or the peripherals.
- (3) Debug your software or applications. For example, if the Interrupt line is correct or if conflict with any other devices.
- (4) Check the power supply with the connector CN2 for "BIF-812aPC104" and PC/104-bus.

What's wrong?

Fill in and send (Letter, Fax, or Email) the Q&A form to MICRO SCIENCE where you didn't find anything wrong.

Although we will study about your system and answer by the letter what you should better to do, we don't write or debug application software.

Sorry, we won't answer with any language but Japanese on the phone. Please write us Japanese or English.

Replace the Board or Repair for free.

MICRO SCIENCE will replace or repair the Board for free which are after examination disclosed to the satisfaction of MICRO SCIENCE to be thus defective, for a period within one year of shipment. This warranty shall not apply which have been subject to misuse, negligence, or accident. See "Caution/Warranty" for details in page-3.

Repair the Board.

MICRO SCIENCE will repair, calibrate, or test the Board on request. These products should have to prepaid the transportation at MICRO SCIENCE. Be sure, give us the information with the products, maybe Q&A form is useful for the report.

Then user have to pay the proper cost in few weeks according to the bill after accept the returned products.

Q & A form (in English or Japanese)

To:
 MICRO SCIENCE., Co. LTD
 Customer Support Div
 2-37-12, Nishiogi-kita,
 Suginami-ku,
 Tokyo, Japan

From:

Fax: +81-3-3301-5593
 Email: qas@microscience.co.jp

Fax:
 Email:

| | | |
|---------------------------|--|--------------------------------------|
| BIF-812aK | serial # = | Purchase Date: |
| Preferences on- Board | (BIF-812aPCI) SW1 = (BIF-812aPC104) SW1= SW2 = | |
| PC/104 Application boards | Product: Occupied Resources: (I/O Address =), (Interrupt =) | |
| PC System Information | CPU: | Assigned I/O BASE address on PCI-bus |
| | OS : | |
| | | |
| Software | Language: | |
| | Compiler: | |

(Information)

<Note> MICR SCIENCE does not answer on phone with any language but Japanese.