DIO4264

Digital I/O Card

Software Manual (V1.1)

健昇科技股份有限公司

JS AUTOMATION CORP.

新北市汐止區中興路 100 號 6 樓

6F., No.100, Zhongxing Rd.,

Xizhi Dist., New Taipei City, Taiwan

TEL：+886-2-2647-6936

FAX：+886-2-2647-6940

http://www.automation.com.tw

http://www.automation-js.com/

E-mail：control.cards@automation.com.tw

Correction record

Version	Record
1.0	for driver v1.0 up
1.0->1.1	Modify all function name (lowercase → uppercase)

Contents

1. How to install the software of DIO4264

1.1 Install the driver

The PC104 card is similar to ISA bus card, before you add a new card on the window system you must assign I/O spaces to it (by hardware). The driver hooked can access the registers of the card.

Please follow the following steps to install your new card.

In Windows 98/2000/XP system you should: (take winXP as example)

Driver install:

1. Execute the DIO4264_install.exe under
   ..\download\Manual_Driver\DioCard\DIO4264\software\win2k_xp\install\
2. Follows the install wizard, you will complete the installation.

Card install:

3. Make sure the power is off
4. Setting the card address on a spare I/O space.
5. Plug in the interface card
6. Power on, it’s ready to use

2. Where to find the file you need

Windows98, 2000,XP

In Windows 98, 2000,XP system, the files will be located under

.. / JS Automation /DIO4264/Api (header files and VB,VC lib files)

.. / JS Automation /DIO4264/Driver (copy of drivers)

.. / JS Automation /DIO4264/exe (demo program and source code)

The system driver is located at ../system32/Driver and the DLL is located at ../system.

3. About the DIO4264 software

DIO4264 software includes a set of dynamic link library (DLL) and system driver that you can utilize to control the I/O card's ports and points separately.

Your DIO4264 software package includes setup driver, test program that help you how to setup and run appropriately, as well as an executable file which you can use to test each of the DIO4264 functions within Windows' operation system environment.

3.1 What you need to get started

To set up and use your DIO4264 software, you need the following:

- DIO4264 software
- DIO4264 hardware
  Main board
  Wiring board (Option)

3.2 Software programming choices

You have several options to choose from when you are programming DIO4264 software. You can use Borland C/C++, Microsoft Visual C/C++, Microsoft Visual Basic, or any other Windows-based compiler that can call into Windows dynamic link libraries (DLLs) for use with the DIO4264 software.

4. DIO4264 Language support

The DIO4264 software library is a DLL used with Windows 98/2000/XP. You can use these DLL with any Windows integrating development environment that can call Windows DLLs.

4.1 Building applications with the DIO4264 software library

The DIO4264 function reference topic contains general information about building DIO4264 applications, describes the nature of the DIO4264 files used in building DIO4264 applications, and explains the basics of making applications using the following tools:

Applications tools

- **Borland C/C++**
- **Microsoft Visual C/C++**
- **Microsoft Visual Basic**

If you are not using one of the tools listed, consult your development tool reference manual for details on creating applications that call DLLs.

4.2 DIO4264 Windows libraries

The DIO4264 for Windows function library is a DLL called **DIO4264.dll**. Since a DLL is used, DIO4264 functions are not linked into the executable files of applications. Only the information about the DIO4264 functions in the DIO4264 import libraries is stored in the executable files.
Import libraries contain information about their DLL-exported functions. They indicate the presence and location of the DLL routines. Depending on the development tools you are using, you can make your compiler and linker aware of the DLL functions through import libraries or through function declarations.

Refer to **Table 1** to determine to which files you need to link and which to include in your development to use the DIO4264 functions in DIO4264.dll.

Header Files and Import Libraries for Different Development Environments		
Development Environment	**Header File**	**Import Library**
Microsoft C/C++	DIO4264.h	DIO4264vc.lib
Borland C/C++	DIO4264.h	DIO4264bc.lib
Microsoft Visual Basic	DIO4264.bas	

Table 1

5. Software overview

These topics describe the features and functionality of the DIO4264 boards and briefly describes the DIO4264 functions.

5.1 Initialization and close

You need to initialize system resource each time you run your application.

DIO4264_initial( ) will do.

Once you want to close your application, call

DIO4264_close( ) to release all the resource.

5.2 I/O Port R/W

Before using the DIO4264 I/O function, you must configure the port function first.

DIO4264_config_port( ) will assign the configuration of all the ports.

Use the following functions for I/O port output value reading and control:

DIO4264_read_port( ) to read a byte data from I/O port,

DIO4264_set_port( ) to output byte data to output port,

DIO4264_set_point( ) to set output bit,

DIO4264_read_point( ) to read I/O bit,

5.3 Error conditions

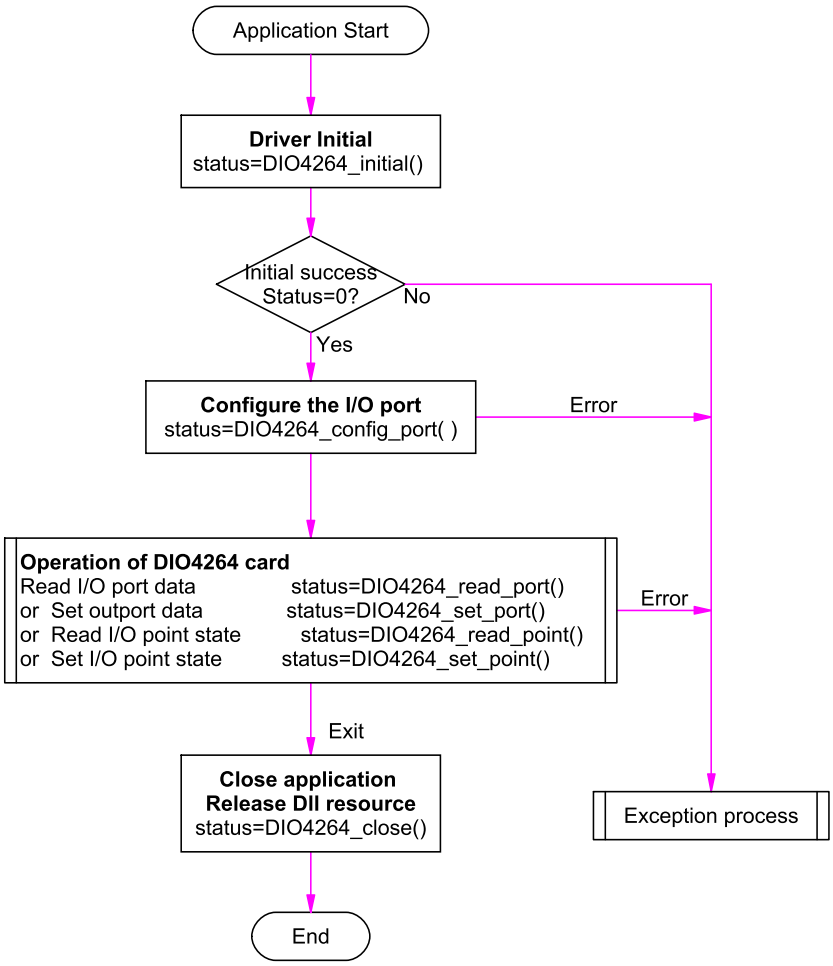
DIO4264 cards minimize error conditions. There are three possible fatal failure modes:

- System Fail Status Bit Valid
- Communication Loss
- Hardware not ready

These error types may indicate an internal hardware problem on the board. Error Codes contains a detailed listing of the error status returned by DIO4264 functions.

6. Flow chart of application implementation

6.1 DIO4264 Flow chart of application implementation



7. Function reference

7.1 Function format

Every DIO4264 function is consist of the following format:

Status = function_name (parameter 1, parameter 2, … parameter n);

Each function returns a value in the **Status** global variable that indicates the success or failure of the function. A returned **Status** equal to zero that indicates the function executed successfully. A non-zero status indicates failure that the function did not execute successfully because of an error, or executed with an error.

Note : **Status** is a 32-bit unsigned integer.

The first parameter to almost every DIO4264 function is the parameter **Address** which is located the driver of DIO4264 board you want to use those given operation. The **Address** is assigned by DIP/ROTARY SW. You can utilize multiple devices with different card **Address** within one application; to do so, simply pass the appropriate **Address** to each function.

Note: **Address** is set by DIP/ROTARY SW (**0x00-0xFF**) and must care not to overlap with the system existing I/O address.

These topics contain detailed descriptions of each DIO4264 function. The functions are arranged alphabetically by function name. Refer to DIO4264 Function Reference for additional information.

7.2 Variable data types

Every function description has a parameter table that lists the data types for each parameter. The following sections describe the notation used in those parameter tables and throughout the manual for variable data types.

Primary Type Names					
Name	**Description**	**Range**	**C/C++**	**Visual BASIC**	**Pascal (Borland Delphi)**
u8	8-bit ASCII character	0 to 255	char	Not supported by BASIC. For functions that require character arrays, use string types instead.	Byte
I16	16-bit signed integer	-32,768 to 32,767	short	Integer (for example: deviceNum%)	SmallInt
U16	16-bit unsigned integer	0 to 65,535	unsigned short for 32-bit compilers	Not supported by BASIC. For functions that require unsigned integers, use the signed integer type instead. See the i16 description.	Word
I32	32-bit signed integer	-2,147,483,648 to 2,147,483,647	long	Long (for example: count&)	LongInt
U32	32-bit unsigned integer	0 to 4,294,967,295	unsigned long	Not supported by BASIC. For functions that require unsigned long integers, use the signed long integer type instead. See the i32 description.	Cardinal (in 32-bit operating systems). Refer to the i32 description.
F32	32-bit single-precision floating-point value	-3.402823E+38 to 3.402823E+38	float	Single (for example: num!)	Single
F64	64-bit double-precision floating-point value	-1.797683134862315E+308 to 1.797683134862315E+308	double	Double (for example: voltage Number)	Double

Table 2

7.3 Programming language considerations

Apart from the data type differences, there are a few language-dependent considerations you need to be aware of when you use the DIO4264 API. Read the following sections that apply to your programming language.

Note: Be sure to include the declaration functions of DIO4264 prototypes by including the appropriate DIO4264 header file in your source code. Refer to Building Applications with the DIO4264 Software Library for the header file appropriate to your compiler.

7.3.1 C/C++

For C or C++ programmers, parameters listed as Input/Output parameters or Output parameters are pass-by-reference parameters, which means a pointer points to the destination variable should be passed into the function. For example, the Read Port function has the following format:

Status = DIO4264_read_port(u16 Address, u8 port, u8*data);

where **Address** and **port** are input parameters, and **data** is an output parameter. Consider the following example:

u16 Address;
u8 port;
u8 data,
u32 Status;
Status = DIO4264_read_port (Address, port, &data);

7.3.2 Visual basic

The file DIO4264.bas contains definitions for constants required for obtaining DIO Card information and declared functions and variable as global variables. You should use these constants symbols in the DIO4264.bas, do not use the numerical values.

In Visual Basic, you can add the entire DIO4264.bas file into your project. Then you can use any of the constants defined in this file and call these constants in any module of your program. To add the DIO4264.bas file for your project in Visual Basic 4.0, go to the **File** menu and select the **Add File... option**. Select Dio4264.bas, which is browsed in the DIO4264 \ api directory. Then, select **Open** to add the file to the project.

To add the DIO4264.bas file to your project in Visual Basic 5.0 and 6.0, go to the **Project** menu and select **Add Module**. Click on the Existing tab page. **Select** DIO4264.bas, which is in the DIO4264 \ api directory. Then, select **Open** to add the file to the project.

7.3.3 Borland C++ builder

To use Borland C++ builder as development tool, you should generate a .lib file from the .dll file by implib.exe.

implib DIO4264bc.lib DIO4264.dll

Then add the **DIO4264bc.lib** to your project and add

#include "DIO4264.h" to main program.

Now you may use the dll functions in your program. For example, the Read Port function has the following format:

Status = DIO4264_read_port(u16 Address, u8 port, u8*data);

where **Address** and **port** are input parameters, and **data** is an output parameter. Consider the following example:

u16 Address;
u8 port;
u8 data,
u32 Status;
Status = DIO4264_read_port (Address, port, &data);

7.4 DIO4264 Functions

Initialization and close

- **DIO4264_initial**

Format : u32 status =DIO4264_ initial (void)

Purpose: Initial the DIO4264 resource when start the Windows applications**.** A success will returns 0 else fail returns 1.

- **DIO4264_close**

Format : u32 status =DIO4264_close (void);

Purpose: Release the DIO4264 resource when close the Windows applications.

I/O Port R/W

- **DIO4264_config_port**

Format : u32 status = DIO4264_config_port(u16 Address,
CONFIG_IO_MODE ConfigIoMode)

Purpose: configure I/O port.

Parameters:

Input:

Name	Type	Description
Address	u16	assigned by DIP SW

ConfigIoMode	CONFIG_IO_MODE	struct CONFIG_IO_MODE
		{
		u8 A0 ;
		u8 B0 ;
		u8 C0_HighNibble;
		u8 C0_LowNibble;
		u8 A1 ;
		u8 B1 ;
		u8 C1_HighNibble;
		u8 C1_LowNibble;
		}
		
		n: the PPI number, maybe 0 or 1
		An=1 : portA as input
		0 : portA as output
		Bn=1 : portB as input
		0 : portB as output
		Cn_HighNibble
		=1
		portC / portD high nibble as input
		Cn_HighNibble
		=0
		portC / portD high nibble as output
		Cn_LowNibble
		=1
		portC / portD low nibble as input
		=0
		portC / portD low nibble as output

Note: All the I/O pins have pull up by 10K resistors.

- **DIO4264_read_port**

Format : **u32 status = DIO4264_read_port (u16 Address , u8 port , u8 *data)**

Purpose: Read the current data of the I/O port.

Parameters:

Input:

Name	Type	Description
Address	u16	assigned by DIP SW
port	u8	port number 0: PPI0A (port bit0~bit7) 1: PPI0B (port bit8~bit15) 2: PPI0C (port bit16~bit23) 3: PPI0D (port bit 24~bit31) 4: PPI1A (port bit 32~bit39) 5: PPI1B (port bit 40~bit47) 6: PPI1C (port bit 48~bit55) 7: PPI1D (port bit 56~bit63)

Output:

Name	Type	Description
data	u8	I/O data, 0~255

- **DIO4264_set_port**

Format : **u32 status = DIO4264_set_port (u16 Address,u8 port, u8 data)**

Purpose: Sets the output port data.

Parameters:

Input:

Name	Type	Description
Address	u16	assigned by DIP SW
port	u8	port number 0: PPI0A (port bit0~bit7) 1: PPI0B (port bit8~bit15) 2: PPI0C (port bit16~bit23) 3: PPI0D (port bit 24~bit31) 4: PPI1A (port bit 32~bit39) 5: PPI1B (port bit 40~bit47) 6: PPI1C (port bit 48~bit55) 7: PPI1D (port bit 56~bit63)
data	u8	output values, 0~255

- **DIO4264_set_point**

Format : **u32 status =DIO4264_set_point(u16 Address, u8 point, u8 state)**

Purpose: Sets the bit data of I/O port.

Parameters:

Input:

Name	Type	Description
Address	u8	assigned by Rotary SW
point	u8	point number 0~63 for bit0~bit63
state	u8	state of I/O point, 0~1

- **DIO4264_read_point**

Format : **u32 status = DIO4264_read_point (u16 Address , u8 point , u8 *state)**

Purpose: Read the current point data of the I/O port.

Parameters:

Input:

Name	Type	Description
Address	u16	assigned by DIP SW
point	u8	point number 0~63 for bit0 ~bit63

Output:

Name	Type	Description
state	u8	I/O point status, 0~1

7.5 Dll list

	Function Name	Description
1	DIO4264_initial( )	DIO4264 initial
2	DIO4264_close( )	DIO4264 close
3	DIO4264_config_port( )	configure the port function of DIO4264
4	DIO4264_read_port( )	read port data
5	DIO4264_set_port( )	set output port data
6	DIO4264_set_point( )	set point state(bit)
7	DIO4264_read_point( )	read point state(bit)

8. DIO4264 Error codes summary

8.1 DIO4264 Error codes table

Error Code	Symbolic Name	Description
0	JSDRV_NO_ERROR	No error.
1	JSDRV_INIT_ERROR	Driver initial error
100	DEVICE_RW_ERROR	Device Read/Write error
101	JSDRV_NO_CARD	No DIO4264 card on the system.
102	JSDRV_DUPLICATE_ID	DIO4264 Address duplicate error.
300	JSDIO_ADDRESS_ERROR	Function address parameter error. Address setting error, Address doesn't match the DIP SW setting
301	JSDIO_PORT_ERROR	Function port parameter error. Parameter out of range.
302	JSDIO_POINT_ERROR	Function point parameter error. Parameter out of range.