

# DIO-3264

## Digital I/O Card

### Software Manual (V1.3)

健昇科技股份有限公司

**JS AUTOMATION CORP.**

台北縣汐止市中興路 100 號 6 樓

6F, No. 100, Chungshin Rd.

Shitsu, Taipei, Taiwan, R.O.C.

TEL : 886-2-2647-6936

FAX : 886-2-2647-6940

<http://www.automation.com.tw>

E-mail : [control.cards@automation.com.tw](mailto:control.cards@automation.com.tw)

## Correction record

Version	Record
1.1->1.2	For driver version 2.0 up
	1 Add software key function
	<i>dio3264_set_password()</i>
	<i>dio3264_change_password()</i>
	<i>dio3264_clear_password()</i>
	<i>dio3264_unlock_security()</i>
	<i>dio3264_read_security_status()</i>
1.2->1.3	1 For driver version 4.0 up
	Revise how to install

# Contents

1.	How to install the software of DIO3264.....	4
1.1	Install the PCI driver.....	4
2.	Where to find the file you need .....	5
3.	About the DIO-3264 software .....	6
3.1	What you need to get started.....	6
3.2	Software programming choices .....	6
4.	DIO3264 Language support.....	7
4.1	Building applications with the DIO3264 software library.....	7
4.2	DIO3264 Windows libraries .....	7
5.	Software overview .....	8
5.1	Initialization .....	8
5.2	I/O Port R/W .....	8
5.3	Error conditions .....	9
6.	Flow chart of implement an application .....	10
6.1	DIO3264 Flow chart of implementation.....	10
7.	Function reference .....	12
7.1	Error codes and CardID .....	12
7.2	Variable data types .....	13
7.3	Programming language considerations .....	14
7.4	DIO3264 Functions .....	16
	Initialization .....	16
	dio3264_initial.....	16
	dio3264_close .....	16
	dio3264_info.....	16
	dio3264_get_device_handle .....	16
	I/O Port R/W .....	17
	dio3264_read_port.....	17
	dio3264_read_in_point.....	17
	dio3264_enable_IRQ.....	18
	dio3264_disable_IRQ.....	18
	dio3264_set_IRQ_mask .....	19
	dio3264_IRQ_status .....	19
	dio3264_link_IRQ_process .....	20
	dio3264_set_password.....	20
	dio3264_change_password.....	21
	dio3264_clear_password .....	21
	dio3264_unlock_security.....	21
	dio3264_read_security_status.....	22
7.5	Dll list .....	23

8.	Reference .....	24
8.1	DIO3264 I/O Port-Point table.....	24
9.	DIO-3264 Error codes summary.....	25
9.1	DIO3264 Error codes table.....	25

# 1. **How to install the software of DIO3264**

## 1.1 Install the PCI driver

The PCI card is a plug and play card, once you add a new card on the window system will detect while it is booting. Please follow the following steps to install your new card.

In Windows 2000/XP/Vista system you should:

1. Make sure the power is off
2. Plug in the interface card
3. Power on
4. A hardware install wizard will appear and tell you it finds a new PCI card
5. Do not response to the wizard, just Install the file  
    \DIO3264\Software\Win2K\_up\DIO3264\_Install.exe
6. After installation, power off
7. Power on, it's ready to use

For more detail of step by step installation guide, please refer the file “installation.pdf “ on the CD come with the product or register as a member of our user's club at:

<http://automation.com.tw/>

to download the complementary documents.

## 2. **Where to find the file you need**

### **Windows 2000, XP and up**

In Windows 2000,XP,Vista system, the demo program can be setup by

**\DIO3264\Software\Win2K\_up\DIO3264\_Install.exe**

The directory will be located at

**../ JS Automation /DIO3264/API** (header files and VB,VC lib files)

**../ JS Automation / DIO3264/Driver** (copy of driver code)

**../ JS Automation /DIO3264/exe** (demo program and source code)

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

### **3. About the DIO-3264 software**

DIO3264 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 DIO3264 software package includes setup driver, tutorial example and 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 DIO3264 functions within Windows' operation system environment.

#### 3.1 What you need to get started

To set up and use your DIO3264 software, you need the following:

- DIO3264 software
- DIO3264 hardware
  - Main board
  - Wiring board (Option)

#### 3.2 Software programming choices

You have several options to choose from when you are programming DIO3264 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 DIO3264 software.

## 4. **DIO3264 Language support**

The DIO3264 software library is a DLL used with Windows 2000/XP/Vista. You can use these DLL with any Windows integrating development environment that can call Windows DLLs.

### 4.1 Building applications with the DIO3264 software library

The DIO3264 function reference topic contains general information about building DIO3264 applications, describes the nature of the DIO3264 files used in building DIO3264 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 DIO3264 Windows libraries

The DIO3264 for Windows function library is a DLL called **dio3264.dll**. Since a DLL is used, DIO3264 functions are not linked into the executable files of applications. Only the information about the DIO3264 functions in the DIO3264 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 DIO3264 functions in dio3264.dll.

Header Files and Import Libraries for Different Development Environments		
<b>Development Environment</b>	<b>Header File</b>	<b>Import Library</b>
<b>Microsoft C/C++</b>	Dio3264.h	Dio3264VC.lib
<b>Borland C/C++</b>	Dio3264.h	Dio3264BC.lib
<b>Microsoft Visual Basic</b>	Dio3264.bas	

**Table 1**

## 5. Software overview

These topics describe the features and functionality of the DIO3264 boards and briefly describes the DIO3264 functions.

### 5.1 Initialization

You need to initialize each time you run your application.

[\*dio3264 initial\(\)\*](#)

[\*dio3264 close\(\)\*](#)

[\*dio3264 info\(\)\*](#)

[\*dio3264 get device handle\(\)\*](#)

### 5.2 I/O Port R/W

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

Read I/O Port

[\*dio3264 read port\(\)\*](#)

Read I/O Point

[\*dio3264 read in point\(\)\*](#)

Interrupt Function

[\*dio3264 enable IRO\(\)\*](#)

[\*dio3264 disable IRO\(\)\*](#)

[\*dio3264 set IRO mask\(\)\*](#)

[\*dio3264 IRO status\(\)\*](#)

[\*dio3264 link IRO process\(\)\*](#)

Software Key Function

[\*dio3264 set password\(\)\*](#)

[\*dio3264 change password\(\)\*](#)

[\*dio3264 clear password\(\)\*](#)

[\*dio3264 unlock security\(\)\*](#)

[\*dio3264 read security status\(\)\*](#)

### 5.3 Error conditions

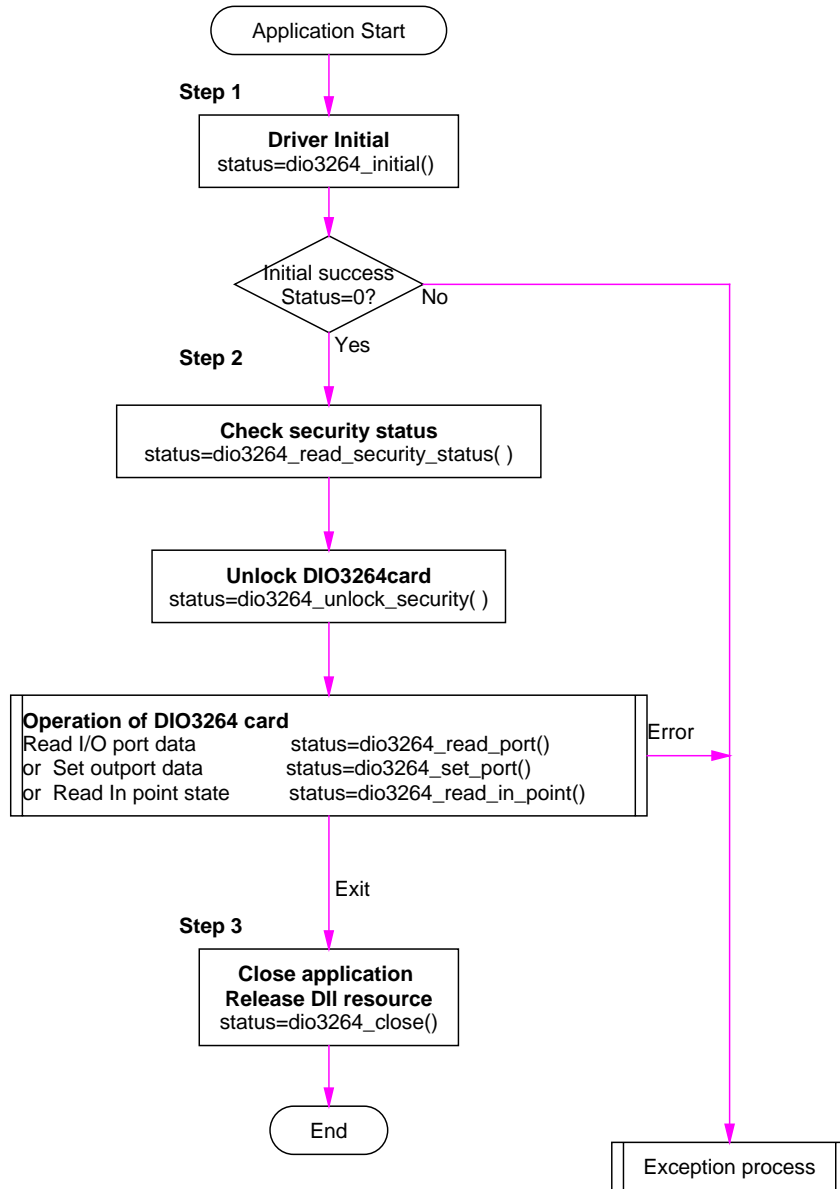
DIO3264 cards minimize error conditions. There are three possible fatal failure modes:

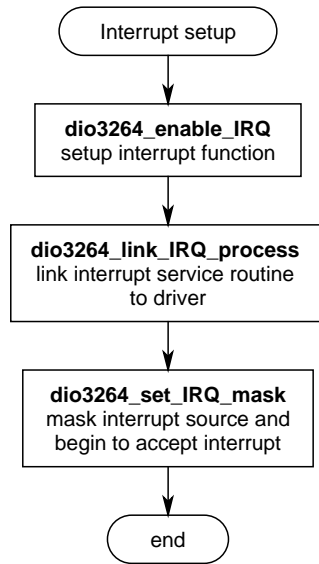
- ◆ System Fail Status Bit Valid
- ◆ Communication Loss
- ◆ Hardware not ready

These error types may indicate an internal hardware problem on the board. Error Codes contains a detailed listing of the error status returned by DIO3264 functions.

## 6. Flow chart of implement an application

### 6.1 DIO3264 Flow chart of implementation





## 7. **Function reference**

These topics contain detailed descriptions of each DIO3264 function. The functions are arranged alphabetically by function name. Refer to DIO3264 Function Reference for additional information.

### 7.1 Error codes and CardID

Every DIO3264 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 DIO3264 function is the parameter **CardID** which is located the driver of DIO3264 board you want to use those given operation. The **CardID** is assigned by DIP SW. You can utilize multiple devices with different card CardID within one application; to do so, simply pass the appropriate **CardID** to each function.

**Note**: **CardID** is set by DIP SW (**0x0-0xF**)

## 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)
<b>u8</b>	8-bit ASCII character	0 to 255	char	Not supported by BASIC. For functions that require character arrays, use string types instead.	Byte
<b>I16</b>	16-bit signed integer	-32,768 to 32,767	short	Integer (for example: deviceNum%)	SmallInt
<b>U16</b>	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
<b>I32</b>	32-bit signed integer	-2,147,483,648 to 2,147,483,647	long	Long (for example: count&)	LongInt
<b>U32</b>	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.
<b>F32</b>	32-bit single-precision floating-point value	-3.402823E+38 to 3.402823E+38	float	Single (for example: num!)	Single
<b>F64</b>	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 DIO3264 API. Read the following sections that apply to your programming language.

**Note:** Be sure to include the declaration functions of DIO3264 prototypes by including the appropriate DIO3264 header file in your source code. Refer to Building Applications with the DIO3264 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 = dio3264_read_port(CardID, port, data);
```

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

```
u8 CardID, port;  
u8 data,  
u32 Status;  
Status = read_port (CardID, port, &data);
```

#### 7.3.2 Visual basic

The file dio3264.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 dio3264.bas, do not use the numerical values.

In Visual Basic, you can add the entire dio3264.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 dio3264.bas file for your project in Visual Basic 4.0, go to the **File** menu and select the **Add File...** option. Select Dio3264.bas, which is browsed in the dio3264 \ api directory. Then, select **Open** to add the file to the project.

To add the dio3264.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** dio3264.bas, which is in the dio3264 \ 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 dio3264bc.lib dio3264.dll**

Then add the **dio3264bc.lib** to your project and add

**#include "dio3264.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 = dio3264\_read\_port(CardID, port, data);**

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

*u16 CardID, port;*

*u8 data;*

*u32 Status;*

*Status = read\_port (CardID, port, &data);*

**Initialization**

● **dio3264\_initial**

**Description:** DIO3264 Initial

**Format :** u32 status =dio3264\_initial (void)

**Purpose:** Initial the DIO3264 resource when start the Windows applications.

● **dio3264\_close**

**Description:** DIO3264 Close

**Format :** u32 status =dio3264\_close (void);

**Purpose:** Release the DIO3264 resource when close the Windows applications.

● **dio3264\_info**

**Format :** u32 status = dio3264\_info(u8 CardID,u16 \*Ven\_ID,u16 \*address);

**Purpose:** Read the physical I/O address assigned by O.S.

**Parameters:**

**Input:**

Name	Type	Description
CardID	u8	assigned by DIP/ROTARY SW

**Output:**

Name	Type	Description
Ven_ID	u16	return sub system ID(0x3264)
address	u16	physical I/O address assigned by OS

● **dio3264\_get\_device\_handle**

**Format :** u32 status =dio3264\_get\_device\_handle(u8 CardID,HANDLE \*DeviceHandle)

**Purpose:** Read the device handle assigned by O.S..

**Parameters:**

**Input:**

Name	Type	Description
CardID	u8	assigned by DIP/ROTARY switch

**Output:**

Name	Type	Description
DeviceHandle	HANDLE	Handle assigned by O.S.

## I/O Port R/W

### ● **dio3264\_read\_port**

**Description:** Read Port Data (byte)

**Format :** u32 status = dio3264\_read\_port (u8 CardID , u8 port , u8 \*data)

**Purpose:** Read the output values of the I/O port.

**Parameters:**

**Input:**

Name	Type	Description
CardID	u8	assigned by DIP SW
port	u8	port number

**Output:**

Name	Type	Description
data	u8	bitmap of output values

#### **Parameter Discussion**

**CardID:** the CardID set by DIP SW (0x0-0xF)

**port :** 0 to 7

**data :** 0 to 255

### ● **dio3264\_read\_in\_point**

**Description:** Read Input Point State(bit)

**Format :** u32 status =dio3264\_read\_in\_point(u8 CardID, u8 point, u8 \*state)

**Purpose:** Read the input state of the I/O points.

**Parameters:**

**Input:**

Name	Type	Description
CardID	u8	assigned by DIP SW
point	u8	point number

**Output:**

Name	Type	Description
state	u8	point of output state

#### **Parameter Discussion**

**CardID:** the CardID set by DIP SW (0x0-0xF)

**point :** DIO3264 range 0 to 63 ;

**state :** 1 = True; 0 = False

● **dio3264 enable IRQ**

**Description:** enable interrupt

**Format :** u32 status = dio3264\_enable\_IRQ (u8 CardID, HANDLE \*phEvent)

**Purpose:** Enable interrupt from IN0, IN1

**Parameters:**

**Input:**

Name	Type	Description
CardID	u8	assigned by DIP SW

**Output:**

Name	Type	Description
phEvent	HANDLE	event handle

**Parameter Discussion**

**CardID:** the CardID set by DIP SW (0x0-0xF)

**phEvent:** the returned handle of driver

● **dio3264 disable IRQ**

**Description:** Disable interrupt

**Format :** u32 status = dio3264\_disable\_IRQ (u8 CardID)

**Purpose:** Disable interrupt from IN0, IN1

**Parameters:**

**Input:**

Name	Type	Description
CardID	u8	assigned by DIP SW

**Parameter Discussion**

**CardID:** the CardID set by DIP SW (0x0-0xF)

- **dio3264 set IRQ mask**

**Description:** mask interrupt

**Format :** u32 status = dio3264\_set\_IRQ\_mask (u8 CardID, u16 Data)

**Purpose:** Mask interrupt from IN0, IN1

**Parameters:**

**Input:**

Name	Type	Description
CardID	u8	assigned by DIP SW
Data	u16	bit0: 0, disable irq from IN0 1, enable irq from IN0 bit1: 0, disable irq from IN1 1, enable irq from IN1

**Parameter Discussion**

**CardID:** the CardID set by DIP SW (0x0-0xF)

- **dio3264 IRQ status**

**Description:** read back interrupt source

**Format :** u32 status = dio3264\_IRQ\_status (u8 CardID, u32 \*Event\_Status)

**Purpose:** To read back the interrupt source to identify IN0 or IN1

**Parameters:**

**Input:**

Name	Type	Description
CardID	u8	assigned by DIP SW

**Output:**

Name	Type	Description
Event_Status	u32	bit0: 1, irq source from IN0 bit1: 1, irq source from IN1

**Parameter Discussion**

**CardID:** the CardID set by DIP SW (0x0-0xF)

- **dio3264 link IRQ process**

**Description:** link interrupt routine

**Format :** `u32 status = dio3264_link_IRQ_process (u8 CardID,  
void ( __stdcall *callbackAddr)(u8 CardID));`

**Purpose:** Link irq service routine to driver

**Parameters:**

**Input:**

Name	Type	Description
CardID	u8	assigned by DIP SW
__stdcall	void	callback address of service routine

**Parameter Discussion**

**CardID:** the CardID set by DIP SW (0x0-0xF)

- **dio3264 set password**

**Format :** `u32 status = dio3264_set_password(u8 CardID,u16 password[5]);`

**Purpose:** To set password and if the password is not all “0”, security function will be enabled.

**Parameters:**

**Input:**

Name	Type	Description
CardID	u8	assigned by DIP/ROTARY SW
password[5]	u16	Password, 5 words

**Note on password:**

If the password is all “0”, the security function is disabled.

- **dio3264 change password**

**Format :** u32 status = dio3264\_change\_password(u8 CardID,u16 Oldpassword[5],  
u16 password[5]);

**Purpose:** To replace old password with new password.

**Parameters:**

**Input:**

Name	Type	Description
CardID	u8	assigned by DIP/ROTARY SW
Oldpassword [5]	u16	The previous password
password[5]	u16	The new password to be set

- **dio3264 clear password**

**Format :** u32 status = dio3264\_clear\_password(u8 CardID,u16 password[5])

**Purpose:** To clear password, to set password to all "0", i.e. disable security function.

**Parameters:**

**Input:**

Name	Type	Description
CardID	u8	assigned by DIP/ROTARY SW
password[5]	u16	The password previous set

- **dio3264 unlock security**

**Format :** u32 status = dio3264\_unlock\_security(u8 CardID,u16 password[5])

**Purpose:** To unlock security function and enable the further operation of this card

**Parameters:**

**Input:**

Name	Type	Description
CardID	u8	assigned by DIP/ROTARY SW
password[5]	u16	The password previous set

- **dio3264 read security status**

**Format :** u32 status = dio3264\_read\_security\_status(u8 CardID,u8 \*lock\_status,  
u8 \*security\_enable );

**Purpose:** To read security status for checking if the card security function is unlocked.

**Parameters:**

**Input:**

Name	Type	Description
CardID	u8	assigned by DIP/ROTARY SW

**Output:**

Name	Type	Description
lock_status	u8	0: security unlocked 1: locked 2: dead lock (must return to original maker to unlock)
security_enable	u8	0: security function disabled 1: security function enabled

**Note on security status:**

The security should be unlocked before using any other function of the card, and any attempt to unlock with the wrong passwords more than 10 times will cause the card at dead lock status. Any further operation even with the correct password will not unlock the card. The only way is to send back to the card distributor or the original maker to unlock to virgin state.

## 7.5 Dll list

	<b>Function Name</b>	<b>Descriptive Name</b>
1	dio3264_initial( )	DIO3264 Initial
2	dio3264_close( )	DIO3264 Close
3	dio3264_info( )	get OS. assigned address
4	dio3264_get_device_handle ( )	Read device handle
5	dio3264_read_port( )	Read Port Data (byte)
6	dio3264_read_in_point( )	Read Input Point State(bit)
7	dio3264_enable_IRQ( )	Enable interrupt function
8	dio3264_disable_IRQ( )	Disable interrupt function
9	dio3264_set_IRQ_mask( )	Set interrupt mask
10	dio3264_IRQ_status( )	Read back irq status
11	dio3264_link_IRQ_process( )	Link interrupt service routine to driver
12	dio3264_set_password( )	Set software key
13	dio3264_change_password( )	Change software key
14	dio3264_clear_password( )	Clear software key
15	dio3264_unlock_security( )	Unlock software key
16	dio3264_read_security_status( )	Read software key status

## 8. Reference

### 8.1 DIO3264 I/O Port-Point table

<b>DIO3264 I/O Port table</b>								
<b>Bit Port</b>	<b>Bit 7</b>	<b>Bit 6</b>	<b>Bit 5</b>	<b>Bit 4</b>	<b>Bit 3</b>	<b>Bit 2</b>	<b>Bit 1</b>	<b>Bit 0</b>
<i>Port A(0)</i>	IN 7	IN 6	IN 5	IN 4	IN 3	IN 2	IN 1	IN 0
<i>Port B(1)</i>	IN 15	IN 14	IN 13	IN 12	IN 11	IN 10	IN 9	IN 8
<i>Port C(2)</i>	IN 23	IN 22	IN 21	IN 20	IN 19	IN 18	IN 17	IN 16
<i>Port D(3)</i>	IN 31	IN 30	IN 29	IN 28	IN 27	IN 26	IN 25	IN 24
<i>Port E(4)</i>	IN 39	IN 38	IN 37	IN 36	IN 35	IN 34	IN 33	IN 32
<i>Port F(5)</i>	IN 47	IN 46	IN 45	IN 44	IN 43	IN 42	IN 41	IN 40
<i>Port G(6)</i>	IN 55	IN 54	IN 53	IN 52	IN 51	IN 50	IN 49	IN 48
<i>Port H(7)</i>	IN 63	IN 62	IN 61	IN 60	IN 59	IN 58	IN 57	IN 56

## 9. DIO-3264 Error codes summary

### 9.1 DIO3264 Error codes table

<b>Error Code</b>	<b>Symbolic Name</b>	<b>Description</b>
<b>0</b>	JSDRV_NO_ERROR	No error.
<b>2</b>	JSDRV_INIT_ERROR	Driver initial error
<b>3</b>	JSDRV_UNLOCK_ERROR	Security unlock failure
<b>4</b>	JSDRV_LOCK_COUNTER_ERROR	Dead lock, unlock failure more than 10 times
<b>5</b>	SDRV_SET_SECURITY_ERROR	Password overwrite error
<b>100</b>	DEVICE_RW_ERROR	Device Read/Write error
<b>101</b>	JSDRV_NO_CARD	No DIO3264 card on the system.
<b>102</b>	JSDRV_DUPLICATE_ID	DIO3264 CardID duplicate error.
<b>300</b>	JSDIO_ID_ERROR	Function input parameter error. CardID setting error, CardID doesn't match the DIP SW setting
<b>301</b>	JSDIO_PORT_ERROR	Function input parameter error. Parameter out of range. ( In port > 7 )
<b>302</b>	JSDIO_IN_POINT_ERROR	Function input parameter error. Parameter out of range. ( point > 63)