
PCI-D64HU

User's Manual

Warranty

All products manufactured by ICP DAS are warranted against defective materials for a period of one year from the date of delivery to the original purchaser.

Warning

ICP DAS assume no liability for damages consequent to the use of this product. ICP DAS reserves the right to change this manual at any time without notice. The information furnished by ICP DAS is believed to be accurate and reliable. However, no responsibility is assumed by ICP DAS for its use, nor for any infringements of patents or other rights of third parties resulting from its use.

Copyright

Copyright 2009 by ICP DAS. All rights are reserved.

Trademark

The names used for identification only may be registered trademarks of their respective companies.

Table of Contents

1. Introduction	3
1.1. General Description	3
1.2. Features.....	4
1.3. Block Diagram.....	5
1.4. Specifications	6
1.5. Product Check List	7
2. Hardware Configuration.....	8
2.1. Board Layout.....	8
2.2. Switch Setting.....	9
2.3. Pin Assignment	10
2.4. Operation Theory.....	11
2.5. Timing Characteristic.....	14

1. Introduction

1.1. General Description

The PCI-D64HU card provides 40 MB/s High-Speed 32-CH Digital Input and 32-CH Digital Output. PCI-D64HU is a high-speed digital I/O card consisting of 32 digital input channels and 32 digital output channels. High-performance designs make this card perfect for high-speed data transfer and pattern generation applications.

The PCI-D64HU has the Card ID switch on board. Users can set Card ID on a board and recognize the board by the ID via software when using two or more PCI-D64HU cards in one computer.

The PCI-D64HU performs high-speed data transfer by bus-mastering DMA via 32-bit PCI bus. The maximum data transfer rate can be up to 40 MB per second. Several digital I/O transfer modes are supported, such as direct programmed I/O control, timer pacer control, external clock mode and handshaking mode. It is a reliable and cost-effective connection interface that works on your computer system to control high-speed peripherals.

1.2. Features

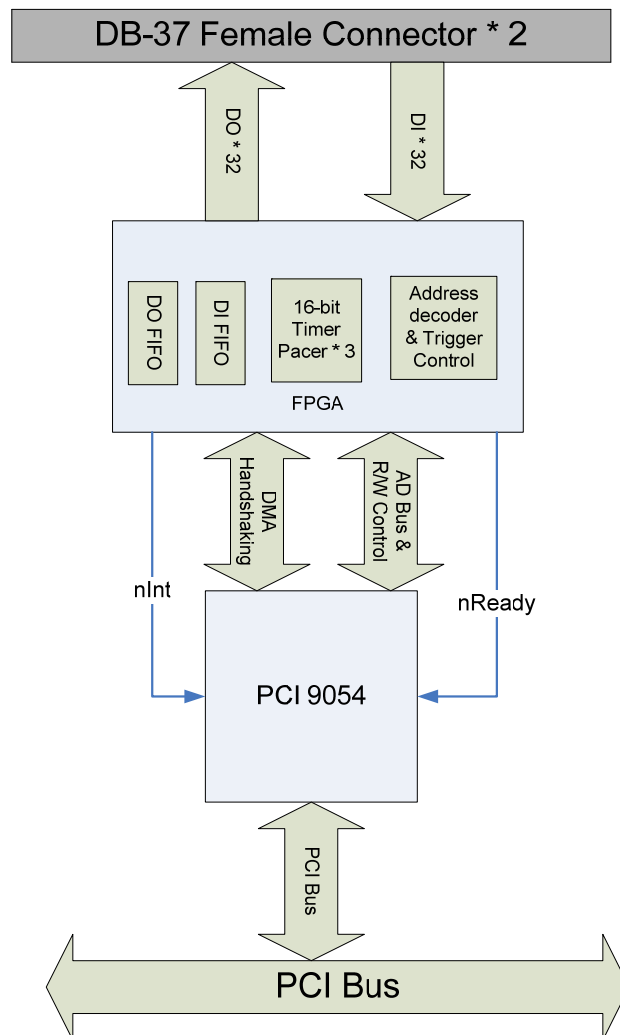
The following is a list of general features for the PCI-D64HU. Check section 1.5 for more details.

- Support 32-bit, 33 MHz Universal PCI bus
- 32-CH 5V TTL Digital Inputs and 32-CH 5V TTL Digital Outputs
- 2-CH Bus Mastering Scatter/Gather DMA
- Data Transfer Rate up to 40 MB/s for Each Channel
- Support 4 Data Transfer Modes
 - Direct Program Control Mode
 - Internal Timer Pacer Mode
 - External Clock Mode (DI Only)
 - Handshaking Mode
- On Board 1 k/2 k DWORD FIFO for DI/DO Respectively
DO FIFO Support Ring Buffer Mode -- No Bus Loading in Repetitive Pattern
- Card ID function

1.3. Generation Application

- Programmable Input Digital Filter for All Input Signals Including Handshaking and Trigger Signals

1.4. Block Diagram



1.5. Specifications

Model Name	PCI-D64HU
Digital Input	
Channels	32
Compatibility	5 V/TTL
Input Voltage	Logic 0: 0.8 V max. Logic 1: 2.0 V min.
Handshaking Signals	I_REQ input , I_ACK output , I_TRG input
Digital Output	
Channels	32
Compatibility	5 V/TTL
Output Voltage	Logic 0: 0.55 V max. Logic 1: 2.0 V min.
Output Capability	Sink: 64 mA @ 0.55 V Source: -32 mA @ 2.0 V
Handshaking Signals	O_REQ output, O_ACK input, O_TRG output
Transfer Speed	40 MB/sec for DI and DO simultaneously (max.)
Programmable Digital Filter	
Applicable Signals	All input signals (including 32 DI signals, I_REQ, I_TRG and O_ACK)
Max. removable noise width	(1 ~ 127) x 25 ns
Programmable Pulse Extender	
Applicable Signals	O_REQ
Pulse width	(2 ~ 256) x 25 ns
Timer/Counter	
Channels	3
Resolution	16-bit
Input Frequency	2.5 ~ 20 MHz
Timer 0	Clock source of DI
Timer 1	Clock source of DO
Timer 2	Base clock of Timer 0 and Timer 1
Interrupt	
Sources	O_ACK, I_REQ, Timer 0, Timer 1 and Timer 2
On Board FIFO	
DI / DO	1 k DWORD (32-bit) 2 k DWORD (32-bit)
Size in Ring Buffer Mode	2 ~ 2 k DWORD (32-bit), DO only
General	
Bus Type	Universal PCI, 32-bit, 33 MHz
Card ID	Yes (4-bit)
I/O Connector	Female DB37 x 1 40-pin Box header x 1
Dimensions (L x W x D)	120 mm x 105 mm x 22 mm
Power Consumption	200 mA @ +5 V typical (output no load)
Operating Temperature	0 ~ 60 °C
Storage Temperature	-20 ~ 70 °C
Humidity	5 ~ 85% RH, non-condensing

1.6. Product Check List

In addition to this manual, the package includes the following items:

- One PCI-D64HU card
- One ICP-DAS software CD-ROM
- One Quick Start Guide

It is recommended to read the Quick Start Guide first. The following important information will be given in the Quick Start Guide:

1. Where you can find the software driver & utility
2. How to install software & utility
3. Location of the diagnostic program

1.7. Ordering Information

Ordering Information:

- PCI-D64HU: 40 MB/s High-speed 32-CH DI and 32-CH DO Universal PCI DIO Card (RoHS). Includes one CA-4037W cable and two CA-4002 D-Sub connectors

Accessories:

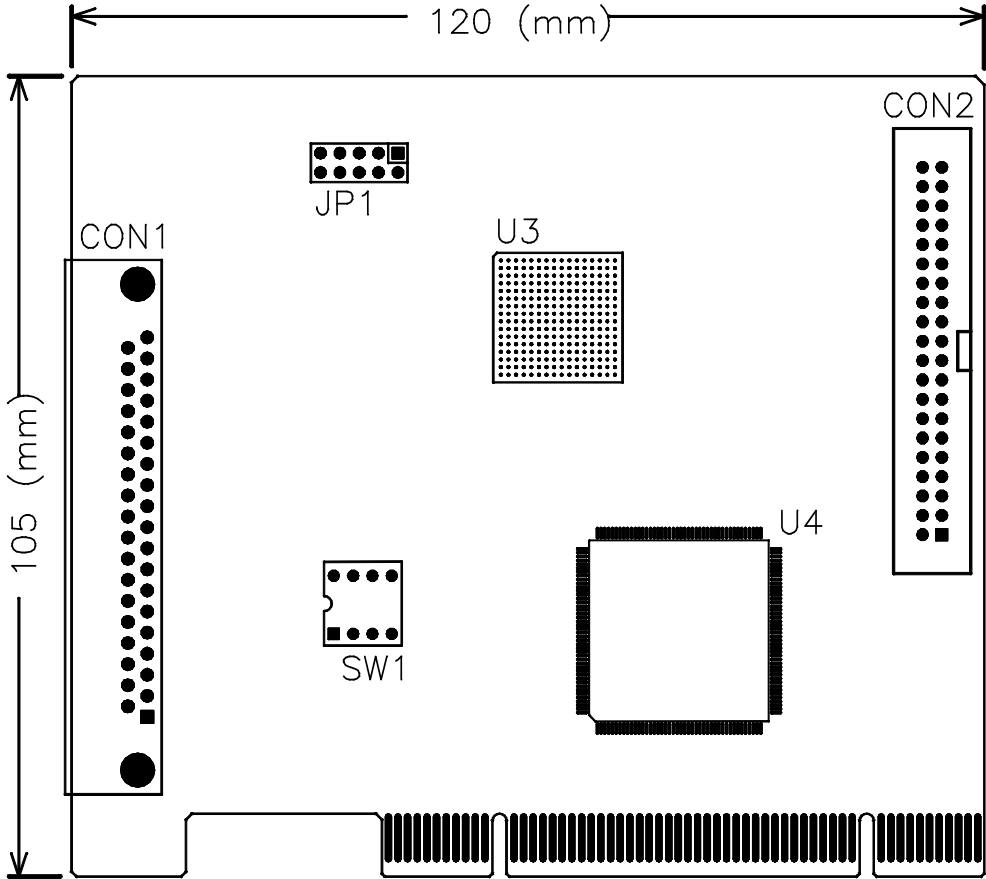
- CA-3710: DB-37 Male-Male D-sub cable 1M (45°)
- CA-3710DM: DB-37 Male-Male D-sub cable 1M (180°)
- DN-37: DIN-Rail Mounting 37-pin Connector (Pitch=5.08mm)
- DN-37-381: DIN-Rail Mounting 37-pin Connector (Pitch=3.81mm)
- DB-37: Directly connect signal to D-Sub 37-pin connector

Attention!

If any of these items are missing or damaged, please contact your local field agent. Save the shipping materials and carton in case you want to ship or store the product in the future.

2. Hardware Configuration

2.1. Board Layout

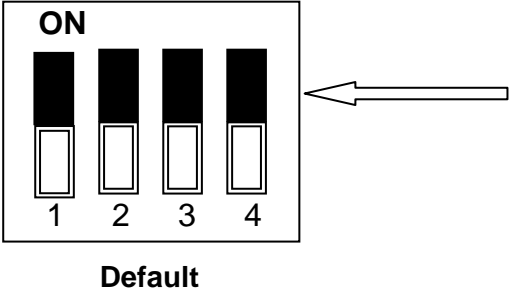


Item	Description
CON1	I/O connector (37-pin D-Sub female)
CON2	I/O connector (40-pin Box Header)
SW1	Card ID switch
JP1	Factory reserved

2.2. Card ID Switch Setting

SW1: Card ID Switch

The SW1 switch is used to set the card ID. The value is from 0 to 15. Please refer to table below for details.

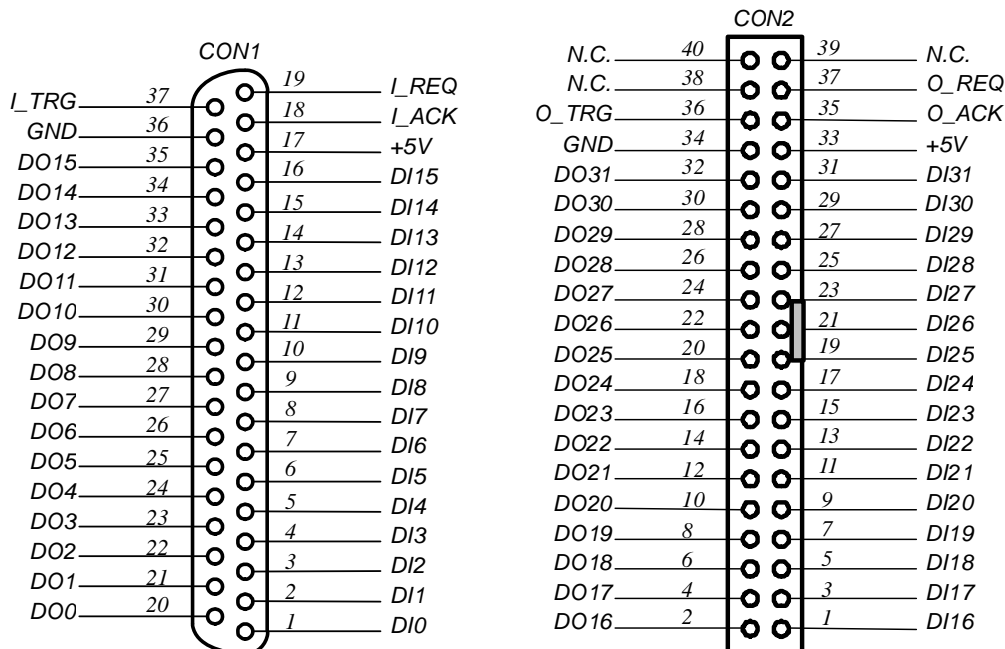


Card ID	Switch Setting (On = 1)			
	1	2	3	4
0(Default)	0	0	0	0
1	1	0	0	0
2	0	1	0	0
3	1	1	0	0
4	0	0	1	0
5	1	0	1	0
6	0	1	1	0
7	1	1	1	0
8	0	0	0	1
9	1	0	0	1
10	0	1	0	1
11	1	1	0	1
12	0	0	1	1
13	1	0	1	1
14	0	1	1	1
15	1	1	1	1

2.3. Pin Assignment

The PCI-D64HU has one 37-pin D-Sub connector (CON1) and one 40-pin pin header (CON2). Please refer figure below for the pin assignment of CON1 and CON2

Signal Name	Description	Direction
DIx	Digital input channel x	Input
DOx	Digital output channel x	Output
GND	Ground of all signals	Power
+5V	+5V power output (Max. 400 mA per pin)	Power
I_TRG	Trigger input to start DI sampling	Input
I_REQ	Request input for DI handshaking	Input
I_ACK	Acknowledge output for DI handshaking	Output
O_TRG	Trigger output controlled by software	Output
O_REQ	Request output for DO handshaking	Output
O_ACK	Acknowledge input for DO handshaking	Input



2.4. Operation Theory

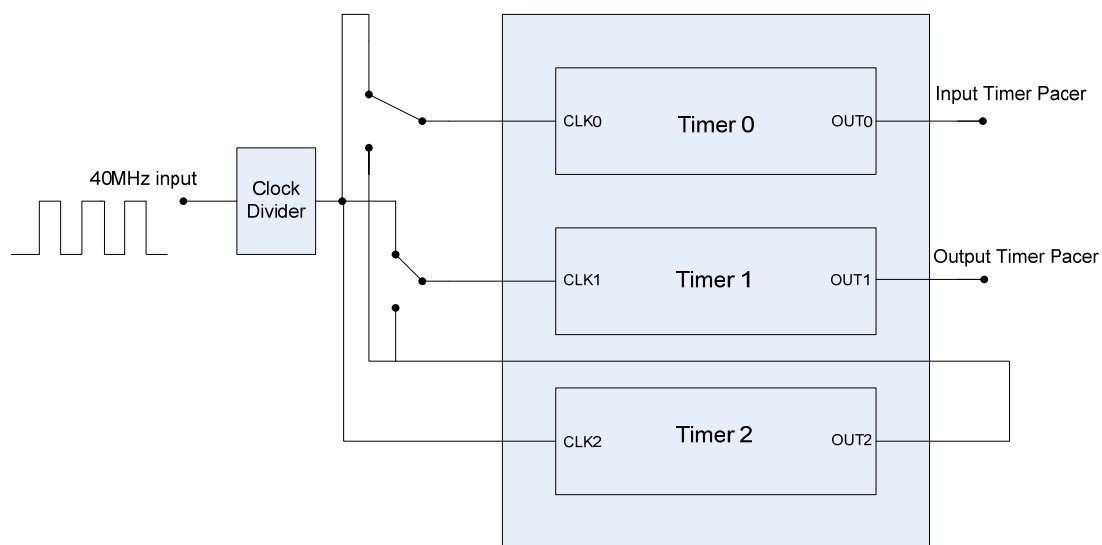
The PCI-D64HU support 4 data transfer modes, they are direct programmed I/O control, timer pacer control, external clock mode and handshaking mode. This chapter describe the detailed operation of these 4 data transfer modes.

Direct Program Control Mode

The status of digital inputs and digital outputs can be directly accessed by I/O port access. The I/O port address is assigned by system BIOS, please refer to the function reference manual for more detailed description.

Internal Timer Pacer Mode

There are three 16-bit timers on board. Timer#0 is for DI update and Timer#1 is for DO update. Both Timer#0 and Timer#1 can be optionally cascaded with Timer#2 for slower timer pacer generation. The base clock of all timers can be programmed from 20 MHz to 2.5 MHz ($40 \text{ MHz} / n$; where $n = 2 \sim 16$).



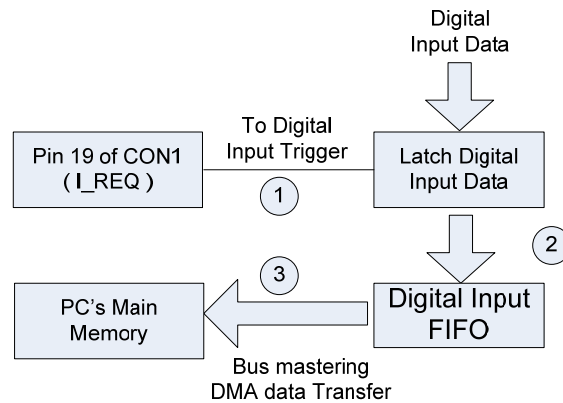
For digital input, the input data will be saved into DI FIFO after a timer pacer pulse is generated. When the DI FIFO is not empty, the saved data will be automatically transferred to the main memory of computer system by the bus mastering DMA controller.

For digital output, the state of output pins will be updated by the data in DO FIFO after a timer pacer pulse is generated. When the DO FIFO is not full, data in the main memory will be automatically transferred to the DO FIFO by the bus mastering DMA controller.

External Clock Mode (DI Only)

The digital input is clocked by external strobe, which is from Pin 19 (I_REQ) of CON1.

The input data will be saved into DI FIFO after a strobe is generated. When the DI FIFO is not empty, the saved data will be automatically transferred to the main memory of computer system by the bus mastering DMA controller.

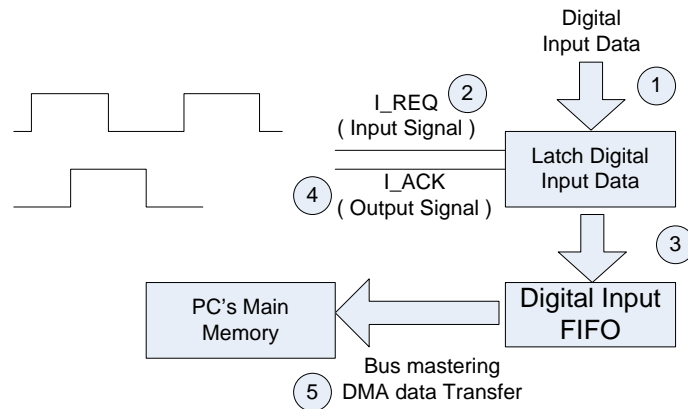


1. Digital Input data is ready and an I_REQ signal is generated by external device.
2. Digital input data is saved to FIFO.
3. If the FIFO is not empty and PCI bus is not occupied, the data will be transferred to main memory.

Handshaking Mode

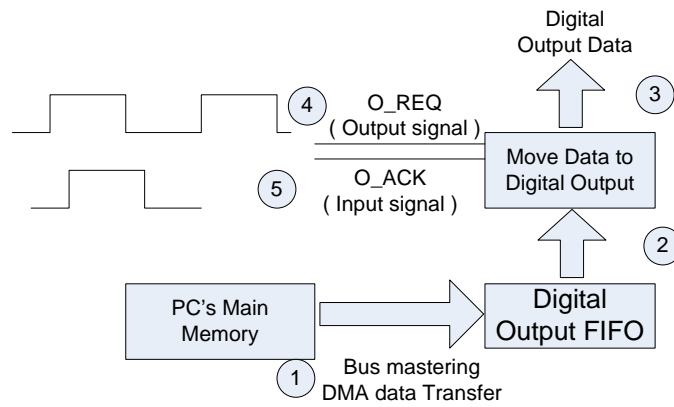
The PCI-D64HU also supports handshaking data transfer mode. The data transfer rate is controlled by REQ and ACK signals to guarantee no data loss.

The operation of DI Handshaking



1. Digital Input data is ready.
2. An I_REQ signal is generated by external device.
3. Digital input data is saved to FIFO.
4. An I_ACK signal is generated and sent to the external device.
5. If the FIFO is not empty and PCI bus is not occupied, the data will be transferred to main memory.

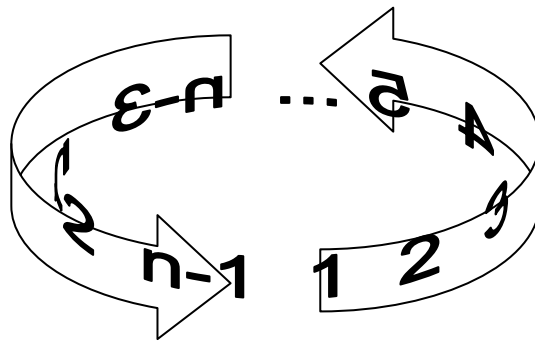
The operation of DO Handshaking



1. Digital output data is moved from PC's memory to DO FIFO by bus mastering DMA data transfer.
 2. Move output data from FIFO to digital output circuit.
 3. Output data is ready.
 4. An O_REQ signal is generated and sent to the external device.
 5. After an O_ACK is captured, steps 2-5 will be repeated.
- ** If the FIFO is not full, the output data is moved form PC's main memory to FIFO automatically.

DO Ring Buffer Mode

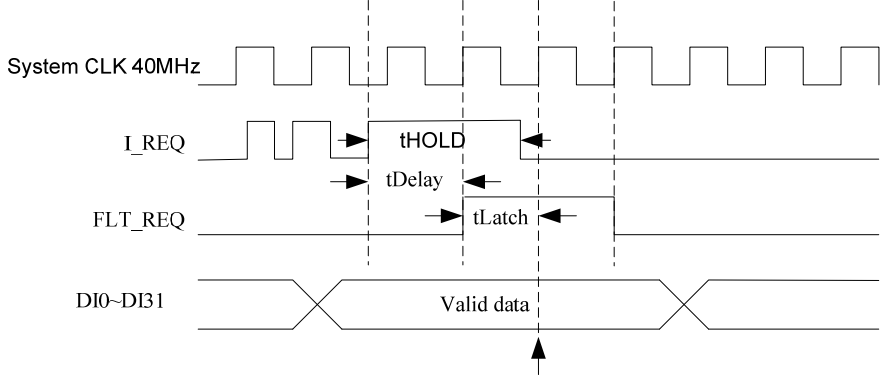
The ring buffer is managed in hardware level and the size of the ring buffer can be set by user. When the DO FIFO is set as ring buffer mode, the last buffer of the DO buffer will be chained with the first buffer. No bus loading is required which makes PCI-D64HU perfect for repetitive pattern generation application.



2.5. Timing Characteristic

Characteristic of Input Digital Filter

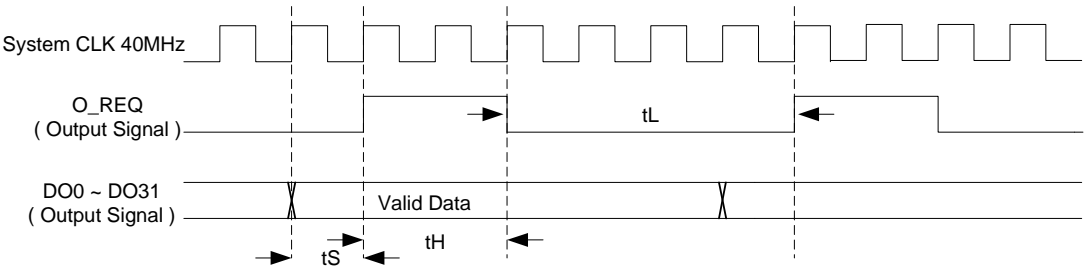
The digital filter is applicable for all DI signals, IREQ, ITRG and OACK. The maximum removable noise width can be programmed from (1~127) x 25ns.



$n * 25ns < tDelay \leq (n+1) * 25ns$	$tHold \geq (n+1) * 25ns$	$tLatch = 25ns$
---------------------------------------	---------------------------	-----------------

Where n=1~127

Characteristic of Output Pulse Extender in Timer Pacer Mode

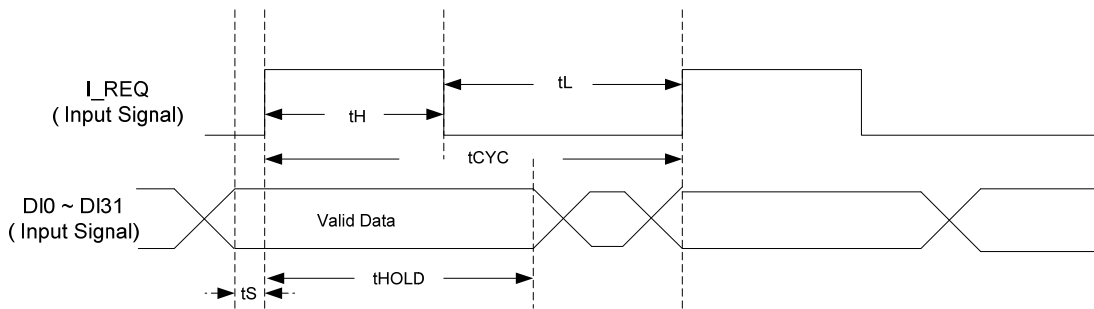


$25ns \leq tS \leq (2^{n+1}) * 25ns$	$tH = (2^{n+1}) * 25ns$
$tL \geq (2^{n+1}) * 25ns$	

Notes: n = 0~7

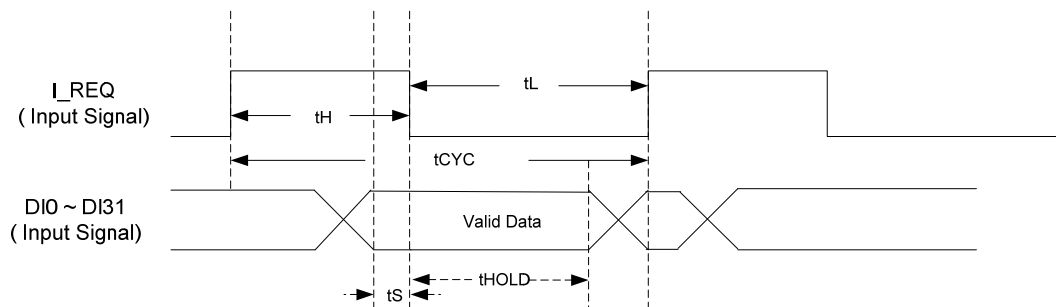
Characteristic of IREQ in External Clock Mode

IREQ Rising Edge Trigger:



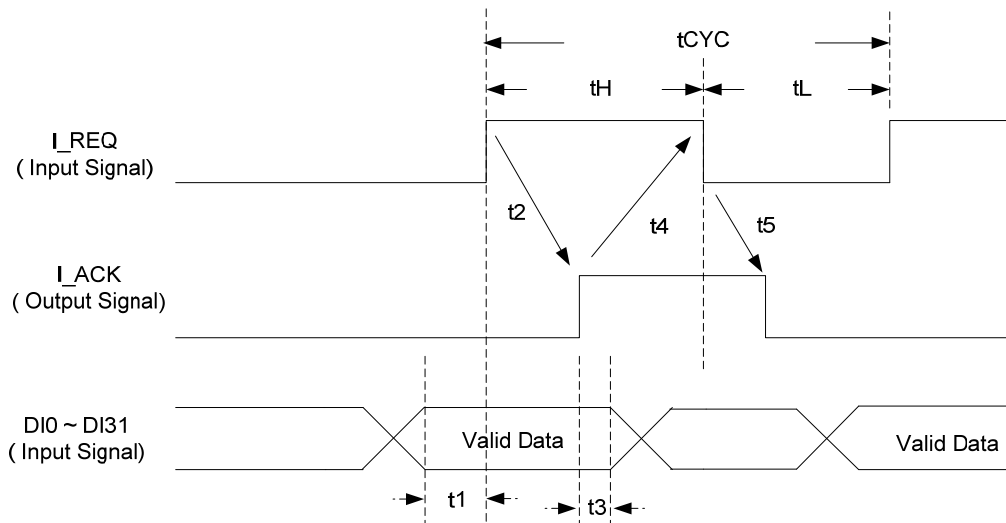
$tH \geq 25ns$	$tL \geq 25ns$	$tCYC \geq 100ns$
$tS \geq 1ns$	$tHOLD \geq 25ns$	

IREQ Falling Edge Trigger:



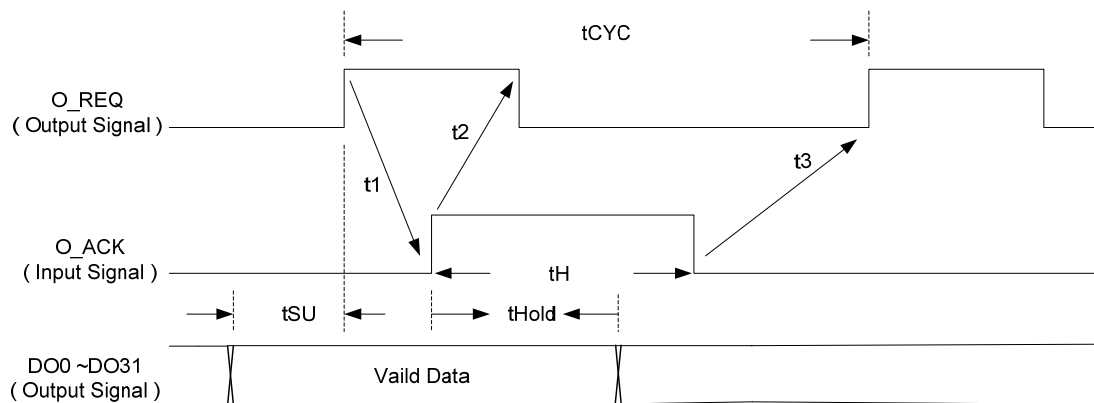
$tH \geq 25ns$	$tL \geq 25ns$	$tCYC \geq 100ns$
$tS \geq 1ns$	$tHOLD \geq 25ns$	

Characteristic of DI Handshaking Mode



$t_1 \geq 1\text{ns}$	$0\text{ns} < t_2 < 29\text{ns}$	$t_3 > 0\text{ns}$	$t_4 > 0\text{ns}$
$0\text{ns} < t_5 < 29\text{ns}$	$t_H \geq 25\text{ns}$	$t_L \geq 25\text{ns}$	$t_{CYC} \geq 100\text{ns}$

Characteristic of DO Handshaking Mode



$t_1 > 0\text{ns}$	$0\text{ns} < t_2 < 36\text{ns}$	$0\text{ns} < t_3 < 36\text{ns}$	$t_H \geq 25\text{ns}$
$t_{SU} = 25\text{ns}$	$25\text{ns} < t_{Hold} < 61\text{ns}$	$t_{CYC} \geq 100\text{ns}$	

PCI-D64HU Function Reference

(Version 1.0)



ICP DAS CO., LTD.

Warranty

All products manufactured by ICPDAS Inc. are warranted against defective materials for a period of one year from the date of delivery to the original purchaser.

Warning

ICPDAS Inc. assumes no liability for damages consequent to the use of this product. ICPDAS Inc. reserves the right to change this manual at any time without notice. The information furnished by ICPDAS Inc. is believed to be accurate and reliable. However, no responsibility is assumed by ICPDAS Inc. for its use, or for any infringements of patents or other rights of third parties resulting from its use.

Trademark

The names used for identification only maybe registered trademarks of their respective companies.

License

The user can use, modify and backup this software on a single machine. The user may not reproduce, transfer or distribute this software, or any copy, in whole or in part.

Contents

INTRODUCTION	5
SYSTEM INITIALIZATION	6
2.1 <i>d64h_scan</i>	6
2.2 <i>d64h_get_cardinfo</i>	8
2.3 <i>d64h_open</i>	9
2.4 <i>d64h_close</i>	10
2.5 <i>d64h_di_available_memory</i>	11
2.6 <i>d64h_do_available_memory</i>	12
2.7 <i>d64h_di_buffer_get</i>	13
2.8 <i>d64h_do_buffer_get</i>	14
DIGITAL INPUT/OUTPUT CONFIGURATION	15
3.1 <i>d64h_di_config</i>	15
3.2 <i>d64h_do_config</i>	17
DIGITAL INPUT FUNCTIONS.....	18
4.1 <i>d64h_di_readport</i>	18
4.2 <i>d64h_di_readline</i>	19
4.3 <i>d64h_di_async_dblbuf_mode</i>	20
4.4 <i>d64h_di_async_dblbuf_halfready</i>	21
4.5 <i>d64h_di_async_dblbuf_transfer</i>	22
4.6 <i>d64h_di_async_check</i>	23
4.7 <i>d64h_di_async_clear</i>	24
4.8 <i>d64h_continue_readport</i>	25
4.9 <i>d64h_conti_di_status</i>	27
DIGITAL OUTPUT FUNCTIONS.....	28
5.1 <i>d64h_do_writeport</i>	28
5.2 <i>d64h_do_writeline</i>	29
5.3 <i>d64h_do_readport</i>	30
5.4 <i>d64h_do_readline</i>	31
5.5 <i>d64h_do_async_dblbuf_mode</i>	32
5.6 <i>d64h_do_async_dblbuf_halfready</i>	33
5.7 <i>d64h_do_async_dblbuf_transfer</i>	34
5.8 <i>d64h_do_async_check</i>	35

5.9	<i>d64h_do_async_clear</i>	36
5.10	<i>d64h_continue_writeport</i>	37
5.11	<i>d64h_continue_pattern_write</i>	39
5.12	<i>d64h_conti_do_status</i>	41
5.13	<i>d64h_do_ext_trigline_write</i>	42
DIGITAL FILTER FUNCTIONS		43
6.1	<i>d64h_DI_filter_set</i>	43
6.2	<i>d64h_IREQ_filter_set</i>	45
6.3	<i>d64h_OACK_filter_set</i>	46
6.4	<i>d64h_ITRG_filter_set</i>	47
6.5	<i>d64h_OREQ_width_set</i>	48
EVENT NOTIFICATION FUNCTIONS		50
7.1	<i>d64h_di_event_callback</i>	50
7.1	<i>d64h_do_event_callback</i>	52
ERROR CODE		54

Introduction

This software package is dedicated to PCI-D64HU high-speed digital input/output card. It includes the WDM (Windows Driver Model) driver and ANSI-C Library for Windows 2000/XP.

One unique Card ID will be referred by each function in this Library. The Card ID is configured with on-board DIP-Switch, and helps to identify multiple PCI-D64HU cards in your system. In other words, you no longer worry about the order that Operating System scans PCI-D64HU cards; the only thing you must take care is the correct relationship between the terminal-boards and PCI-D64HU cards.

There are samples that are provided for Microsoft® Visual Studio 6.0 (VC and VB) to demonstrate the functions of PCI-D64HU Library.

This documentation provides the detailed information of PCI-D64HU APIs, including the function-declaration, definitions of both parameters and return codes. The APIs will be cataloged and described in the following chapters:

- CHAPTER 2 – System Initialization
- CHAPTER 3 – Digital Input/Output Configuration
- CHAPTER 4 – Digital Input Functions
- CHAPTER 5 – Digital Output Functions
- CHAPTER 6 – Digital Filter Functions
- CHAPTER 7 – Event Notification Functions

System Initialization

The functions in this chapter provide the interface to Operating-System. By calling these functions, your applications can scan all active PCI-D64HU cards in your system, and get the specific Card-ID that is configured with the on-board Dip-Switch. Open the card before calling other functions in PCI-D64HU Library.

2.1 d64h_scan

VC6

short d64h_scan(short* pCardNum, BYTE* pAvailCards = NULL)

VB6

d64h_scan(ByRef pCardNum As Integer, Optional pAvailCards As Byte = 0) As Integer

Description:

This function scans all active PCI-D64HU cards in your system. The pCardNum saves the numbers of active PCI-D64HU cards. The optional user-provided Array, pAvailCards, indicates the presence of active PCI-D64HU card. (1: present, 0: absent)

Parameters:

pCardNum: The pointer to the memory that stores the numbers of active PCI-D64HU cards.

pAvailCard: The address of user-provided **BYTE**-Array. Based on the Card ID, each element indicates the presence of active PCI-D64HU card. The user must prepare one **BYTE**-Array with **D64H_MaxCards** elements.

For instance, there are three active PCI-D64HU cards with Card ID 3, 5 and 7. The content of pAvailCard Array will be

```
{ 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0 }
```

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_NO_CARD_FOUND: There is no active card available in your system.

ERROR_CARD_ID_DUPLICATED: There are multiple cards that are assigned the same Card ID, please check the settings of on-board Dip-Switch.

ERROR_ACCESS_VIOLATION_DATA_COPY: Some system exception occurs while copying memory, please check the pointer-type parameter you assign to this function.

2.2 d64h_get_cardinfo

VC6

short d64h_get_cardinfo(int ScannedIndex, BYTE* pCardID)

VB6

d64h_get_cardinfo(ByVal ScannedIndex As Integer, ByRef pCardID As Byte) As Integer

Description:

This function returns the Card ID based on the scanned-index. This routine will get the Card ID configured with on-board Dip-Switch.

Parameters:

ScannedIndex: The index that the active PCI-D64HU is scanned. This index begins from 0, and is less than the active PCI-D64HU cards.

pCardID: The pointer to the memory that stores the specific Card ID.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_NO_CARD_FOUND: There is no active card available in your system.

ERROR_INVALID_SCANNED_INDEX: Indicates the **ScannedIndex** parameter is larger than the numbers of active PCI-D64HU cards.

ERROR_ACCESS_VIOLATION_DATA_COPY: Some system exception occurs while copying memory, please check the pointer-type parameter you assign to this function.

2.3 d64h_open

VC6

short d64h_open(BYTE bCardID)

VB6

d64h_open(ByVal bCardID As Byte) As Integer

Description:

This function opens the device node of PCI-D64HU based on the specific Card ID. If this function returns successfully, the process that calls this function will own that specific device until d64h_close() is called. The device node of PCI-D64HU is ought to be owned before calling other functions. It's recommended to call d64h_scan() and d64h_get_cardinfo() to get the Card ID.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_DEVICE_OPEN: Fail to open the device-node of PCI-D64HU. Please make sure no other process owns that PCI-D64HU card.

ERROR_EVENT_CREATE_FAILED: Fail to create the related events for digital input/output operations.

ERROR_MEMORY_MAP: Indicates the Memory-Mapping is failed, please check the event logs in Event Viewer.

2.4 d64h_close

VC6

short d64h_close(BYTE bCardID)

VB6

d64h_close(ByVal bCardID As Byte) As Integer

Description:

This function closes the device node of PCI-D64HU based on the specific Card ID. After calling this function, the PCI-D64HU card will be released, and other process can open it.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no opened PCI-D64HU card with assigned Card ID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_INVALID_MAPPED_ADDRESS: Indicates the mapped address is invalid.

ERROR_DI_EVENT_DETACH: Fail to detach the digital-input related event.

ERROR_DO_EVENT_DETACH: Fail to detach the digital-output related event.

ERROR_DEVICE_CLOSE: Fail to close the device-node of PCI-D64HU.

ERROR_MEMORY_UNMAP: Indicates the Memory-Un-mapping is failed, please check the event logs in Event Viewer.

2.5 d64h_di_available_memory

VC6

short d64h_di_available_memory(BYTE bCardID, U32 *pMemSize)

VB6

d64h_di_available_memory (ByVal bCardID As Byte, ByRef pMemSize As Long) As Integer

Description:

This function gets the size of DI buffer that is allocated in driver. The unit of allocated buffer is reported in kilo-bytes.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

pMemSize: The pointer to the size of DI buffer, in kilobytes (KB).

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no opened PCI-D64HU card with assigned Card ID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_INVALID_MAPPED_ADDRESS: Indicates the mapped address is invalid.

ERROR_ACCESS_VIOLATION_DATA_COPY: Some system exception occurs while copying memory, please check the pointer-type parameter you assign to this function.

2.6 d64h_do_available_memory

VC6

short d64h_do_available_memory(BYTE bCardID, U32 *pMemSize)

VB6

d64h_do_available_memory (ByVal bCardID As Byte, ByRef pMemSize As Long) As Integer

Description:

This function gets the size of DO buffer that is allocated in driver. The unit of allocated buffer is reported in kilo-bytes.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

pMemSize: The pointer to the size of DO buffer, in kilobytes (KB)

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no opened PCI-D64HU card with assigned Card ID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_INVALID_MAPPED_ADDRESS: Indicates the mapped address is invalid.

ERROR_ACCESS_VIOLATION_DATA_COPY: Some system exception occurs while copying memory, please check the pointer-type parameter you assign to this function.

2.7 d64h_di_buffer_get

VC6

```
short d64h_di_buffer_get(BYTE bCardID, U32 *pMemSize, PVOID* pLowBufAddr, PVOID*  
pHighBufAddr )
```

Description:

This function gets the size of DI buffer and the virtual address to access this DI buffer. These buffer addresses help programmer to get the DI acquisition data directly.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

pMemSize: The pointer to the size of DI buffer, in kilobytes (KB)

pLowBufAddr: The pointer to the address of low-part buffer.

pHighBufAddr: The pointer to the address of high-part buffer.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no opened PCI-D64HU card with assigned Card ID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_INVALID_MAPPED_ADDRESS: Indicates the mapped address is invalid.

ERROR_ACCESS_VIOLATION_DATA_COPY: Some system exception occurs while copying memory, please check the pointer-type parameter you assign to this function.

2.8 d64h_do_buffer_get

VC6

```
short d64h_do_buffer_get(BYTE bCardID, U32 *pMemSize, PVOID* pLowBufAddr, PVOID*  
pHighBufAddr )
```

Description:

This function gets the size of DO buffer and the virtual address to access this DO buffer. These buffer addresses help programmer to update the DO output data directly.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

pMemSize: The pointer to the size of DO buffer, in kilobytes (KB)

pLowBufAddr: The pointer to the address of low-part buffer.

pHighBufAddr: The pointer to the address of high-part buffer.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no opened PCI-D64HU card with assigned Card ID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_INVALID_MAPPED_ADDRESS: Indicates the mapped address is invalid.

ERROR_ACCESS_VIOLATION_DATA_COPY: Some system exception occurs while copying memory, please check the pointer-type parameter you assign to this function.

Digital Input/Output Configuration

The functions in this chapter configure the operation mode of digital input/output. These configurations will be applied to next continuous digital input/output functions, say `d64h_continue_readport()`, `d64h_continue_writeport()` and `d64h_continue_pattern_write()`. And these specific settings will be reset after calling `d64h_di_async_clear()` and `d64h_do_async_clear()`.

When the trigger-source is configured as `TRIG_SOURCE_INT_PACER`, the desired sampling-rate will be generated with the internal clock-source (20MHz) and dividers. Therefore, the actual sampling-rate is not exactly equal to desired sampling-rate. For instance, for the 1024000Hz desired sampling-rate, data acquisition is performed with the 1052631.8Hz frequency actually.

3.1 d64h_di_config

VC6

short d64h_di_config(BYTE bCardID, U16 wTrigSource, U16 wExtTrigEnable, U16 wTrigPolarity, U16 wl_REQ_Polarity)

VB6

d64h_di_config(ByVal bCardID As Byte, ByVal wTrigSource As Integer, ByVal wExtTrigEnable As Integer, ByVal wTrigPolarity As Integer, ByVal wl_REQ_Polarity As Integer) As Integer

Description:

This function configures the functionality of the following continuous digital-input.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

wTrigSource: `TRIG_SOURCE_INT_PACER`, `TRIG_SOURCE_EXT_STROBE` or `TRIG_SOURCE_HANDSHAKE`.

wExtTrigEnable: `DI_WAITING` or `DI_NOWAITING`.

wTrigPolarity: DI_TRIG_RISING or DI_TRIG_FALLING. This parameter is required only when wExtTrigEnable is set as DI_WAITING.

wI_REQ_Polarity: IREQ_RISING or IREQ_FALLING. This parameter is required only when wTrigSource is set as TRIG_SOURCE_EXT_STROBE or TRIG_SOURCE_HANDSHAKE.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_INVALID_MAPPED_ADDRESS: Indicates the mapped address is invalid.

ERROR_INVALID_TRIGGER_SOURCE: Indicates the invalid trigger-source is assigned to parameter **wTrigSource**.

ERROR_INVALID_TRIGGER_ENABLE: Neither DI_WAITING nor DI_NOWAITING is assigned to parameter **wExtTrigEnable**.

ERROR_INVALID_TRIGGER_POLARITY: Neither DI_TRIG_RISING nor DI_TRIG_FALLING is assigned to parameter **wTrigPolarity**.

ERROR_INVALID_IREQ_POLARITY: Neither IREQ_RISING nor IREQ_FALLING is assigned to parameter **wI_REG_Polarity**.

ERROR_DI_CONFIG: Cannot configure the digital-input operation, please call GetLastError() for further system information.

3.2 d64h_do_config

VC6

short d64h_do_config(BYTE bCardID, U16 wTrigSource, U16 wO_REQ_Enable, U16 wOutTrigHigh)

VB6

d64h_do_config(ByVal bCardID As Byte, ByVal wTrigSource As Integer, ByVal wO_REQ_Enable As Integer, ByVal wOutTrigHigh As Integer) As Integer

Description:

This function configures the functionality of the following continuous digital-output.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

wTrigSource: TRIG_SOURCE_INT_PACER or TRIG_SOURCE_HANDSHAKE.

wO_REQ_Enable: OREQ_ENABLE or OREQ_DISABLE.

wOutTrigHigh: OTRIG_HIGH or OTRIG_LOW. This parameter is required only when wO_REQ_Enable is set as OREQ_ENABLE.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_INVALID_MAPPED_ADDRESS: Indicates the mapped address is invalid.

ERROR_INVALID_TRIGGER_SOURCE: Neither TRIG_SOURCE_INT_PACER nor TRIG_SOURCE_HANDSHAKE is assigned to parameter **wTrigSource**.

ERROR_INVALID_OREQ_ENABLE: Neither OREQ_ENABLE nor OREQ_DISABLE is assigned to parameter **wO_REG_Enable**.

ERROR_INVALID_OTRIG_LEVEL: Neither DI_TRIG_RISING nor DI_TRIG_FALLING is assigned to parameter **wTigPolarity**.

ERROR_INVALID_IREQ_POLARITY: Neither OTRIG_HIGH nor OTRIG_LOW is assigned to parameter **wOutTrigHigh**.

ERROR_DO_CONFIG: Cannot configure the digital-input operation, please call GetLastError() for further system information.

Digital Input Functions

The functions in this chapter are provided for digital-input operations.

4.1 d64h_di_readport

VC6

short d64h_di_readport(BYTE bCardID, U32* pValue)

VB6

d64h_di_readport(ByVal bCardID As Byte, ByRef pValue As Long) As Integer

Description:

This function reads all digital-input, and stores the data into one 32-bit variable.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

pValue: The pointer to the 32-bit variable that stores all digital-input.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_ACCESS_VIOLATION_DATA_COPY: Some system exception occurs while copying memory, please check the pointer-type parameter you assign to this function.

ERROR_DI_PIO_READ: Cannot read the digital-input port, please call GetLastError() for further system information.

4.2 d64h_di_readline

VC6

short d64h_di_readline(BYTE bCardID, U16 wLine, U16* pValue)

VB6

d64h_di_readline(ByVal bCardID As Byte, ByVal wLine As Integer, ByRef pValue As Integer)
As Integer

Description:

This function reads specific digital-input line.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

wLine: The parameter indicates the specific line to be input.

pValue: The pointer to the 16-bit variable that stores the specific digital-input line.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_INVALID_PORT_LINE: Indicates the invalid setting is assigned to parameter **wLine**.

ERROR_ACCESS_VIOLATION_DATA_COPY: Some system exception occurs while copying memory, please check the pointer-type parameter you assign to this function.

ERROR_DI_PIO_READ: Cannot read the specific digital-input line, please call GetLastError() for further system information.

4.3 d64h_di_async_dblbuf_mode

VC6

short d64h_di_async_dblbuf_mode (BYTE bCardID, U16 wDbIBufEnable)

VB6

d64h_di_async_dblbuf_mode (ByVal bCardID As Byte, ByVal wDbIBufEnable As Integer) As Integer

Description:

This function configures the operation mode of digital-input acquisition.

Parameters:

bCardID: the specific Card ID that is configured with the on-board Dip-Switch.

wDbIBufEnable: DOUBLE_BUFFER_MODE_ENABLE or DOUBLE_BUFFER_MODE_DISABLE.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_INVALID_MAPPED_ADDRESS: Indicates the mapped address is invalid.

ERROR_INVALID_DOUBLE_BUFFER_MODE: Neither DOUBLE_BUFFER_MODE_ENABLE nor DOUBLE_BUFFER_MODE_DISABLE is assigned to parameter **wDbIBufEnable**.

ERROR_DOUBLE_BUFFER_MODE: Cannot configure the digital-input acquisition as double-buffer mode.

4.4 d64h_di_async_dblbuf_halfready

VC6

short d64h_di_async_dblbuf_halfready (BYTE bCardID, BOOLEAN *pHalfReady)

VB6

d64h_di_async_dblbuf_halfready (ByVal bCardID As Byte, ByRef pHalfReady As Byte) As Integer

Description:

This function checks the availability of ring-buffer when the digital-input acquisition is configured as double-buffer mode. If the pHalfReady indicates the ring-buffer is ready, please call d64h_di_async_dblbuf_transfer() to copy the available acquisition data.

Parameters:

bCardID: the specific Card ID that is configured with the on-board Dip-Switch.

pHalfReady: the pointer to the memory that stores the availability of ring-buffer.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_ACCESS_VIOLATION_DATA_COPY: Some system exception occurs while copying memory, please check the pointer-type parameter you assign to this function.

ERROR_DI_ASYNC_HALF_READY: Fail to check the availability of ring-buffer.

4.5 d64h_di_async_dblbuf_transfer

VC6

short d64h_di_async_dblbuf_transfer (BYTE bCardID, void *pBuffer)

VB6

d64h_di_async_dblbuf_transfer (ByVal bCardID As Byte, pBuffer As Any) As Integer

Description:

This function extract the DI acquisition data from the ready half-buffer.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

pBuffer: The pointer to the user-provided buffer that the acquisition data will be transferred to.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_INVALID_MAPPED_ADDRESS: Indicates the mapped address is invalid.

ERROR_INVALID_DOUBLE_BUFFER_MODE: Indicates the acquisition operation is NOT configured as double-buffer mode.

ERROR_ACCESS_VIOLATION_DATA_COPY: Some system exception occurs while copying memory, please check the pointer-type parameter you assign to this function.

4.6 d64h_di_async_check

VC6

short d64h_di_async_check (BYTE bCardID, BOOLEAN *pStopped, U32 *pAccessCount)

VB6

d64h_di_async_check (ByVal bCardID As Byte, ByRef Stopped As Byte, ByRef AccessCnt As Long) As Integer

Description:

This function checks if the asynchronous operation is completed. If the digital-input is completed, the number of samples will be returned.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

pStopped: The pointer to the variable that stores the status of digital-input acquisition.

pAccessCount: The pointer to the variable that stores the number of samples when the digital-input is completed.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_SYNCH_OP_WITH_ASYNC_CHECK: Indicates the digital-input acquisition is configured as SYNCNH_OP mode, and d64h_di_async_check() is conflict with this operation mode.

ERROR_ACCESS_VIOLATION_DATA_COPY: Some system exception occurs while copying memory, please check the pointer-type parameter you assign to this function.

ERROR_DI_ASYNC_CHECK: Cannot get the status of digital-input buffer, please call GetLastError() for further system information.

4.7 d64h_di_async_clear

VC6

short d64h_di_async_clear (BYTE bCardID, U32 *pAccessCount)

VB6

d64h_di_async_clear (ByVal bCardID As Byte, ByRef AccessCnt As Long) As Integer

Description:

This function terminates the in-progress digital-input acquisition, d64h_continue_readport()

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

pAccessCount: The pointer to the variable that stores the number of acquired-samples

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_ACCESS_VIOLATION_DATA_COPY: Some system exception occurs while copying memory, please check the pointer-type parameter you assign to this function.

ERROR_DI_ASYNC_CLEAR: Cannot terminate the on-going digital-input acquisition.

4.8 d64h_continue_readport

VC6

short d64h_continue_readport(BYTE bCardID, void *pBuffer, U32 dwSampleCount, F64* pSampleRate, U16 wSyncMode)

VB6

d64h_continue_readport (ByVal bCardID As Byte, pBuffer As Any, ByVal dwSampleCount As Long, ByRef pSampleRate As Double, ByVal wSyncMode As Integer) As Integer

Description:

This function starts the continuous digital-input. The 32-bit acquisition data will be recorded into driver buffer continuously.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

pBuffer: The address of user-provided DI buffer. This parameter is ignored when enabling double-buffer mode.

dwSampleCount: The samples of each acquisition.

pSampleRate: The pointer to the address that stores sampling-rate. This call-by-reference parameter passes the desired sampling-rate to library; and stores the actual sampling-rate when this function returns. (0.001 valid sampling-rate 10,000,000)

wSyncMode : The digital-input acquisition mode, SYNCH_OP or ASYNCH_OP.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_SAMPLE_COUNT_TOO_LARGE: Indicates the value of **dwSampleCount** is larger than the DI buffer in driver.

ERROR_SAMPLE_COUNT_IS_ODD: Indicate the value of **dwSampleCount** is odd, and this is conflict with the double-buffer mode

ERROR_INVALID_SAMPLE_RATE: Indicates the invalid value is assigned to address that parameter **pSampleRate** point to

ERROR_INVALID_SYNCH_OP_MODE: Neither SYNCH_OP nor ASYNCH_OP is assigned to parameter **wSyncMode**.

ERROR_SYNCH_OP_WITH_DOUBLE_BUFFER_MODE: Indicates the digital-input is configured as double-buffer mode, and the SYCNH_OP setting is conflict with this mode. Please call `d64h_di_async_dblbuf_mode()` to change the operation mode.

ERROR_INVALID_MAPPED_ADDRESS: Indicates the kernel-mapped address is invalid.

ERROR_INVALID_BUFFER_ADDRESS: Indicates the user-provided buffer is invalid.

ERROR_DMA_IN_PROGRESS: Indicates the digital-input acquisition is in progress, please call `d64h_di_async_clear()` to terminate the current acquisition.

ERROR_OVERLAP_EVENT_CREATE: Indicates the Event-Object creating is failed, please call `GetLastError()` for further system information.

ERROR_ACCESS_VIOLATION_DATA_COPY: Some system exception occurs while copying memory, please check the pointer-type parameter you assign to this function.

ERROR_DI_ASYNC_CLEAR: Cannot terminate the current acquisition, please call `GetLastError()` for further system information.

ERROR_CONTINUE_DI_START: Cannot start continuous digital-input acquisition, please call `GetLastError()` for further system information.

4.9 d64h_conti_di_status

VC6

short d64h_conti_di_status (BYTE bCardID, U16 *pStatus)

VB6

d64h_conti_di_status (ByVal bCardID As Byte, ByRef pStatus As Integer) As Integer

Description:

This function reports the buffer-overflow status of digital-input buffer.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

pStatus: The pointer to the variable that stores the status of digital-input buffer. The value in this variable will be 0 or DI_OVERRUN_STATUS.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_ACCESS_VIOLATION_DATA_COPY: Some system exception occurs while copying memory, please check the pointer-type parameter you assign to this function.

ERROR_FIFO_STATUS_GET: Cannot get the status of digital-input buffer, please call GetLastError() for further system information.

Digital Output Functions

The functions in this chapter are provided for digital-input operations.

5.1 d64h_do_writeport

VC6

short d64h_do_writeport (BYTE bCardID, U32 dwValue)

VB6

d64h_do_readport (ByVal bCardID As Byte, ByVal dwValue As Long) As Integer

Description:

This function writes all digital-output with the user-assigned data.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

dwValue: The data to be written to digital-output port.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_DO_PIO_WRITE: Cannot write the digital-output port, please call GetLastError() for further system information.

5.2 d64h_do_writeline

VC6

short d64h_do_writeline (BYTE bCardID, U16 wLine, U16 wValue)

VB6

d64h_do_writeline (ByVal bCardID As Byte, ByVal wLine As Integer, ByVal wValue As Integer)
As Integer

Description:

This function writes specific digital-input line.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

wLine: The parameter indicates the specific line to be input.

wValue: The data to be written to specific line of digital-output port.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_INVALID_PORT_LINE: Indicates the invalid setting is assigned to parameter **wLine**.

ERROR_DO_PIO_LINE_WRITE: Cannot read the specific digital-input line, please call GetLastError() for further system information.

5.3 d64h_do_readport

VC6

short d64h_do_readport(BYTE bCardID, U32* pValue)

VB6

d64h_do_readport(ByVal bCardID As Byte, ByRef pValue As Long) As Integer

Description:

This function reads all digital-output data, and stores in one 32-bit variable.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

pValue: The pointer to the 32-bit variable that stores all digital-output data.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_ACCESS_VIOLATION_DATA_COPY: Some system exception occurs while copying memory, please check the pointer-type parameter you assign to this function.

ERROR_DO_PIO_READ: Cannot read the digital-output port, please call GetLastError() for further system information.

5.4 d64h_do_readline

VC6

short d64h_do_readline(BYTE bCardID, U16 wLine, U16* pValue)

VB6

d64h_do_readline(ByVal bCardID As Byte, ByVal wLine As Integer, ByRef pValue As Integer)
As Integer

Description:

This function reads specific digital-input line.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

wLine: The parameter indicates the specific line to be output.

pValue: The pointer to the 16-bit variable that stores the specific digital-output line.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_INVALID_PORT_LINE: Indicates the invalid setting is assigned to parameter **wLine**.

ERROR_ACCESS_VIOLATION_DATA_COPY: Some system exception occurs while copying memory, please check the pointer-type parameter you assign to this function.

ERROR_DO_PIO_READ: Cannot read the specific digital-output line, please call GetLastError() for further system information.

5.5 d64h_do_async_dblbuf_mode

VC6

short d64h_do_async_dblbuf_mode (BYTE bCardID, U16 wDbIBufEnable)

VB6

d64h_do_async_dblbuf_mode (ByVal bCardID As Byte, ByVal wDbIBufEnable As Integer) As Integer

Description:

This function configures the operation mode of digital-output acquisition.

Parameters:

bCardID: the specific Card ID that is configured with the on-board Dip-Switch.

wDbIBufEnable: DOUBLE_BUFFER_MODE_ENABLE or DOUBLE_BUFFER_MODE_DISABLE.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_INVALID_MAPPED_ADDRESS: Indicates the mapped address is invalid.

ERROR_INVALID_DOUBLE_BUFFER_MODE: Neither DOUBLE_BUFFER_MODE_ENABLE nor DOUBLE_BUFFER_MODE_DISABLE is assigned to parameter **wDbIBufEnable**.

ERROR_DOUBLE_BUFFER_MODE: Cannot configure the digital-input acquisition as double-buffer mode

5.6 d64h_do_async_dblbuf_halfready

VC6

short d64h_do_async_dblbuf_halfready (BYTE bCardID, BOOLEAN *pHalfReady)

VB6

d64h_do_async_dblbuf_halfready (ByVal bCardID As Byte, ByVal pHalfReady As Byte) As Integer

Description:

This function checks the availability of ring-buffer when the digital-output data is transferred as double-buffer mode. If the pHalfReady indicates the ring-buffer is ready, the d64h_do_async_dblbuf_transfer() helps to update the output data.

Parameters:

bCardID: the specific Card ID that is configured with the on-board Dip-Switch.

pHalfReady: the pointer to the memory that stores the availability of ring-buffer.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_ACCESS_VIOLATION_DATA_COPY: Some system exception occurs while copying memory, please check the pointer-type parameter you assign to this function.

ERROR_DO_ASYNC_HALF_READY: Fail to check the availability of ring-buffer.

5.7 d64h_do_async_dblbuf_transfer

VC6

short d64h_do_async_dblbuf_transfer (BYTE bCardID, void *pBuffer)

VB6

d64h_do_async_dblbuf_transfer (ByVal bCardID As Byte, pBuffer As Any) As Integer

Description:

This function updates the DO output data to the idle half-buffer.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

pBuffer: The pointer to the user-provided buffer that contains the DO output data.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_INVALID_MAPPED_ADDRESS: Indicates the mapped address is invalid.

ERROR_INVALID_DOUBLE_BUFFER_MODE: Indicates the acquisition operation is NOT configured as double-buffer mode.

ERROR_ACCESS_VIOLATION_DATA_COPY: Some system exception occurs while copying memory, please check the pointer-type parameter you assign to this function.

5.8 d64h_do_async_check

VC6

short d64h_do_async_check (BYTE bCardID, BOOLEAN *pStopped, U32 *pAccessCount)

VB6

d64h_do_async_check (ByVal bCardID As Byte, ByRef Stopped As Byte, ByRef AccessCnt As Long) As Integer

Description:

This function checks if the asynchronous operation is completed. If the digital-output is completed, the number of samples will be returned.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

pStopped: The pointer to the variable that stores the status of digital-output operation.

pAccessCount: The pointer to the variable that stores the number of samples when the digital-output is completed.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_SYNCH_OP_WITH_ASYNC_CHECK: Indicates the digital-input acquisition is configured as SYNCNH_OP mode, and d64h_do_async_check() is conflict with this operation mode.

ERROR_ACCESS_VIOLATION_DATA_COPY: Some system exception occurs while copying memory, please check the pointer-type parameter you assign to this function.

ERROR_DO_ASYNC_CHECK: Cannot get the status of digital-input buffer, please call GetLastError() for further system information.

5.9 d64h_do_async_clear

VC6

short d64h_do_async_clear (BYTE bCardID, U32 *pAccessCount)

VB6

d64h_do_async_clear (ByVal bCardID As Byte, ByRef AccessCnt As Long) As Integer

Description:

This function terminates the in-progress digital-output operation, d64h_continue_writeport() and d64h_continue_pattern_write().

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

pAccessCount: The pointer to the variable that stores the number of acquired-samples

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_ACCESS_VIOLATION_DATA_COPY: Some system exception occurs while copying memory, please check the pointer-type parameter you assign to this function.

ERROR_DO_ASYNC_CLEAR: Cannot terminate the on-going digital-input acquisition.

5.10 d64h_continue_writeport

VC6

short d64h_continue_writeport (BYTE bCardID, void *pBuffer, U32 dwSampleCount, U16 wIterations, F64* pSampleRate, U16 wSyncMode)

VB6

d64h_continue_writeport (ByVal bCardID As Byte, pBuffer As Any, ByVal dwSampleCount As Long, ByVal wIterations As Integer, ByRef pSampleRate As Double, ByVal wSyncMode As Integer) As Integer

Description:

This function starts the continuous digital-input. The 32-bit acquisition data will be recorded into driver buffer continuously. If the digital-data is needed to be output repeatedly, please call `d64h_do_async_dblbuf_mode()` and enable double-buffer mode.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

pBuffer: The address of user-provided DO buffer.

dwSampleCount: The samples of each acquisition.

wIterations: This parameter is reserved for future use.

pSampleRate: The pointer to the address that stores sampling-rate. This call-by-reference parameter passes the desired sampling-rate to library; and stores the actual sampling-rate when this function returns. (0.001 valid sampling-rate 10,000,000)

wSyncMode : The digital-output operation mode, SYNCH_OP or ASYNCH_OP.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_SAMPLE_COUNT_TOO_LARGE: Indicates the value of **dwSampleCount** is larger than the DI buffer in driver.

ERROR_INVALID_SAMPLE_RATE: Indicates the invalid value is assigned to address that parameter **pSampleRate** point to.

ERROR_SYNCH_OP_WITH_DOUBLE_BUFFER_MODE: Indicates the digital-output is configured as double-buffer mode, and the SYCNH_OP setting is conflict with this mode. Please call `d64h_do_async_dblbuf_mode()` to change the operation mode.

ERROR_INVALID_SYNCH_OP_MODE: Neither SYNCH_OP nor ASYNCH_OP is assigned to parameter **wSyncMode**.

ERROR_INVALID_MAPPED_ADDRESS: Indicates the kernel-mapped address is invalid.

ERROR_INVALID_BUFFER_ADDRESS: Indicates the user-provided buffer is invalid.

ERROR_DMA_IN_PROGRESS: Indicates the digital-input acquisition is in progress, please call `d64h_do_async_clear()` to terminate the current acquisition.

ERROR_OVERLAP_EVENT_CREATE: Indicates the Event-Object creating is failed, please call `GetLastError()` for further system information.

ERROR_ACCESS_VIOLATION_DATA_COPY: Some system exception occurs while copying memory, please check the pointer-type parameter you assign to this function.

ERROR_DO_ASYNC_CLEAR: Cannot terminate the current acquisition, please call `GetLastError()` for further system information.

ERROR_CONTINUE_DO_START: Cannot start continuous digital-input acquisition, please call `GetLastError()` for further system information.

5.11 d64h_continue_pattern_write

VC6

short d64h_continue_pattern_write (BYTE bCardID, void *pBuffer, U32 dwSampleCount, F64* pSampleRate)

VB6

d64h_conti_pattern_write (ByVal bCardID As Byte, pBuffer As Any, ByVal dwSampleCount As Long, ByRef pSampleRate As Double) As Integer

Description:

This function reads the output-pattern and stores it into a circular buffer, and loops the output infinitely. This feature is supported by onboard-circuit, and neither computing-power nor bus-bandwidth is consumed.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

pBuffer: The address of user-provided DO buffer.

dwSampleCount: The samples of each acquisition. (2 < dwSampleCount ≤ 2048)

pSampleRate: The pointer to the address that stores sampling-rate. This call-by-reference parameter passes the desired sampling-rate to library; and stores the actual sampling-rate when this function returns. (0.001 ≤ valid sampling-rate ≤ 10,000,000)

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_SAMPLE_COUNT_TOO_LARGE: Indicates the value of **dwSampleCount** is larger than the DI buffer in driver.

ERROR_INVALID_SAMPLE_RATE: Indicates the invalid value is assigned to address that parameter **pSampleRate** point to.

ERROR_PATTERN_OUT_WITH_DOUBLE_BUFFER_MODE: Indicates the digital-output is configured as double-buffer mode, this mode is invalid for Patten-Output operation. Please call d64h_do_async_dblbuf_mode() to change the operation mode.

ERROR_INVALID_MAPPED_ADDRESS: Indicates the kernel-mapped address is invalid.

ERROR_INVALID_BUFFER_ADDRESS: Indicates the user-provided buffer is invalid.

ERROR_INVALID_DO_ITERATIONS: Indicates the digital-output operation is configured as SYNCH_OP, and the infinite-iteration setting is conflict with this operation mode

ERROR_DMA_IN_PROGRESS: Indicates the digital-input acquisition is in progress, please call `d64h_do_async_clear()` to terminate the current acquisition.

ERROR_OVERLAP_EVENT_CREATE: Indicates the Event-Object creating is failed, please call `GetLastError()` for further system information.

ERROR_ACCESS_VIOLATION_DATA_COPY: Some system exception occurs while copying memory, please check the pointer-type parameter you assign to this function.

ERROR_DO_ASYNC_CLEAR: Cannot terminate the current acquisition, please call `GetLastError()` for further system information.

ERROR_CONTINUE_DO_START: Cannot start continuous digital-input acquisition, please call `GetLastError()` for further system information.

5.12 d64h_conti_do_status

VC6

short d64h_conti_do_status (BYTE bCardID, U16 *pStatus)

VB6

d64h_conti_do_status (ByVal bCardID As Byte, ByRef pStatus As Integer) As Integer

Description:

This function reports the buffer-overflow status of digital-output buffer.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

pStatus: The pointer to the variable that stores the status of digital-output buffer. The value in this variable will be 0 or DO_UNDERRUN_STATUS.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_ACCESS_VIOLATION_DATA_COPY: Some system exception occurs while copying memory, please check the pointer-type parameter you assign to this function.

ERROR_FIFO_STATUS_GET: Cannot get the status of digital-output buffer, please call GetLastError() for further system information.

5.13 d64h_do_ext_trigline_write

VC6

short d64h_do_ext_trigline_write (BYTE bCardID, U16 wValue)

VB6

d64h_do_ext_trigline_write (ByVal bCardID As Byte, ByVal wValue As Integer) As Integer

Description:

This function reports the buffer-overflow status of digital-input buffer.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

wValue: The state to be set on external trigger line, OTRIG_HIGH or OTRIG_LOW.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_DO_EXT_TRIGLINE_WRITE: Cannot set the state of external trigger line, please call GetLastError() for further system information.

Digital Filter Functions

This chapter introduces the functions to configure the digital filters and the width of O_REQ pulse.

6.1 d64h_DI_filter_set

VC6

short d64h_DI_filter_set (BYTE bCardID, U16 wDIFilter)

VB6

d64h_DI_filter_set (ByVal bCardID As Byte, ByVal wDIFilter As Integer) As Integer

Description:

This function set the digital filter for all DI lines.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

wDIFilter: the delay factor for digital filter (0 wDIFilter 127).

$$wDIFilter * 25ns < \text{delay for data-latching} \quad (wDIFilter + 1) * 25ns$$

Notice:

Please take the input sampling-rate into consideration when you set this parameter.

Assuming the output data is sampled at 10MHz, the input data may update every 100ns.

Therefore,

$$wDIFilter * 25ns < 100ns$$

$$wDIFilter < 4$$

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_INVALID_FILTER_SETTING: Indicates the filter-setting is out of range.

ERROR_DIGITAL_FILTER_SET: Cannot set the digital filter, please call GetLastError() for further system information.

6.2 d64h_IREQ_filter_set

VC6

short d64h_IREQ_filter_set (BYTE bCardID, U16 wIREQFilter)

VB6

d64h_IREQ_filter_set (ByVal bCardID As Byte, ByVal wIREQFilter As Integer) As Integer

Description:

This function set the digital filter for I_REQ input signal.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

wIREQFilter: the delay factor for digital filter (0 wIREQFilter 127).

$wIREQFilter * 25ns < \text{delay for data-latching} < (wIREQFilter + 1) * 25ns$

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_INVALID_FILTER_SETTING: Indicates the filter-setting is out of range.

ERROR_DIGITAL_FILTER_SET: Cannot set the I_REQ filter, please call GetLastError() for further system information

6.3 d64h_OACK_filter_set

VC6

short d64h_OACK_filter_set (BYTE bCardID, U16 wOACKFilter)

VB6

d64h_OACK_filter_set (ByVal bCardID As Byte, ByVal wOACKFilter As Integer) As Integer

Description:

This function set the digital filter for O_ACK input signal.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

wOACKFilter: the delay factor for digital filter (0 wOACKFilter 127).

$wOACKFilter * 25ns < \text{delay for data-latching} < (wOACKFilter + 1) * 25ns$

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_INVALID_FILTER_SETTING: Indicates the filter-setting is out of range.

ERROR_DIGITAL_FILTER_SET: Cannot set the O_ACK filter, please call GetLastError() for further system information

6.4 d64h_ITRG_filter_set

VC6

short d64h_ITRG_filter_set (BYTE bCardID, U16 wITRGFilter)

VB6

d64h_ITRG_filter_set (ByVal bCardID As Byte, ByVal wITRGFilter As Integer) As Integer

Description:

This function set the digital filter for I_TRG input signal.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

wITRGFilter: the delay factor for digital filter (0 wITRGFilter 127).

$wITRGFilter * 25ns < \text{delay for data-latching} < (wITRGFilter + 1) * 25ns$

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_INVALID_FILTER_SETTING: Indicates the filter-setting is out of range.

ERROR_DIGITAL_FILTER_SET: Cannot set the I_TRG filter, please call GetLastError() for further system information

6.5 d64h_OREQ_width_set

VC6

short d64h_OREQ_width_set (BYTE bCardID, U16 wOREQWidth)

VB6

d64h_OREQ_width_set (ByVal bCardID As Byte, ByVal wOREQWidth As Integer) As Integer

Description:

This function set the width of O_REQ output signal.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

wOREQWidth: the delay factor for digital filter (0 wOREQWidth 7).

$$\text{Width of O_REQ} = (2^{\text{wOREQWidth} + 1}) * 25\text{ns}$$

Notice:

The period of O_REQ signal will be twice as long as the time at high-level. Please take the output update-rate into consideration when you set this parameter.

Assuming the output date is updated at 1MHz, the maximum period of O_REQ signal is 1000ns.

Therefore,

$$2 * (2^{\text{wOREQWidth} + 1}) * 25\text{ns} \leq 1000\text{ns}$$
$$2^{\text{wOREQWidth} + 1} \leq 20, \text{ wOREQWidth} < 4$$

Notice:

The period of O_REQ signal will be twice longer than the time at high-level. Therefore, please take the output update-rate into consideration when you set this parameter

The period of O_REQ signal will be twice longer than the time at high-level. Therefore, please take the output update-rate into consideration when you set this parameter.

$$(\text{O_REQ Period} > 2 * (2^{\text{wOREQWidth} + 1}) * 25\text{ns})$$

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_INVALID_OREQ_WIDTH: Indicates the width-setting is out of range.

ERROR_OREQ_WIDTH_SET: Cannot set the width of O_REQ, please call GetLastError() for further system information

Event Notification Functions

This chapter introduces the functions to register the related routine for the relevant events.

7.1 d64h_di_event_callback

VC6

```
short d64h_di_event_callback (BYTE bCardID, U16 wEventEnable, U16 wEventType, void  
(*callbackAddr)() )
```

Description:

This function adds/removes the event-notification of digital-input acquisition, and registers the related callback function.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

wEventEnable: ADD_NOTIFICATION or REMOVE_NOTIFICATION.

wEventType: DIEnd or DBEvent.

callbackAddr: the function-pointer of callback function.

Notice:

One internal thread will be created to monitor that specific event. Once that event is triggered by driver, the internal thread will call the function-pointer that is stored in parameter **callbackAddr**.

To avoid missing event-notification, the callback function must return before next triggered-event.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_INVALID_EVENT_ACTION: Neither ADD_NOTIFICATION nor REMOVE_NOTIFICATION is assigned to parameter **wEventEnable**.

ERROR_INVALID_CALLBACK_ADDRESS: Indicates the function-pointer of callback-function is invalid.

ERROR_INVALID_EVENT_ACTION: Neither DIEnd nor DBEvent is assigned to parameter **wEventType**.

ERROR_DI_EVENT_ATTACH: Cannot attach the notification-event to driver, please call GetLastError() for further system information.

ERROR_DI_EVENT_DETACH: Cannot detach the notification-event from driver, please call GetLastError() for further system information.

7.1 d64h_do_event_callback

VC6

short d64h_do_event_callback (BYTE bCardID, U16 wEventEnable, U16 wEventType, void (*callbackAddr)())

Description:

This function adds/removes the event-notification of digital-output operation, and registers the related callback function.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

wEventEnable: ADD_NOTIFICATION or REMOVE_NOTIFICATION.

wEventType: DOEnd or DBEvent.

callbackAddr: the function-pointer of callback function. It's recommended

Notice:

One internal thread will be created to monitor that specific event. Once that event is triggered by driver, the internal thread will call the function-pointer that is stored in parameter **callbackAddr**.

To avoid missing event-notification, the callback function must return before next triggered-event.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_INVALID_EVENT_ACTION: Neither ADD_NOTIFICATION nor REMOVE_NOTIFICATION is assigned to parameter **wEventEnable**.

ERROR_INVALID_CALLBACK_ADDRESS: Indicates the function-pointer of callback-function is invalid.

ERROR_INVALID_EVENT_ACTION: Indicates incorrect setting is assigned to parameter **wEventType**.

ERROR_DO_EVENT_ATTACH: Cannot attach the notification-event to driver, please call GetLastError() for further system information.

ERROR_DO_EVENT_DETACH: Cannot detach the notification-event from driver, please call GetLastError() for further system information.



Error Code

The Error Codes are divided into three parts: System Error, Parameter Error and Runtime Error.

SUCCESS_NO_ERROR	0
------------------	---

System Error:

ERROR_ROUTINE_FAIL_BASE	-100
ERROR_GET_CARD_ID	-101
ERROR_DEVICE_OPEN	-102
ERROR_DEVICE_CLOSE	-103
ERROR_DOUBLE_BUFFER_MODE	-104
ERROR_DI_CONFIG	-105
ERROR_DI_ASYNC_CLEAR	-106
ERROR_DI_ASYNC_CHECK	-107
ERROR_DI_ASYNC_HALF_READY	-108
ERROR_DI_PIO_READ	-109
ERROR_DO_CONFIG	-110
ERROR_DO_ASYNC_CLEAR	-111
ERROR_DO_ASYNC_CHECK	-112
ERROR_DO_PIO_READ	-113
ERROR_DO_PIO_WRITE	-114
ERROR_DO_PIO_LINE_WRITE	-115
ERROR_CONTINUE_DI_START	-116
ERROR_CONTINUE_DO_START	-117
ERROR_FIFO_STATUS_GET	-118
ERROR_DO_EXT_TRIGLINE_WRITE	-119
ERROR_DO_HALF_READY	-120
ERROR_DIGITAL_FILTER_SET	-121
ERROR_OREQ_WIDTH_SET	-122

Parameter Error:

ERROR_INVALID_PARAMETER_BASE	-200
ERROR_INVALID_CARD_ID	-201
ERROR_INVALID_SCANNED_INDEX	-202
ERROR_CARD_ID_DUPLICATED	-203
ERROR_INVALID_DOUBLE_BUFFER_MODE	-204
ERROR_INVALID_PORT_LINE	-205
ERROR_INVALID_TRIGGER_SOURCE	-206
ERROR_INVALID_TRIGGER_ENABLE	-207
ERROR_INVALID_TRIGGER_POLARITY	-208
ERROR_INVALID_IREQ_POLARITY	-209
ERROR_INVALID_OTRIG_LEVEL	-210
ERROR_INVALID_OREG_ENABLE	-211
ERROR_INVALID_DOUBLE_BUFFER_MODE	-212
ERROR_INVALID_SYNCH_OP_MODE	-213
ERROR_SAMPLE_COUNT_IS_ODD	-214
ERROR_INVALID_DO_ITERATIONS	-215
ERROR_INVALID_EVENT_ACTION	-216
ERROR_INVALID_CALLBACK_ADDRESS	-217
ERROR_INVALID_BUFFER_ADDRESS	-218
ERROR_INVALID_PATTERN_SIZE	-219
ERROR_INVALID_FILTER_SETTING	-220
ERROR_INVALID_OREG_WIDTH	-221

Runtime Error:

ERROR_RUNTIME_BASE	-300
--------------------	------

ERROR_NO_CARD_FOUND	-316
ERROR_MEMORY_MAP	-317
ERROR_MEMORY_UNMAP	-318
ERROR_ACCESS_VIOLATION_DATA_COPY	-319
ERROR_VARIABLE_PITCH_SET	-320
ERROR_DI_EVENT_ATTACH	-321
ERROR_DI_EVENT_DETACH	-322
ERROR_DO_EVENT_ATTACH	-323
ERROR_DO_EVENT_DETACH	-324
ERROR_EVENT_CREATE_FAILED	-325
ERROR_OVERLAP_EVENT_CREATE	-326
ERROR_IOCTL_FAILED	-361
ERROR_UNDEFINED_EXCEPTION	-362

DIO Runtime Error:

ERROR_DIO_RUNTIME_BASE	-500
ERROR_DEVICE_POWER_DOWN	-501
ERROR_INVALID_MAPPED_ADDRESS	-502
ERROR_BUFFER_NOT_ENOUGH	-503
ERROR_DMA_NOT_AVAILABLE	-504
ERROR_DMA_NOT_COMPLETE	-505
ERROR_DMA_NOT_STARTED	-506
ERROR_DMA_NOT_PAUSED	-507
ERROR_DMA_IS_PAUSED	-508
ERROR_DMA_IN_PROGRESS	-509
ERROR_INVALID_SAMPLE_RATE	-510
ERROR_SAMPLE_COUNT_TOO_LARGE	-511
ERROR_SYNCH_OP_WITH_DOUBLE_BUFFER_MODE	-512
ERROR_SYNCH_OP_WITH_ASYNC_CHECK	-513
ERROR_UNSUPPORTED_SETTING	-514
ERROR_PATTERN_OUT_WITH_DOUBLE_BUFFER_MODE	-515
ERROR_PATTERN_OUT_IN_PROGRESS	-516

PCI-D64HU Function Reference

(Version 1.0)



ICP DAS CO., LTD.

Warranty

All products manufactured by ICPDAS Inc. are warranted against defective materials for a period of one year from the date of delivery to the original purchaser.

Warning

ICPDAS Inc. assumes no liability for damages consequent to the use of this product. ICPDAS Inc. reserves the right to change this manual at any time without notice. The information furnished by ICPDAS Inc. is believed to be accurate and reliable. However, no responsibility is assumed by ICPDAS Inc. for its use, or for any infringements of patents or other rights of third parties resulting from its use.

Trademark

The names used for identification only maybe registered trademarks of their respective companies.

License

The user can use, modify and backup this software on a single machine. The user may not reproduce, transfer or distribute this software, or any copy, in whole or in part.

Contents

INTRODUCTION	5
SYSTEM INITIALIZATION	6
2.1 <i>d64h_scan</i>	6
2.2 <i>d64h_get_cardinfo</i>	8
2.3 <i>d64h_open</i>	9
2.4 <i>d64h_close</i>	10
2.5 <i>d64h_di_available_memory</i>	11
2.6 <i>d64h_do_available_memory</i>	12
2.7 <i>d64h_di_buffer_get</i>	13
2.8 <i>d64h_do_buffer_get</i>	14
DIGITAL INPUT/OUTPUT CONFIGURATION	15
3.1 <i>d64h_di_config</i>	15
3.2 <i>d64h_do_config</i>	17
DIGITAL INPUT FUNCTIONS.....	18
4.1 <i>d64h_di_readport</i>	18
4.2 <i>d64h_di_readline</i>	19
4.3 <i>d64h_di_async_dblbuf_mode</i>	20
4.4 <i>d64h_di_async_dblbuf_halfready</i>	21
4.5 <i>d64h_di_async_dblbuf_transfer</i>	22
4.6 <i>d64h_di_async_check</i>	23
4.7 <i>d64h_di_async_clear</i>	24
4.8 <i>d64h_continue_readport</i>	25
4.9 <i>d64h_conti_di_status</i>	27
DIGITAL OUTPUT FUNCTIONS.....	28
5.1 <i>d64h_do_writeport</i>	28
5.2 <i>d64h_do_writeline</i>	29
5.3 <i>d64h_do_readport</i>	30
5.4 <i>d64h_do_readline</i>	31
5.5 <i>d64h_do_async_dblbuf_mode</i>	32
5.6 <i>d64h_do_async_dblbuf_halfready</i>	33
5.7 <i>d64h_do_async_dblbuf_transfer</i>	34
5.8 <i>d64h_do_async_check</i>	35

5.9	<i>d64h_do_async_clear</i>	36
5.10	<i>d64h_continue_writeport</i>	37
5.11	<i>d64h_continue_pattern_write</i>	39
5.12	<i>d64h_conti_do_status</i>	41
5.13	<i>d64h_do_ext_trigline_write</i>	42
DIGITAL FILTER FUNCTIONS		43
6.1	<i>d64h_DI_filter_set</i>	43
6.2	<i>d64h_IREQ_filter_set</i>	45
6.3	<i>d64h_OACK_filter_set</i>	46
6.4	<i>d64h_ITRG_filter_set</i>	47
6.5	<i>d64h_OREQ_width_set</i>	48
EVENT NOTIFICATION FUNCTIONS		50
7.1	<i>d64h_di_event_callback</i>	50
7.1	<i>d64h_do_event_callback</i>	52
ERROR CODE		54

Introduction

This software package is dedicated to PCI-D64HU high-speed digital input/output card. It includes the WDM (Windows Driver Model) driver and ANSI-C Library for Windows 2000/XP.

One unique Card ID will be referred by each function in this Library. The Card ID is configured with on-board DIP-Switch, and helps to identify multiple PCI-D64HU cards in your system. In other words, you no longer worry about the order that Operating System scans PCI-D64HU cards; the only thing you must take care is the correct relationship between the terminal-boards and PCI-D64HU cards.

There are samples that are provided for Microsoft® Visual Studio 6.0 (VC and VB) to demonstrate the functions of PCI-D64HU Library.

This documentation provides the detailed information of PCI-D64HU APIs, including the function-declaration, definitions of both parameters and return codes. The APIs will be cataloged and described in the following chapters:

- CHAPTER 2 – System Initialization
- CHAPTER 3 – Digital Input/Output Configuration
- CHAPTER 4 – Digital Input Functions
- CHAPTER 5 – Digital Output Functions
- CHAPTER 6 – Digital Filter Functions
- CHAPTER 7 – Event Notification Functions

System Initialization

The functions in this chapter provide the interface to Operating-System. By calling these functions, your applications can scan all active PCI-D64HU cards in your system, and get the specific Card-ID that is configured with the on-board Dip-Switch. Open the card before calling other functions in PCI-D64HU Library.

2.1 d64h_scan

VC6

short d64h_scan(short* pCardNum, BYTE* pAvailCards = NULL)

VB6

d64h_scan(ByRef pCardNum As Integer, Optional pAvailCards As Byte = 0) As Integer

Description:

This function scans all active PCI-D64HU cards in your system. The pCardNum saves the numbers of active PCI-D64HU cards. The optional user-provided Array, pAvailCards, indicates the presence of active PCI-D64HU card. (1: present, 0: absent)

Parameters:

pCardNum: The pointer to the memory that stores the numbers of active PCI-D64HU cards.

pAvailCard: The address of user-provided **BYTE**-Array. Based on the Card ID, each element indicates the presence of active PCI-D64HU card. The user must prepare one **BYTE**-Array with **D64H_MaxCards** elements.

For instance, there are three active PCI-D64HU cards with Card ID 3, 5 and 7. The content of pAvailCard Array will be

```
{ 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0 }
```

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_NO_CARD_FOUND: There is no active card available in your system.

ERROR_CARD_ID_DUPLICATED: There are multiple cards that are assigned the same Card ID, please check the settings of on-board Dip-Switch.

ERROR_ACCESS_VIOLATION_DATA_COPY: Some system exception occurs while copying memory, please check the pointer-type parameter you assign to this function.

2.2 d64h_get_cardinfo

VC6

short d64h_get_cardinfo(int ScannedIndex, BYTE* pCardID)

VB6

d64h_get_cardinfo(ByVal ScannedIndex As Integer, ByRef pCardID As Byte) As Integer

Description:

This function returns the Card ID based on the scanned-index. This routine will get the Card ID configured with on-board Dip-Switch.

Parameters:

ScannedIndex: The index that the active PCI-D64HU is scanned. This index begins from 0, and is less than the active PCI-D64HU cards.

pCardID: The pointer to the memory that stores the specific Card ID.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_NO_CARD_FOUND: There is no active card available in your system.

ERROR_INVALID_SCANNED_INDEX: Indicates the **ScannedIndex** parameter is larger than the numbers of active PCI-D64HU cards.

ERROR_ACCESS_VIOLATION_DATA_COPY: Some system exception occurs while copying memory, please check the pointer-type parameter you assign to this function.

2.3 d64h_open

VC6

short d64h_open(BYTE bCardID)

VB6

d64h_open(ByVal bCardID As Byte) As Integer

Description:

This function opens the device node of PCI-D64HU based on the specific Card ID. If this function returns successfully, the process that calls this function will own that specific device until d64h_close() is called. The device node of PCI-D64HU is ought to be owned before calling other functions. It's recommended to call d64h_scan() and d64h_get_cardinfo() to get the Card ID.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_DEVICE_OPEN: Fail to open the device-node of PCI-D64HU. Please make sure no other process owns that PCI-D64HU card.

ERROR_EVENT_CREATE_FAILED: Fail to create the related events for digital input/output operations.

ERROR_MEMORY_MAP: Indicates the Memory-Mapping is failed, please check the event logs in Event Viewer.

2.4 d64h_close

VC6

short d64h_close(BYTE bCardID)

VB6

d64h_close(ByVal bCardID As Byte) As Integer

Description:

This function closes the device node of PCI-D64HU based on the specific Card ID. After calling this function, the PCI-D64HU card will be released, and other process can open it.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no opened PCI-D64HU card with assigned Card ID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_INVALID_MAPPED_ADDRESS: Indicates the mapped address is invalid.

ERROR_DI_EVENT_DETACH: Fail to detach the digital-input related event.

ERROR_DO_EVENT_DETACH: Fail to detach the digital-output related event.

ERROR_DEVICE_CLOSE: Fail to close the device-node of PCI-D64HU.

ERROR_MEMORY_UNMAP: Indicates the Memory-Un-mapping is failed, please check the event logs in Event Viewer.

2.5 d64h_di_available_memory

VC6

short d64h_di_available_memory(BYTE bCardID, U32 *pMemSize)

VB6

d64h_di_available_memory (ByVal bCardID As Byte, ByRef pMemSize As Long) As Integer

Description:

This function gets the size of DI buffer that is allocated in driver. The unit of allocated buffer is reported in kilo-bytes.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

pMemSize: The pointer to the size of DI buffer, in kilobytes (KB).

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no opened PCI-D64HU card with assigned Card ID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_INVALID_MAPPED_ADDRESS: Indicates the mapped address is invalid.

ERROR_ACCESS_VIOLATION_DATA_COPY: Some system exception occurs while copying memory, please check the pointer-type parameter you assign to this function.

2.6 d64h_do_available_memory

VC6

short d64h_do_available_memory(BYTE bCardID, U32 *pMemSize)

VB6

d64h_do_available_memory (ByVal bCardID As Byte, ByRef pMemSize As Long) As Integer

Description:

This function gets the size of DO buffer that is allocated in driver. The unit of allocated buffer is reported in kilo-bytes.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

pMemSize: The pointer to the size of DO buffer, in kilobytes (KB)

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no opened PCI-D64HU card with assigned Card ID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_INVALID_MAPPED_ADDRESS: Indicates the mapped address is invalid.

ERROR_ACCESS_VIOLATION_DATA_COPY: Some system exception occurs while copying memory, please check the pointer-type parameter you assign to this function.

2.7 d64h_di_buffer_get

VC6

```
short d64h_di_buffer_get(BYTE bCardID, U32 *pMemSize, PVOID* pLowBufAddr, PVOID*  
pHighBufAddr )
```

Description:

This function gets the size of DI buffer and the virtual address to access this DI buffer. These buffer addresses help programmer to get the DI acquisition data directly.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

pMemSize: The pointer to the size of DI buffer, in kilobytes (KB)

pLowBufAddr: The pointer to the address of low-part buffer.

pHighBufAddr: The pointer to the address of high-part buffer.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no opened PCI-D64HU card with assigned Card ID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_INVALID_MAPPED_ADDRESS: Indicates the mapped address is invalid.

ERROR_ACCESS_VIOLATION_DATA_COPY: Some system exception occurs while copying memory, please check the pointer-type parameter you assign to this function.

2.8 d64h_do_buffer_get

VC6

```
short d64h_do_buffer_get(BYTE bCardID, U32 *pMemSize, PVOID* pLowBufAddr, PVOID*  
pHighBufAddr )
```

Description:

This function gets the size of DO buffer and the virtual address to access this DO buffer. These buffer addresses help programmer to update the DO output data directly.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

pMemSize: The pointer to the size of DO buffer, in kilobytes (KB)

pLowBufAddr: The pointer to the address of low-part buffer.

pHighBufAddr: The pointer to the address of high-part buffer.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no opened PCI-D64HU card with assigned Card ID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_INVALID_MAPPED_ADDRESS: Indicates the mapped address is invalid.

ERROR_ACCESS_VIOLATION_DATA_COPY: Some system exception occurs while copying memory, please check the pointer-type parameter you assign to this function.

Digital Input/Output Configuration

The functions in this chapter configure the operation mode of digital input/output. These configurations will be applied to next continuous digital input/output functions, say `d64h_continue_readport()`, `d64h_continue_writeport()` and `d64h_continue_pattern_write()`. And these specific settings will be reset after calling `d64h_di_async_clear()` and `d64h_do_async_clear()`.

When the trigger-source is configured as `TRIG_SOURCE_INT_PACER`, the desired sampling-rate will be generated with the internal clock-source (20MHz) and dividers. Therefore, the actual sampling-rate is not exactly equal to desired sampling-rate. For instance, for the 1024000Hz desired sampling-rate, data acquisition is performed with the 1052631.8Hz frequency actually.

3.1 d64h_di_config

VC6

```
short d64h_di_config(BYTE bCardID, U16 wTrigSource, U16 wExtTrigEnable, U16  
wTrigPolarity, U16 wl_REQ_Polarity)
```

VB6

```
d64h_di_config(ByVal bCardID As Byte, ByVal wTrigSource As Integer, ByVal wExtTrigEnable  
As Integer, ByVal wTrigPolarity As Integer, ByVal wl_REQ_Polarity As Integer) As Integer
```

Description:

This function configures the functionality of the following continuous digital-input.

Parameters:

`bCardID`: The specific Card ID that is configured with the on-board Dip-Switch.

`wTrigSource`: `TRIG_SOURCE_INT_PACER`, `TRIG_SOURCE_EXT_STROBE` or
`TRIG_SOURCE_HANDSHAKE`.

`wExtTrigEnable`: `DI_WAITING` or `DI_NOWAITING`.

wTrigPolarity: DI_TRIG_RISING or DI_TRIG_FALLING. This parameter is required only when wExtTrigEnable is set as DI_WAITING.

wI_REQ_Polarity: IREQ_RISING or IREQ_FALLING. This parameter is required only when wTrigSource is set as TRIG_SOURCE_EXT_STROBE or TRIG_SOURCE_HANDSHAKE.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_INVALID_MAPPED_ADDRESS: Indicates the mapped address is invalid.

ERROR_INVALID_TRIGGER_SOURCE: Indicates the invalid trigger-source is assigned to parameter **wTrigSource**.

ERROR_INVALID_TRIGGER_ENABLE: Neither DI_WAITING nor DI_NOWAITING is assigned to parameter **wExtTrigEnable**.

ERROR_INVALID_TRIGGER_POLARITY: Neither DI_TRIG_RISING nor DI_TRIG_FALLING is assigned to parameter **wTrigPolarity**.

ERROR_INVALID_IREQ_POLARITY: Neither IREQ_RISING nor IREQ_FALLING is assigned to parameter **wI_REG_Polarity**.

ERROR_DI_CONFIG: Cannot configure the digital-input operation, please call GetLastError() for further system information.

3.2 d64h_do_config

VC6

short d64h_do_config(BYTE bCardID, U16 wTrigSource, U16 wO_REQ_Enable, U16 wOutTrigHigh)

VB6

d64h_do_config(ByVal bCardID As Byte, ByVal wTrigSource As Integer, ByVal wO_REQ_Enable As Integer, ByVal wOutTrigHigh As Integer) As Integer

Description:

This function configures the functionality of the following continuous digital-output.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

wTrigSource: TRIG_SOURCE_INT_PACER or TRIG_SOURCE_HANDSHAKE.

wO_REQ_Enable: OREQ_ENABLE or OREQ_DISABLE.

wOutTrigHigh: OTRIG_HIGH or OTRIG_LOW. This parameter is required only when wO_REQ_Enable is set as OREQ_ENABLE.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_INVALID_MAPPED_ADDRESS: Indicates the mapped address is invalid.

ERROR_INVALID_TRIGGER_SOURCE: Neither TRIG_SOURCE_INT_PACER nor TRIG_SOURCE_HANDSHAKE is assigned to parameter **wTrigSource**.

ERROR_INVALID_OREQ_ENABLE: Neither OREQ_ENABLE nor OREQ_DISABLE is assigned to parameter **wO_REG_Enable**.

ERROR_INVALID_OTRIG_LEVEL: Neither DI_TRIG_RISING nor DI_TRIG_FALLING is assigned to parameter **wTigPolarity**.

ERROR_INVALID_IREQ_POLARITY: Neither OTRIG_HIGH nor OTRIG_LOW is assigned to parameter **wOutTrigHigh**.

ERROR_DO_CONFIG: Cannot configure the digital-input operation, please call GetLastError() for further system information.

Digital Input Functions

The functions in this chapter are provided for digital-input operations.

4.1 d64h_di_readport

VC6

short d64h_di_readport(BYTE bCardID, U32* pValue)

VB6

d64h_di_readport(ByVal bCardID As Byte, ByRef pValue As Long) As Integer

Description:

This function reads all digital-input, and stores the data into one 32-bit variable.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

pValue: The pointer to the 32-bit variable that stores all digital-input.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_ACCESS_VIOLATION_DATA_COPY: Some system exception occurs while copying memory, please check the pointer-type parameter you assign to this function.

ERROR_DI_PIO_READ: Cannot read the digital-input port, please call GetLastError() for further system information.

4.2 d64h_di_readline

VC6

short d64h_di_readline(BYTE bCardID, U16 wLine, U16* pValue)

VB6

d64h_di_readline(ByVal bCardID As Byte, ByVal wLine As Integer, ByRef pValue As Integer)
As Integer

Description:

This function reads specific digital-input line.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

wLine: The parameter indicates the specific line to be input.

pValue: The pointer to the 16-bit variable that stores the specific digital-input line.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_INVALID_PORT_LINE: Indicates the invalid setting is assigned to parameter **wLine**.

ERROR_ACCESS_VIOLATION_DATA_COPY: Some system exception occurs while copying memory, please check the pointer-type parameter you assign to this function.

ERROR_DI_PIO_READ: Cannot read the specific digital-input line, please call GetLastError() for further system information.

4.3 d64h_di_async_dblbuf_mode

VC6

short d64h_di_async_dblbuf_mode (BYTE bCardID, U16 wDbIBufEnable)

VB6

d64h_di_async_dblbuf_mode (ByVal bCardID As Byte, ByVal wDbIBufEnable As Integer) As Integer

Description:

This function configures the operation mode of digital-input acquisition.

Parameters:

bCardID: the specific Card ID that is configured with the on-board Dip-Switch.

wDbIBufEnable: DOUBLE_BUFFER_MODE_ENABLE or DOUBLE_BUFFER_MODE_DISABLE.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_INVALID_MAPPED_ADDRESS: Indicates the mapped address is invalid.

ERROR_INVALID_DOUBLE_BUFFER_MODE: Neither DOUBLE_BUFFER_MODE_ENABLE nor DOUBLE_BUFFER_MODE_DISABLE is assigned to parameter **wDbIBufEnable**.

ERROR_DOUBLE_BUFFER_MODE: Cannot configure the digital-input acquisition as double-buffer mode.

4.4 d64h_di_async_dblbuf_halfready

VC6

short d64h_di_async_dblbuf_halfready (BYTE bCardID, BOOLEAN *pHalfReady)

VB6

d64h_di_async_dblbuf_halfready (ByVal bCardID As Byte, ByRef pHalfReady As Byte) As Integer

Description:

This function checks the availability of ring-buffer when the digital-input acquisition is configured as double-buffer mode. If the pHalfReady indicates the ring-buffer is ready, please call d64h_di_async_dblbuf_transfer() to copy the available acquisition data.

Parameters:

bCardID: the specific Card ID that is configured with the on-board Dip-Switch.

pHalfReady: the pointer to the memory that stores the availability of ring-buffer.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_ACCESS_VIOLATION_DATA_COPY: Some system exception occurs while copying memory, please check the pointer-type parameter you assign to this function.

ERROR_DI_ASYNC_HALF_READY: Fail to check the availability of ring-buffer.

4.5 d64h_di_async_dblbuf_transfer

VC6

short d64h_di_async_dblbuf_transfer (BYTE bCardID, void *pBuffer)

VB6

d64h_di_async_dblbuf_transfer (ByVal bCardID As Byte, pBuffer As Any) As Integer

Description:

This function extract the DI acquisition data from the ready half-buffer.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

pBuffer: The pointer to the user-provided buffer that the acquisition data will be transferred to.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_INVALID_MAPPED_ADDRESS: Indicates the mapped address is invalid.

ERROR_INVALID_DOUBLE_BUFFER_MODE: Indicates the acquisition operation is NOT configured as double-buffer mode.

ERROR_ACCESS_VIOLATION_DATA_COPY: Some system exception occurs while copying memory, please check the pointer-type parameter you assign to this function.

4.6 d64h_di_async_check

VC6

short d64h_di_async_check (BYTE bCardID, BOOLEAN *pStopped, U32 *pAccessCount)

VB6

d64h_di_async_check (ByVal bCardID As Byte, ByRef Stopped As Byte, ByRef AccessCnt As Long) As Integer

Description:

This function checks if the asynchronous operation is completed. If the digital-input is completed, the number of samples will be returned.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

pStopped: The pointer to the variable that stores the status of digital-input acquisition.

pAccessCount: The pointer to the variable that stores the number of samples when the digital-input is completed.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_SYNCH_OP_WITH_ASYNC_CHECK: Indicates the digital-input acquisition is configured as SYNCNH_OP mode, and d64h_di_async_check() is conflict with this operation mode.

ERROR_ACCESS_VIOLATION_DATA_COPY: Some system exception occurs while copying memory, please check the pointer-type parameter you assign to this function.

ERROR_DI_ASYNC_CHECK: Cannot get the status of digital-input buffer, please call GetLastError() for further system information.

4.7 d64h_di_async_clear

VC6

short d64h_di_async_clear (BYTE bCardID, U32 *pAccessCount)

VB6

d64h_di_async_clear (ByVal bCardID As Byte, ByRef AccessCnt As Long) As Integer

Description:

This function terminates the in-progress digital-input acquisition, d64h_continue_readport()

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

pAccessCount: The pointer to the variable that stores the number of acquired-samples

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_ACCESS_VIOLATION_DATA_COPY: Some system exception occurs while copying memory, please check the pointer-type parameter you assign to this function.

ERROR_DI_ASYNC_CLEAR: Cannot terminate the on-going digital-input acquisition.

4.8 d64h_continue_readport

VC6

short d64h_continue_readport(BYTE bCardID, void *pBuffer, U32 dwSampleCount, F64* pSampleRate, U16 wSyncMode)

VB6

d64h_continue_readport (ByVal bCardID As Byte, pBuffer As Any, ByVal dwSampleCount As Long, ByRef pSampleRate As Double, ByVal wSyncMode As Integer) As Integer

Description:

This function starts the continuous digital-input. The 32-bit acquisition data will be recorded into driver buffer continuously.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

pBuffer: The address of user-provided DI buffer. This parameter is ignored when enabling double-buffer mode.

dwSampleCount: The samples of each acquisition.

pSampleRate: The pointer to the address that stores sampling-rate. This call-by-reference parameter passes the desired sampling-rate to library; and stores the actual sampling-rate when this function returns. (0.001 valid sampling-rate 10,000,000)

wSyncMode : The digital-input acquisition mode, SYNCH_OP or ASYNCH_OP.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_SAMPLE_COUNT_TOO_LARGE: Indicates the value of **dwSampleCount** is larger than the DI buffer in driver.

ERROR_SAMPLE_COUNT_IS_ODD: Indicate the value of **dwSampleCount** is odd, and this is conflict with the double-buffer mode

ERROR_INVALID_SAMPLE_RATE: Indicates the invalid value is assigned to address that parameter **pSampleRate** point to

ERROR_INVALID_SYNCH_OP_MODE: Neither SYNCH_OP nor ASYNCH_OP is assigned to parameter **wSyncMode**.

ERROR_SYNCH_OP_WITH_DOUBLE_BUFFER_MODE: Indicates the digital-input is configured as double-buffer mode, and the SYCNH_OP setting is conflict with this mode. Please call `d64h_di_async_dblbuf_mode()` to change the operation mode.

ERROR_INVALID_MAPPED_ADDRESS: Indicates the kernel-mapped address is invalid.

ERROR_INVALID_BUFFER_ADDRESS: Indicates the user-provided buffer is invalid.

ERROR_DMA_IN_PROGRESS: Indicates the digital-input acquisition is in progress, please call `d64h_di_async_clear()` to terminate the current acquisition.

ERROR_OVERLAP_EVENT_CREATE: Indicates the Event-Object creating is failed, please call `GetLastError()` for further system information.

ERROR_ACCESS_VIOLATION_DATA_COPY: Some system exception occurs while copying memory, please check the pointer-type parameter you assign to this function.

ERROR_DI_ASYNC_CLEAR: Cannot terminate the current acquisition, please call `GetLastError()` for further system information.

ERROR_CONTINUE_DI_START: Cannot start continuous digital-input acquisition, please call `GetLastError()` for further system information.

4.9 d64h_conti_di_status

VC6

short d64h_conti_di_status (BYTE bCardID, U16 *pStatus)

VB6

d64h_conti_di_status (ByVal bCardID As Byte, ByRef pStatus As Integer) As Integer

Description:

This function reports the buffer-overflow status of digital-input buffer.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

pStatus: The pointer to the variable that stores the status of digital-input buffer. The value in this variable will be 0 or DI_OVERRUN_STATUS.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_ACCESS_VIOLATION_DATA_COPY: Some system exception occurs while copying memory, please check the pointer-type parameter you assign to this function.

ERROR_FIFO_STATUS_GET: Cannot get the status of digital-input buffer, please call GetLastError() for further system information.

Digital Output Functions

The functions in this chapter are provided for digital-input operations.

5.1 d64h_do_writeport

VC6

short d64h_do_writeport (BYTE bCardID, U32 dwValue)

VB6

d64h_do_readport (ByVal bCardID As Byte, ByVal dwValue As Long) As Integer

Description:

This function writes all digital-output with the user-assigned data.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

dwValue: The data to be written to digital-output port.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_DO_PIO_WRITE: Cannot write the digital-output port, please call GetLastError() for further system information.

5.2 d64h_do_writeline

VC6

short d64h_do_writeline (BYTE bCardID, U16 wLine, U16 wValue)

VB6

d64h_do_writeline (ByVal bCardID As Byte, ByVal wLine As Integer, ByVal wValue As Integer)
As Integer

Description:

This function writes specific digital-input line.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

wLine: The parameter indicates the specific line to be input.

wValue: The data to be written to specific line of digital-output port.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_INVALID_PORT_LINE: Indicates the invalid setting is assigned to parameter **wLine**.

ERROR_DO_PIO_LINE_WRITE: Cannot read the specific digital-input line, please call GetLastError() for further system information.

5.3 d64h_do_readport

VC6

short d64h_do_readport(BYTE bCardID, U32* pValue)

VB6

d64h_do_readport(ByVal bCardID As Byte, ByRef pValue As Long) As Integer

Description:

This function reads all digital-output data, and stores in one 32-bit variable.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

pValue: The pointer to the 32-bit variable that stores all digital-output data.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_ACCESS_VIOLATION_DATA_COPY: Some system exception occurs while copying memory, please check the pointer-type parameter you assign to this function.

ERROR_DO_PIO_READ: Cannot read the digital-output port, please call GetLastError() for further system information.

5.4 d64h_do_readline

VC6

short d64h_do_readline(BYTE bCardID, U16 wLine, U16* pValue)

VB6

d64h_do_readline(ByVal bCardID As Byte, ByVal wLine As Integer, ByRef pValue As Integer)
As Integer

Description:

This function reads specific digital-input line.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

wLine: The parameter indicates the specific line to be output.

pValue: The pointer to the 16-bit variable that stores the specific digital-output line.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_INVALID_PORT_LINE: Indicates the invalid setting is assigned to parameter **wLine**.

ERROR_ACCESS_VIOLATION_DATA_COPY: Some system exception occurs while copying memory, please check the pointer-type parameter you assign to this function.

ERROR_DO_PIO_READ: Cannot read the specific digital-output line, please call GetLastError() for further system information.

5.5 d64h_do_async_dblbuf_mode

VC6

short d64h_do_async_dblbuf_mode (BYTE bCardID, U16 wDbIBufEnable)

VB6

d64h_do_async_dblbuf_mode (ByVal bCardID As Byte, ByVal wDbIBufEnable As Integer) As Integer

Description:

This function configures the operation mode of digital-output acquisition.

Parameters:

bCardID: the specific Card ID that is configured with the on-board Dip-Switch.

wDbIBufEnable: DOUBLE_BUFFER_MODE_ENABLE or DOUBLE_BUFFER_MODE_DISABLE.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_INVALID_MAPPED_ADDRESS: Indicates the mapped address is invalid.

ERROR_INVALID_DOUBLE_BUFFER_MODE: Neither DOUBLE_BUFFER_MODE_ENABLE nor DOUBLE_BUFFER_MODE_DISABLE is assigned to parameter **wDbIBufEnable**.

ERROR_DOUBLE_BUFFER_MODE: Cannot configure the digital-input acquisition as double-buffer mode

5.6 d64h_do_async_dblbuf_halfready

VC6

short d64h_do_async_dblbuf_halfready (BYTE bCardID, BOOLEAN *pHalfReady)

VB6

d64h_do_async_dblbuf_halfready (ByVal bCardID As Byte, ByVal pHalfReady As Byte) As Integer

Description:

This function checks the availability of ring-buffer when the digital-output data is transferred as double-buffer mode. If the pHalfReady indicates the ring-buffer is ready, the d64h_do_async_dblbuf_transfer() helps to update the output data.

Parameters:

bCardID: the specific Card ID that is configured with the on-board Dip-Switch.

pHalfReady: the pointer to the memory that stores the availability of ring-buffer.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_ACCESS_VIOLATION_DATA_COPY: Some system exception occurs while copying memory, please check the pointer-type parameter you assign to this function.

ERROR_DO_ASYNC_HALF_READY: Fail to check the availability of ring-buffer.

5.7 d64h_do_async_dblbuf_transfer

VC6

short d64h_do_async_dblbuf_transfer (BYTE bCardID, void *pBuffer)

VB6

d64h_do_async_dblbuf_transfer (ByVal bCardID As Byte, pBuffer As Any) As Integer

Description:

This function updates the DO output data to the idle half-buffer.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

pBuffer: The pointer to the user-provided buffer that contains the DO output data.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_INVALID_MAPPED_ADDRESS: Indicates the mapped address is invalid.

ERROR_INVALID_DOUBLE_BUFFER_MODE: Indicates the acquisition operation is NOT configured as double-buffer mode.

ERROR_ACCESS_VIOLATION_DATA_COPY: Some system exception occurs while copying memory, please check the pointer-type parameter you assign to this function.

5.8 d64h_do_async_check

VC6

short d64h_do_async_check (BYTE bCardID, BOOLEAN *pStopped, U32 *pAccessCount)

VB6

d64h_do_async_check (ByVal bCardID As Byte, ByRef Stopped As Byte, ByRef AccessCnt As Long) As Integer

Description:

This function checks if the asynchronous operation is completed. If the digital-output is completed, the number of samples will be returned.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

pStopped: The pointer to the variable that stores the status of digital-output operation.

pAccessCount: The pointer to the variable that stores the number of samples when the digital-output is completed.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_SYNCH_OP_WITH_ASYNC_CHECK: Indicates the digital-input acquisition is configured as SYNC_OP mode, and d64h_do_async_check() is conflict with this operation mode.

ERROR_ACCESS_VIOLATION_DATA_COPY: Some system exception occurs while copying memory, please check the pointer-type parameter you assign to this function.

ERROR_DO_ASYNC_CHECK: Cannot get the status of digital-input buffer, please call GetLastError() for further system information.

5.9 d64h_do_async_clear

VC6

short d64h_do_async_clear (BYTE bCardID, U32 *pAccessCount)

VB6

d64h_do_async_clear (ByVal bCardID As Byte, ByRef AccessCnt As Long) As Integer

Description:

This function terminates the in-progress digital-output operation, d64h_continue_writeport() and d64h_continue_pattern_write().

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

pAccessCount: The pointer to the variable that stores the number of acquired-samples

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_ACCESS_VIOLATION_DATA_COPY: Some system exception occurs while copying memory, please check the pointer-type parameter you assign to this function.

ERROR_DO_ASYNC_CLEAR: Cannot terminate the on-going digital-input acquisition.

5.10 d64h_continue_writeport

VC6

short d64h_continue_writeport (BYTE bCardID, void *pBuffer, U32 dwSampleCount, U16 wIterations, F64* pSampleRate, U16 wSyncMode)

VB6

d64h_continue_writeport (ByVal bCardID As Byte, pBuffer As Any, ByVal dwSampleCount As Long, ByVal wIterations As Integer, ByRef pSampleRate As Double, ByVal wSyncMode As Integer) As Integer

Description:

This function starts the continuous digital-input. The 32-bit acquisition data will be recorded into driver buffer continuously. If the digital-data is needed to be output repeatedly, please call d64h_do_async_dblbuf_mode() and enable double-buffer mode.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

pBuffer: The address of user-provided DO buffer.

dwSampleCount: The samples of each acquisition.

wIterations: This parameter is reserved for future use.

pSampleRate: The pointer to the address that stores sampling-rate. This call-by-reference parameter passes the desired sampling-rate to library; and stores the actual sampling-rate when this function returns. (0.001 valid sampling-rate 10,000,000)

wSyncMode : The digital-output operation mode, SYNCH_OP or ASYNCH_OP.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_SAMPLE_COUNT_TOO_LARGE: Indicates the value of **dwSampleCount** is larger than the DI buffer in driver.

ERROR_INVALID_SAMPLE_RATE: Indicates the invalid value is assigned to address that parameter **pSampleRate** point to.

ERROR_SYNCH_OP_WITH_DOUBLE_BUFFER_MODE: Indicates the digital-output is configured as double-buffer mode, and the SYCNH_OP setting is conflict with this mode. Please call d64h_do_async_dblbuf_mode() to change the operation mode.

ERROR_INVALID_SYNCH_OP_MODE: Neither SYNCH_OP nor ASYNCH_OP is assigned to parameter **wSyncMode**.

ERROR_INVALID_MAPPED_ADDRESS: Indicates the kernel-mapped address is invalid.

ERROR_INVALID_BUFFER_ADDRESS: Indicates the user-provided buffer is invalid.

ERROR_DMA_IN_PROGRESS: Indicates the digital-input acquisition is in progress, please call `d64h_do_async_clear()` to terminate the current acquisition.

ERROR_OVERLAP_EVENT_CREATE: Indicates the Event-Object creating is failed, please call `GetLastError()` for further system information.

ERROR_ACCESS_VIOLATION_DATA_COPY: Some system exception occurs while copying memory, please check the pointer-type parameter you assign to this function.

ERROR_DO_ASYNC_CLEAR: Cannot terminate the current acquisition, please call `GetLastError()` for further system information.

ERROR_CONTINUE_DO_START: Cannot start continuous digital-input acquisition, please call `GetLastError()` for further system information.

5.11 d64h_continue_pattern_write

VC6

short d64h_continue_pattern_write (BYTE bCardID, void *pBuffer, U32 dwSampleCount, F64* pSampleRate)

VB6

d64h_conti_pattern_write (ByVal bCardID As Byte, pBuffer As Any, ByVal dwSampleCount As Long, ByRef pSampleRate As Double) As Integer

Description:

This function reads the output-pattern and stores it into a circular buffer, and loops the output infinitely. This feature is supported by onboard-circuit, and neither computing-power nor bus-bandwidth is consumed.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

pBuffer: The address of user-provided DO buffer.

dwSampleCount: The samples of each acquisition. (2 < dwSampleCount ≤ 2048)

pSampleRate: The pointer to the address that stores sampling-rate. This call-by-reference parameter passes the desired sampling-rate to library; and stores the actual sampling-rate when this function returns. (0.001 ≤ valid sampling-rate ≤ 10,000,000)

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_SAMPLE_COUNT_TOO_LARGE: Indicates the value of **dwSampleCount** is larger than the DI buffer in driver.

ERROR_INVALID_SAMPLE_RATE: Indicates the invalid value is assigned to address that parameter **pSampleRate** point to.

ERROR_PATTERN_OUT_WITH_DOUBLE_BUFFER_MODE: Indicates the digital-output is configured as double-buffer mode, this mode is invalid for Patten-Output operation. Please call d64h_do_async_dblbuf_mode() to change the operation mode.

ERROR_INVALID_MAPPED_ADDRESS: Indicates the kernel-mapped address is invalid.

ERROR_INVALID_BUFFER_ADDRESS: Indicates the user-provided buffer is invalid.

ERROR_INVALID_DO_ITERATIONS: Indicates the digital-output operation is configured as SYNCH_OP, and the infinite-iteration setting is conflict with this operation mode

ERROR_DMA_IN_PROGRESS: Indicates the digital-input acquisition is in progress, please call `d64h_do_async_clear()` to terminate the current acquisition.

ERROR_OVERLAP_EVENT_CREATE: Indicates the Event-Object creating is failed, please call `GetLastError()` for further system information.

ERROR_ACCESS_VIOLATION_DATA_COPY: Some system exception occurs while copying memory, please check the pointer-type parameter you assign to this function.

ERROR_DO_ASYNC_CLEAR: Cannot terminate the current acquisition, please call `GetLastError()` for further system information.

ERROR_CONTINUE_DO_START: Cannot start continuous digital-input acquisition, please call `GetLastError()` for further system information.

5.12 d64h_conti_do_status

VC6

short d64h_conti_do_status (BYTE bCardID, U16 *pStatus)

VB6

d64h_conti_do_status (ByVal bCardID As Byte, ByRef pStatus As Integer) As Integer

Description:

This function reports the buffer-overflow status of digital-output buffer.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

pStatus: The pointer to the variable that stores the status of digital-output buffer. The value in this variable will be 0 or DO_UNDERRUN_STATUS.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_ACCESS_VIOLATION_DATA_COPY: Some system exception occurs while copying memory, please check the pointer-type parameter you assign to this function.

ERROR_FIFO_STATUS_GET: Cannot get the status of digital-output buffer, please call GetLastError() for further system information.

5.13 d64h_do_ext_trigline_write

VC6

short d64h_do_ext_trigline_write (BYTE bCardID, U16 wValue)

VB6

d64h_do_ext_trigline_write (ByVal bCardID As Byte, ByVal wValue As Integer) As Integer

Description:

This function reports the buffer-overflow status of digital-input buffer.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

wValue: The state to be set on external trigger line, OTRIG_HIGH or OTRIG_LOW.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_DO_EXT_TRIGLINE_WRITE: Cannot set the state of external trigger line, please call GetLastError() for further system information.

Digital Filter Functions

This chapter introduces the functions to configure the digital filters and the width of O_REQ pulse.

6.1 d64h_DI_filter_set

VC6

short d64h_DI_filter_set (BYTE bCardID, U16 wDIFilter)

VB6

d64h_DI_filter_set (ByVal bCardID As Byte, ByVal wDIFilter As Integer) As Integer

Description:

This function set the digital filter for all DI lines.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

wDIFilter: the delay factor for digital filter (0 wDIFilter 127).

$$wDIFilter * 25ns < \text{delay for data-latching} < (wDIFilter + 1) * 25ns$$

Notice:

Please take the input sampling-rate into consideration when you set this parameter.

Assuming the output data is sampled at 10MHz, the input data may update every 100ns.

Therefore,

$$wDIFilter * 25ns < 100ns$$

$$wDIFilter < 4$$

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_INVALID_FILTER_SETTING: Indicates the filter-setting is out of range.

ERROR_DIGITAL_FILTER_SET: Cannot set the digital filter, please call GetLastError() for further system information.

6.2 d64h_IREQ_filter_set

VC6

short d64h_IREQ_filter_set (BYTE bCardID, U16 wIREQFilter)

VB6

d64h_IREQ_filter_set (ByVal bCardID As Byte, ByVal wIREQFilter As Integer) As Integer

Description:

This function set the digital filter for I_REQ input signal.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

wIREQFilter: the delay factor for digital filter (0 wIREQFilter 127).

$wIREQFilter * 25ns < \text{delay for data-latching} < (wIREQFilter + 1) * 25ns$

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_INVALID_FILTER_SETTING: Indicates the filter-setting is out of range.

ERROR_DIGITAL_FILTER_SET: Cannot set the I_REQ filter, please call GetLastError() for further system information

6.3 d64h_OACK_filter_set

VC6

short d64h_OACK_filter_set (BYTE bCardID, U16 wOACKFilter)

VB6

d64h_OACK_filter_set (ByVal bCardID As Byte, ByVal wOACKFilter As Integer) As Integer

Description:

This function set the digital filter for O_ACK input signal.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

wOACKFilter: the delay factor for digital filter (0 wOACKFilter 127).

$wOACKFilter * 25ns < \text{delay for data-latching} < (wOACKFilter + 1) * 25ns$

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_INVALID_FILTER_SETTING: Indicates the filter-setting is out of range.

ERROR_DIGITAL_FILTER_SET: Cannot set the O_ACK filter, please call GetLastError() for further system information

6.4 d64h_ITRG_filter_set

VC6

short d64h_ITRG_filter_set (BYTE bCardID, U16 wITRGFilter)

VB6

d64h_ITRG_filter_set (ByVal bCardID As Byte, ByVal wITRGFilter As Integer) As Integer

Description:

This function set the digital filter for I_TRG input signal.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

wITRGFilter: the delay factor for digital filter (0 wITRGFilter 127).

$wITRGFilter * 25ns < \text{delay for data-latching} < (wITRGFilter + 1) * 25ns$

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_INVALID_FILTER_SETTING: Indicates the filter-setting is out of range.

ERROR_DIGITAL_FILTER_SET: Cannot set the I_TRG filter, please call GetLastError() for further system information

6.5 d64h_OREQ_width_set

VC6

short d64h_OREQ_width_set (BYTE bCardID, U16 wOREQWidth)

VB6

d64h_OREQ_width_set (ByVal bCardID As Byte, ByVal wOREQWidth As Integer) As Integer

Description:

This function set the width of O_REQ output signal.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

wOREQWidth: the delay factor for digital filter (0 wOREQWidth 7).

$$\text{Width of O_REQ} = (2^{\text{wOREQWidth} + 1}) * 25\text{ns}$$

Notice:

The period of O_REQ signal will be twice as long as the time at high-level. Please take the output update-rate into consideration when you set this parameter.

Assuming the output date is updated at 1MHz, the maximum period of O_REQ signal is 1000ns.

Therefore,

$$2 * (2^{\text{wOREQWidth} + 1}) * 25\text{ns} \leq 1000\text{ns}$$
$$2^{\text{wOREQWidth} + 1} \leq 20, \text{ wOREQWidth} < 4$$

Notice:

The period of O_REQ signal will be twice longer than the time at high-level. Therefore, please take the output update-rate into consideration when you set this parameter

The period of O_REQ signal will be twice longer than the time at high-level. Therefore, please take the output update-rate into consideration when you set this parameter.

$$(\text{O_REQ Period} > 2 * (2^{\text{wOREQWidth} + 1}) * 25\text{ns})$$

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_INVALID_OREQ_WIDTH: Indicates the width-setting is out of range.

ERROR_OREQ_WIDTH_SET: Cannot set the width of O_REQ, please call GetLastError() for further system information

Event Notification Functions

This chapter introduces the functions to register the related routine for the relevant events.

7.1 d64h_di_event_callback

VC6

```
short d64h_di_event_callback (BYTE bCardID, U16 wEventEnable, U16 wEventType, void (*callbackAddr)() )
```

Description:

This function adds/removes the event-notification of digital-input acquisition, and registers the related callback function.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

wEventEnable: ADD_NOTIFICATION or REMOVE_NOTIFICATION.

wEventType: DIEnd or DBEvent.

callbackAddr: the function-pointer of callback function.

Notice:

One internal thread will be created to monitor that specific event. Once that event is triggered by driver, the internal thread will call the function-pointer that is stored in parameter **callbackAddr**.

To avoid missing event-notification, the callback function must return before next triggered-event.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_INVALID_EVENT_ACTION: Neither ADD_NOTIFICATION nor REMOVE_NOTIFICATION is assigned to parameter **wEventEnable**.

ERROR_INVALID_CALLBACK_ADDRESS: Indicates the function-pointer of callback-function is invalid.

ERROR_INVALID_EVENT_ACTION: Neither DIEnd nor DBEvent is assigned to parameter **wEventType**.

ERROR_DI_EVENT_ATTACH: Cannot attach the notification-event to driver, please call GetLastError() for further system information.

ERROR_DI_EVENT_DETACH: Cannot detach the notification-event from driver, please call GetLastError() for further system information.

7.1 d64h_do_event_callback

VC6

short d64h_do_event_callback (BYTE bCardID, U16 wEventEnable, U16 wEventType, void (*callbackAddr)())

Description:

This function adds/removes the event-notification of digital-output operation, and registers the related callback function.

Parameters:

bCardID: The specific Card ID that is configured with the on-board Dip-Switch.

wEventEnable: ADD_NOTIFICATION or REMOVE_NOTIFICATION.

wEventType: DOEnd or DBEvent.

callbackAddr: the function-pointer of callback function. It's recommended

Notice:

One internal thread will be created to monitor that specific event. Once that event is triggered by driver, the internal thread will call the function-pointer that is stored in parameter **callbackAddr**.

To avoid missing event-notification, the callback function must return before next triggered-event.

Return Code:

SUCCESS_NO_ERROR: The function returns successfully.

ERROR_INVALID_CARD_ID: There is no active PCI-D64HU card configured with bCardID, or the given Card ID is invalid (for instance, Card ID is assigned to 254).

ERROR_INVALID_EVENT_ACTION: Neither ADD_NOTIFICATION nor REMOVE_NOTIFICATION is assigned to parameter **wEventEnable**.

ERROR_INVALID_CALLBACK_ADDRESS: Indicates the function-pointer of callback-function is invalid.

ERROR_INVALID_EVENT_ACTION: Indicates incorrect setting is assigned to parameter **wEventType**.

ERROR_DO_EVENT_ATTACH: Cannot attach the notification-event to driver, please call GetLastError() for further system information.

ERROR_DO_EVENT_DETACH: Cannot detach the notification-event from driver, please call GetLastError() for further system information.



Error Code

The Error Codes are divided into three parts: System Error, Parameter Error and Runtime Error.

SUCCESS_NO_ERROR	0
------------------	---

System Error:

ERROR_ROUTINE_FAIL_BASE	-100
ERROR_GET_CARD_ID	-101
ERROR_DEVICE_OPEN	-102
ERROR_DEVICE_CLOSE	-103
ERROR_DOUBLE_BUFFER_MODE	-104
ERROR_DI_CONFIG	-105
ERROR_DI_ASYNC_CLEAR	-106
ERROR_DI_ASYNC_CHECK	-107
ERROR_DI_ASYNC_HALF_READY	-108
ERROR_DI_PIO_READ	-109
ERROR_DO_CONFIG	-110
ERROR_DO_ASYNC_CLEAR	-111
ERROR_DO_ASYNC_CHECK	-112
ERROR_DO_PIO_READ	-113
ERROR_DO_PIO_WRITE	-114
ERROR_DO_PIO_LINE_WRITE	-115
ERROR_CONTINUE_DI_START	-116
ERROR_CONTINUE_DO_START	-117
ERROR_FIFO_STATUS_GET	-118
ERROR_DO_EXT_TRIGLINE_WRITE	-119
ERROR_DO_HALF_READY	-120
ERROR_DIGITAL_FILTER_SET	-121
ERROR_OREQ_WIDTH_SET	-122

Parameter Error:

ERROR_INVALID_PARAMETER_BASE	-200
ERROR_INVALID_CARD_ID	-201
ERROR_INVALID_SCANNED_INDEX	-202
ERROR_CARD_ID_DUPLICATED	-203
ERROR_INVALID_DOUBLE_BUFFER_MODE	-204
ERROR_INVALID_PORT_LINE	-205
ERROR_INVALID_TRIGGER_SOURCE	-206
ERROR_INVALID_TRIGGER_ENABLE	-207
ERROR_INVALID_TRIGGER_POLARITY	-208
ERROR_INVALID_IREQ_POLARITY	-209
ERROR_INVALID_OTRIG_LEVEL	-210
ERROR_INVALID_OREG_ENABLE	-211
ERROR_INVALID_DOUBLE_BUFFER_MODE	-212
ERROR_INVALID_SYNCH_OP_MODE	-213
ERROR_SAMPLE_COUNT_IS_ODD	-214
ERROR_INVALID_DO_ITERATIONS	-215
ERROR_INVALID_EVENT_ACTION	-216
ERROR_INVALID_CALLBACK_ADDRESS	-217
ERROR_INVALID_BUFFER_ADDRESS	-218
ERROR_INVALID_PATTERN_SIZE	-219
ERROR_INVALID_FILTER_SETTING	-220
ERROR_INVALID_OREG_WIDTH	-221

Runtime Error:

ERROR_RUNTIME_BASE	-300
--------------------	------

ERROR_NO_CARD_FOUND	-316
ERROR_MEMORY_MAP	-317
ERROR_MEMORY_UNMAP	-318
ERROR_ACCESS_VIOLATION_DATA_COPY	-319
ERROR_VARIABLE_PITCH_SET	-320
ERROR_DI_EVENT_ATTACH	-321
ERROR_DI_EVENT_DETACH	-322
ERROR_DO_EVENT_ATTACH	-323
ERROR_DO_EVENT_DETACH	-324
ERROR_EVENT_CREATE_FAILED	-325
ERROR_OVERLAP_EVENT_CREATE	-326
ERROR_IOCTL_FAILED	-361
ERROR_UNDEFINED_EXCEPTION	-362

DIO Runtime Error:

ERROR_DIO_RUNTIME_BASE	-500
ERROR_DEVICE_POWER_DOWN	-501
ERROR_INVALID_MAPPED_ADDRESS	-502
ERROR_BUFFER_NOT_ENOUGH	-503
ERROR_DMA_NOT_AVAILABLE	-504
ERROR_DMA_NOT_COMPLETE	-505
ERROR_DMA_NOT_STARTED	-506
ERROR_DMA_NOT_PAUSED	-507
ERROR_DMA_IS_PAUSED	-508
ERROR_DMA_IN_PROGRESS	-509
ERROR_INVALID_SAMPLE_RATE	-510
ERROR_SAMPLE_COUNT_TOO_LARGE	-511
ERROR_SYNCH_OP_WITH_DOUBLE_BUFFER_MODE	-512
ERROR_SYNCH_OP_WITH_ASYNC_CHECK	-513
ERROR_UNSUPPORTED_SETTING	-514
ERROR_PATTERN_OUT_WITH_DOUBLE_BUFFER_MODE	-515
ERROR_PATTERN_OUT_IN_PROGRESS	-516