NDS-DLL6 ver. 3.30

Dynamic Linking Library
for NuDAM 6000 Modules

Function Reference

Manual first edition: October 15, 1997
Manual Rev. 3.30: March 1, 1999

Trademarks

CONTENTS

Function Description 1

Function Description

The function calls in DLL6.DLL use intuitive names that reflect the operations they perform. For example, ND_ReadAI reads the analog input value from an analog input module.

Some sample programs are also included in this disk. It will help you to understand how to use the driver more quickly.

The detailed description of each function in the DLL driver is specified in the following sections.

ND_ClearCtr

@ Description

Clear the value of counter0 or counter1 at a counter/frequency module in the specified COM port.

@ Modules Support

6080

@ Syntax

Visual C/C++, Borland C++

```
I16 ND_ClearCtr (U16 com_port, U16 m_id, U16 counter)
```

Visual Basic

```
ND_ClearCtr (ByVal com_port As Integer, ByVal m_id As Integer, ByVal counter
        As Integer) As Integer
```

Delphi

```
ND_ClearCtr (com_port : Word; m_id : Word; counter : Word) : Integer
```

@ Argument

com_port :	The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.
m_id :	the module ID, the valid value must be within 0 and 255.
counter :	which counter to clear value. Either 0 or 1.

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND
INVALID_COUNTER

ND_ClearLatchAlarm

@ Description

Clear both the high and low alarm state at an analog input module in the specified COM port.

@ Modules Support

6011, 6012, 6014D

@ Syntax

Visual C/C++, Borland C++

I16 ND_ClearLatchAlarm (U16 com_port, U16 m_id)

Visual Basic

ND_ClearLatchAlarm (ByVal com_port As Integer, ByVal m_id As Integer) As Integer

Delphi

ND_ClearLatchAlarm (com_port : Word; m_id : Word) : Integer

@ Argument :

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.

m_id : the module ID, the valid value must be within 0 and 255.

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND

ND_ConfigLeadingCode

@ Description

Change the command leading code setting at a module in the specified COM port.

@ Modules Support

All of the ND-60XX modules

@ Syntax

Visual C/C++, Borland C++

I16 ND_ConfigLeadingCode (U16 com_port, U16 m_id, DATA leading_code_string)

Visual Basic

ND_ ConfigLeadingCode (ByVal com_port As Integer, ByVal m_id As Integer, ByVal leading_code_string As String) As Integer

Delphi

ND_ConfigLeadingCode (com_port : Word; m_id : Word; leading_code_string : PChar) : Integer

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.

m_id : the module ID, the valid value must be within 0 and 255.

leading_code_string : 6-character string represents new leading codes of six types of command.
character 1: for read configuration status, firmware version, etc. default is $.
character 2: for read synchronize sampling, digital output. default is #.
character 3: for change configuration. default is %.
character 4: for read alarm status, enable alarm, etc. default is @.
character 5: for read command leading code, change leading code, etc. default is ~.
character 6: this leading code is reserved for future use. default is *.

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND
NO_ERROR

ND_DisableAlarm

@ Description

Disable High/Low alarm functions at an analog input module in the specified COM port.

@ Modules Support

6011, 6012, 6014D

@ Syntax

Visual C/C++, Borland C++

I16 ND_DisableAlarm (U16 com_port, U16 m_id)

Visual Basic

ND_DisableAlarm (ByVal com_port As Integer, ByVal m_id As Integer) As Integer

Delphi

ND_DisableAlarm (com_port : Word; m_id : Word) : Integer

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.

m_id : the module ID, the valid value must be within 0 and 255.

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND

ND_DisableChecksum

@ Description

A checksum helps you to detect errors in communication between host and modules. If the checksum feature of an COM port is enabled, two extra checksum characters will be added to the message (command or response string) between host and modules in this COM port. This function disable the checksum feature in the specified COM port.

@ Syntax

Visual C/C++, Borland C++

I16 ND_DisableChecksum (U16 com_port)

Visual Basic

ND_DisableChecksum (ByVal com_port As Integer) As Integer

Delphi

ND_DisableChecksum (com_port : Word) : Integer

@ Argument

com_port : The COM device number that checksum status to be disabled. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.

@ Return Code
NO_ERROR

ND_DisableCtrAlarm

@ Description
Enables alarm function of counter0 or counter1 at a counter/frequency module in the specified COM port.

@ Modules Support
6080

@ Syntax

Visual C/C++, Borland C++
I16 ND_DisableCtrAlarm (U16 com_port, U16 m_id, U16 counter)

Visual Basic
ND_DisableCtrAlarm (ByVal com_port As Integer, ByVal m_id As Integer, ByVal counter As Integer) As Integer

Delphi
ND_DisableCtrAlarm (com_port : Word; m_id : Word; counter : Word):Integer

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.
m_id : the module ID, the valid value must be within 0 and 255.
counter : which counter's alarm to disable. Either 0 or 1.

@ Return Code
NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND
INVALID_COUNTER

ND_EnableAlarm

@ Description
Enable alarm to LATCH mode or MOMENTARY mode at an analog input module in the specified COM port.

@ Modules Support
6011, 6012, 6014D

@ Syntax

Visual C/C++, Borland C++
I16 ND_EnableAlarm (U16 com_port, U16 m_id, U16 alarm_type)

Visual Basic
ND_EnableAlarm (ByVal com_port As Integer, ByVal m_id As Integer, ByVal alarm_type As Integer) As Integer

Delphi
ND_EnableAlarm (com_port : Word; m_id : Word; alarm_type : Word) : Integer

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.

m_id : the module ID, the valid value must be within 0 and 255.

alarm_type : what type of alarm mode to enable. The value can be MOMENTARY or LATCH. These two constants are defined in DLL6.H (for C/C++) and DLL6.BAS (for VB), and DLL6.PAS (for Delphi).

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND

ND_EnableChecksum

@ Description

A checksum helps you to detect errors in communication between host and modules. If the checksum feature of an COM port is enabled, two extra checksum characters will be added to the message (command or response string) between host and modules in this COM port.

@ Syntax

Visual C/C++, Borland C++

I16 ND_EnableChecksum (U16 com_port)

Visual Basic

ND_EnableChecksum (ByVal com_port As Integer) As Integer

Delphi

ND_EnableChecksum (com_port : Word) : Integer

@ Argument

com_port : The COM device number that checksum status to be enabled. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.

@ Return Code

NO_ERROR

ND_EnableCtrAlarm

@ Description

Enables alarm function of counter0 or counter1 at a counter/frequency module in the specified COM port.

@ Modules Support

6080

@ Syntax

Visual C/C++, Borland C++

I16 ND_EnableCtrAlarm (U16 com_port, U16 m_id, U16 counter)

Visual Basic

ND_EnableCtrAlarm (ByVal com_port As Integer, ByVal m_id As Integer, ByVal counter As Integer) As Integer

Delphi

ND_EnableCtrAlarm (com_port : Word; m_id : Word; counter : Word) : Integer

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.

m_id : the module ID, the valid value must be within 0 and 255.

counter : which counter's alarm to enable. Either 0 or 1.

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND
INVALID_COUNTER

ND_EnableDigitalFilter

@ Description

Set the initial count value of counter0 or counter1 at a counter/frequency module in the specified COM port.

@ Modules Support

6080

@ Syntax

Visual C/C++, Borland C++

```
I16 ND_EnableDigitalFilter (U16 com_port, U16 m_id, U16 enable)
```

Visual Basic

```
ND_EnableDigitalFilter (ByVal com_port As Integer, ByVal m_id As Integer,
        ByVal enable As Integer) As Integer
```

Delphi

```
ND_EnableDigitalFilter (com_port : Word; m_id : Word; enable : Word) : Integer
```

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.

m_id : the module ID, the valid value must be within 0 and 255.

enable : **0:** disable filter
1: enable filter

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND

ND_EnableLinearMapping

@ Description

Enable or disable the linear mapping function at a ND-6014D module in the specified COM port.

@ Modules Support

6014D

@ Syntax

Visual C/C++, Borland C++

I16 ND_EnableLinearMapping (U16 com_port, U16 m_id, U16 enable)

Visual Basic

ND_EnableLinearMapping (ByVal com_port As Integer, ByVal m_id As Integer, ByVal enable As Integer) As Integer

Delphi

ND_EnableLinearMapping (com_port : Word; m_id : Word; enable : Word) : Integer

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.

m_id : the module ID, the valid value must be within 0 and 255.

enable : 1: enable; 0: disable

@ Return Code

NO_ERROR

ND_GetAlarmStatus

@ Description

Read the alarm status at a module in the specified COM port.

@ Modules Support

6011, 6012, 6014D

@ Syntax

Visual C/C++, Borland C++

I16 ND_GetAlarmStatus (U16 com_port, U16 m_id, I16 *alarm_status)

Visual Basic

ND_GetAlarmStatus (ByVal com_port As Integer, ByVal m_id As Integer, alarm_status As Integer) As Integer

Delphi

ND_GetAlarmStatus (com_port : Word; m_id : Word; var alarm_status : Word) : Integer

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.

m_id : the module ID, the valid value must be within 0 and 255.

alarm_status : alarm status read. It can be one of the following values: NOALARM, MOMENTARY, or LATCH. These constants are defined in DLL6.H (for C/C++), DLL6.BAS (for VB), and DLL6.PAS (for Delphi).

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND

ND_GetConfig

@ Description

Get the configuration of a module in the specified COM port.

@ Modules Support

All of the ND-60XX modules

@ Syntax

Visual C/C++, Borland C++

```
I16 ND_GetConfig (U16 com_port, U16 m_id, NDCONFIG *config)
```

Visual Basic

```
ND_GetConfig (ByVal com_port As Integer, ByVal m_id As Integer, config As
        NDCONFIG) As Integer
```

Delphi

```
ND_GetConfig (com_port : Word; m_id : Word; var config : NDCONFIG) : Integer
```

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.

m_id : the module ID, the valid value must be within 0 and 255.

config : the configuration data returned from the specified module. The configuration data includes the following data:

address:	the ID of module
type:	module type, such as 6011, 6012, 6014, etc.
range:	analog input range code, see NuDAM manual for detail.
baudRate:	BAUDRATE_1200, BAUDRATE_2400, BAUDRATE_4800, BAUDRATE_9600, BAUDRATE_19200, or BAUDRATE_38400.
checkSum:	checksum status. 0 for disabled, 1 for enabled.
dataFormat:	ENGINEERING_UNITS, PERCENTAGE_FSR, HEXADECIMAL, or OHMS. These values are defined in DLL6.H (for C/C++), DLL6.BAS (for VB), and DLL6.PAS (for Delphi). See NuDAM manual for detail.
alarm:	alarm status. NOALARM, MOMENTARY, or LATCH. These values are defined in DLL6.H (for C/C++), DLL6.BAS (for VB), and DLL6.PAS (for Delphi). See NuDAM manual for detail.
highLimit:	high alarm limit value
lowLimit:	low alarm limit value

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND

ND_GetCtrAlarmStatus

@ Description

Read the status of the alarm of counter0 or counter1 at a counter/frequency module in the specified COM port.

@ Modules Support

6080

@ Syntax

Visual C/C++, Borland C++

```
I16 ND_GetCtrAlarmStatus (U16 com_port, U16 m_id, U16 counter, U16
        *alarm_status)
```

Visual Basic

ND_GetCtrAlarmStatus (ByVal com_port As Integer, ByVal m_id As Integer, ByVal counter As Integer, alarm_status As Integer) As Integer

Delphi

ND_GetCtrAlarmStatus (com_port : Word; m_id : Word; counter : Word; var alarm_status : Word) : Integer

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.

m_id : the module ID, the valid value must be within 0 and 255.

counter : which counter's alarm status to read. Either 0 or 1.

alarm_status : The alarm status read.

0: disabled

1: enabled

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND
INVALID_COUNTER

ND_GetHighAlarmLimit

@ Description

Read the high alarm limit value setting at a module in the specified COM port.

@ Modules Support

6011, 6012, 6014D

@ Syntax

Visual C/C++, Borland C++

I16 ND_GetHighAlarmLimit (U16 com_port, U16 m_id, DATA hi_alarm)

Visual Basic

ND_GetHighAlarmLimit (ByVal com_port As Integer, ByVal m_id As Integer, ByVal hi_alarm As String) As Integer

Delphi

ND_GetHighAlarmLimit (com_port : Word; m_id : Word; hi_alarm : PChar) : Integer

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.

m_id : the module ID, the valid value must be within 0 and 255.

hi_alarm : the string contained the high alarm limit value read. The content of *hi_alarm* is in engineering units format. Please refer to the chapter of "Data Format and Input Range" on NuDAM's manual.

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND

ND_GetLowAlarmLimit

@ Description
Read the low alarm limit value setting at a module in the specified COM port.

@ Modules Support
6011, 6012, 6014D

@ Syntax

Visual C/C++, Borland C++
```
I16 ND_GetLowAlarmLimit (U16 com_port, U16 m_id, DATA low_alarm)
```

Visual Basic
```
ND_GetLowAlarmLimit (ByVal com_port As Integer, ByVal m_id As Integer,
        ByVal low_alarm As String) As Integer
```

Delphi
```
ND_GetLowAlarmLimit (com_port : Word; m_id : Word; lo_alarm : PChar) :
        Integer
```

@ Argument

com_port :	The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.
m_id :	the module ID, the valid value must be within 0 and 255.
low_alarm :	the string contained the low alarm limit value read. The content of *low_alarm* is in engineering units format. Please refer to the chapter of "Data Format and Input Range" on NuDAM's manual.

@ Return Code
NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND

ND_GetModuleName

@ Description
Read the module name from a module in the specified COM port.

@ Modules Support
All of the ND-60XX modules

@ Syntax

Visual C/C++, Borland C++
```
I16 ND_GetModuleName (U16 com_port, U16 m_id, DATA module_name)
```

Visual Basic
```
ND_GetModuleName (ByVal com_port As Integer, ByVal m_id As Integer,
        ByVal module_name As String) As Integer
```

Delphi
```
ND_GetModuleName (com_port : Word; m_id : Word; module_name : PChar) :
        Integer
```

@ Argument

com_port :	The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.
m_id :	the module ID, the valid value must be within 0 and 255.
module_name :	NuDAM module's name. such as "6011", "6012", "6014D", etc.

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND

ND_GetVersion

@ Description

Read the firmware version from a module in the specified COM port.

@ Modules Support

All of the ND-60XX modules

@ Syntax

Visual C/C++, Borland C++

I16 ND_GetVersion (U16 com_port, U16 m_id, DATA version)

Visual Basic

ND_GetVersion (ByVal com_port As Integer, ByVal m_id As Integer, ByVal version As String) As Integer

Delphi

ND_GetVersion (com_port : Word; m_id : Word; version : PChar) : Integer

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.
m_id : the module ID, the valid value must be within 0 and 255.
version : NuDAM module's firmware version.

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND

ND_GetWDT

@ Description

Read the host watchdog timer and safety value setting at a module in the specified COM port.

@ Modules Support

All of the ND-60XX modules

@ Syntax

Visual C/C++, Borland C++

I16 ND_GetWDT (U16 com_port, U16 m_id, U16 *enable, U16 *timeout, U16 *safe_state)

Visual Basic

ND_GetWDT (ByVal com_port As Integer, ByVal m_id As Integer, enable As Integer, timeout As Integer, safe_state As Integer) As Integer

Delphi

ND_GetWDT (com_port : Word; m_id : Word; var enable : Word; var timeout : Word; var safe_state : Word) : Integer

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.

m_id : the module ID, the valid value must be within 0 and 255.

enable : 1: host watchdog timer is enabled
0: host watchdog timer is disabled

timeout : host time-out value setting. The value should be within 0 and 255. One unit is 53.3 ms.

safe_state : 2-channel safety value of digital output channels when host is failed. The value should be within 0 and 3.

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND

ND_HostOK

@ Description

When host watchdog timer is enabled, host computer must use this function to send ‘Host is OK’ message to every module before timeout, otherwise the output value of modules with host watchdog timer enabled will go to safety state output value.

@ Syntax

Visual C/C++, Borland C++

I16 ND_HostOK (U16 com_port)

Visual Basic

ND_HostOK (ByVal com_port As Integer) As Integer

Delphi

ND_HostOK (com_port : Word) : Integer

@ Argument

com_port : The COM port that host computer want to send ‘Host is OK’ message.

@ Return Code

NO_ERROR
COMMUNICATION_ERROR

ND_InitialComm

@ Description

Open and initialize a COM port for connecting NuDAM modules.

@ Syntax

Visual C/C++, Borland C++

I16 ND_InitialComm (U16 com_port, U16 baud_rate, U16 data_bits, U16 parity, U16 stop_bits)

Visual Basic

ND_InitialComm (ByVal com_port As Integer, ByVal baud_rate As Integer, ByVal data_bits as Integer, ByVal parity As Integer, ByVal stop_bits As Integer) As Integer

Delphi

ND_InitialComm (com_port : Word; baud_rate : Word; data_bits : Word; parity : Word; stop_bits : Word) : Integer

@ Argument

comm_port : The device number of COM port to be initialized. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.

baud_rate : communication baud rate. The valid values are BAUDRATE_1200, BAUDRATE_2400, BAUDRATE_4800, BAUDRATE_9600, BAUDRATE_19200, and BAUDRATE_38400. These values are defined in file DLL6.H (for C/C++), DLL6.BAS (for VB), and DLL6.PAS (for Delphi).

data_bits : number of bits that represent data in each byte. data_bits must be within 4 and 8.

parity : parity check setting. The valid values are NOPARITY, ODDPARITY, and EVENPARITY. These values are defined in file DLL6.H (for C/C++), DLL6.BAS (for VB), and DLL6.PAS (for Delphi).

stop_bits : The valid values are ONESTOPBIT, ONE5STOPBITS, and TWOSTOPBITS. These values are defined in file DLL6.H (for C/C++), DLL6.BAS (for VB), and DLL6.PAS (for Delphi).

Note: The current release of NDS-DLL6 only supports the configuration of 8 data-bits, no parity, one stop-bits.

@ Return Code

NO_ERROR
DEVICE_ALREADY_OPEN
DEVICE_NOT_OPEN
MEMORY_ERROR
DEVICE_LOCK
INVALID_BAUDRATE

ND_ReadAI

@ Description

Read the analog input value from an analog input module in the specified COM port.

@ Modules Support

6011, 6012, 6013, 6014D

@ Syntax

Visual C/C++, Borland C++

I16 ND_ReadAI (U16 com_port, U16 m_id, DATA data)

Visual Basic

ND_ReadAI (ByVal com_port As Integer, ByVal m_id As Integer, ByVal data_str As String) As Integer

Delphi

ND_ReadAI (com_port : Word; m_id : Word; data : PChar) : Integer

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.

m_id : the module ID, the valid value must be within 0 and 255.

data : the string contained the data read. The content of *data* is in the configured data format of this module.

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND

ND_ReadAllAI

@ Description

Read all the enable analog input channel value from an analog input module in the specified COM port.

@ Modules Support

6013

@ Syntax

Visual C/C++, Borland C++

I16 ND_ReadAllAI (U16 com_port, U16 m_id, DATA data0, DATA data1, DATA data2)

Visual Basic

ND_ReadAllAI (ByVal com_port As Integer, ByVal m_id As Integer, ByVal data0 As String, ByVal data1 As String, ByVal data2 As String) As Integer

Delphi

ND_ReadAllAI (com_port : Word; m_id : Word; data0 : Pchar; data1 : PChar; data2 : PChar) : Integer

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.
m_id : the module ID, the valid value must be within 0 and 255.
data0 : the string contained the data read from channel 0. The format of *data0* is a + or – sign with five decimal digits and a fixed decimal point.
data1 : the string contained the data read from channel 1. The format of *data1* is a + or – sign with five decimal digits and a fixed decimal point.
data2 : the string contained the data read from channel 2. The format of *data2* is a + or – sign with five decimal digits and a fixed decimal point.

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND

ND_ReadBackAO

@ Description

Return the last analog output value of the specified analog output module connected to specified COM port. If the module never perform the analog data output operation, then it returns the start-up output value.

@ Modules Support

6021

@ Syntax

Visual C/C++, Borland C++

I16 ND_ReadBackAO (U16 com_port, U16 m_id, DATA data)

Visual Basic

ND_ReadSyncData (ByVal com_port As Integer, ByVal m_id As Integer, ByVal data_str As String) As Integer

Delphi

ND_ReadBackAO (com_port : Word; m_id : Word; data : PChar) : Integer

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.

m_id : the module ID, the valid value must be within 0 and 255.

data : the string contained the last analog output data readback. The content of *data* is in the configured data format of this module. Please refer to the chapter of "Data Format and Input Range" on NuDAM's manual.

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND

ND_ReadBackDO

@ Description

Readback the digital output channel value from a module in the specified COM port.

@ Modules Support

6011, 6012, 6014D, 6050, 6060, 6080

@ Syntax

Visual C/C++, Borland C++

I16 ND_ReadBackDO (U16 com_port, U16 m_id, U16 *do_data, U16 module_type)

Visual Basic

ND_ReadBackDO (ByVal com_port As Integer, ByVal m_id As Integer, do_data As Integer, ByVal module_type As Integer) As Integer

Delphi

ND_ReadBackDO (com_port : Word; m_id : Word; var do_data : Word; module_type : Word) : Integer

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.

m_id : the module ID, the valid value must be within 0 and 255.

do_data : the value of digital output channel.

module_type : the module type of the specified module, e.g. 6011, 6012, 6014, or 6050, etc.

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR

INVALID_COMMAND

ND_ReadChanAI

@ Description

Read the analog input value of a specified A/D channel from an analog input module in the specified COM port.

@ Modules Support

6017, 6018

@ Syntax

Visual C/C++, Borland C++

I16 ND_ReadChanAI (U16 com_port, U16 m_id, U16 ch_no, DATA data)

Visual Basic

ND_ReadChanAI (ByVal com_port As Integer, ByVal m_id As Integer, ByVal ch_no As Integer, ByVal data_str As String) As Integer

Delphi

ND_ReadChanAI (com_port : Word; m_id : Word; ch_no : Word; data : PChar) : Integer

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.

m_id : the module ID, the valid value must be within 0 and 255.

ch_no : analog input channel number.

data : the string contained the analog input data read. It should contain 7 characters. It consists an + or - sign followed by five decimal digits and a decimal fixed point. The format is engineering units.

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND

ND_ReadCJC

@ Description

Read the CJC (Cold Junction Compensation) sensors data at an analog input module in the specified COM port.

@ Modules Support

6011, 6018

@ Syntax

Visual C/C++, Borland C++

I16 ND_ReadCJC (U16 com_port, U16 m_id, DATA cjc_data)

Visual Basic

ND_ReadCJC (ByVal com_port As Integer, ByVal m_id As Integer, ByVal cjc_data As String) As Integer

Delphi

ND_ReadCJC (com_port : Word; m_id : Word; cjc_data : PChar) : Integer

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.
m_id : the module ID, the valid value must be within 0 and 255.
cjc_data : the string contained the CJC sensor's data read. It should contain 7 characters. It consists an + or - sign followed by five decimal digits and a decimal fixed point. The format is engineering units, that is, in degrees Celsius. The resolution is 0.1°C.

@ Return Code
NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND

ND_ReadCtrAlarmLimit

@ Description
Read the alarm limit value of counter0 or counter1 at a counter/frequency module in the specified COM port.

@ Modules Support
6080

@ Syntax
Visual C/C++, Borland C++
I16 ND_ReadCtrAlarmLimit (U16 com_port, U16 m_id, U16 counter, U32 *alarm_value)

Visual Basic
ND_ReadCtrAlarmLimit (ByVal com_port As Integer, ByVal m_id As Integer, ByVal counter As Integer, alarm_value As Long) As Integer

Delphi
ND_ReadCtrAlarmLimit (com_port : Word; m_id : Word; counter : Word; var alarm_value : Cardinal) : Integer

@ Argument
com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.
m_id : the module ID, the valid value must be within 0 and 255.
counter : which counter's alarm limit value to read. Either 0 or 1.
alarm_value : The alarm limit value read.

@ Return Code
NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND
INVALID_COUNTER

ND_ReadCtrGateMode

@ Description
Read the gate status at a counter/frequency module in the specified COM port.

@ Modules Support
6080

@ Syntax

Visual C/C++, Borland C++
I16 ND_ReadCtrGateMode (U16 com_port, U16 m_id, U16 *gate_mode)

Visual Basic
ND_ReadCtrGateMode (ByVal com_port As Integer, ByVal m_id As Integer, gate_mode As Integer) As Integer

Delphi
ND_ReadCtrGateMode (com_port : Word; m_id : Word; var gate_mode : Word) : Integer

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.
m_id : the module ID, the valid value must be within 0 and 255.
gate_mode : the gate mode read.
0: the gate is low
1: the gate is high
2: the gate is disable

@ Return Code
NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND

ND_ReadCtrInitialValue

@ Description
Read the initial count value of counter0 or counter1 at a counter/frequency module in the specified COM port.

@ Modules Support
6080

@ Syntax

Visual C/C++, Borland C++
I16 ND_ReadCtrInitialValue (U16 com_port, U16 m_id, U16 counter, U32 *init_value)

Visual Basic
ND_ReadCtrInitialValue (ByVal com_port As Integer, ByVal m_id As Integer, ByVal counter As Integer, init_value As Long) As Integer

Delphi
ND_ReadCtrInitialValue (com_port : Word; m_id : Word; counter : Word; var init_value : Cardinal) : Integer

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.
m_id : the module ID, the valid value must be within 0 and 255.
counter : which counter to read. Either 0 or 1.
init_value : the initial count value read.

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND
INVALID_COUNTER

ND_ReadCtrInputMode

@ Description

Read the input signal mode at a counter/frequency module in the specified COM port.

@ Modules Support

6080

@ Syntax

Visual C/C++, Borland C++

```
I16 ND_ReadCtrInputMode (U16 com_port, U16 m_id, U16  *input_mode)
```

Visual Basic

```
ND_ReadCtrInputMode (ByVal com_port As Integer, ByVal m_id As Integer,
        input_mode As Integer) As Integer
```

Delphi

```
ND_ReadCtrInputMode (com_port : Word; m_id : Word; var input_mode : Word) :
        Integer
```

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.

m_id : the module ID, the valid value must be within 0 and 255.

input_mode : the input signal mode read.
0: TTL input
1: photo isolated input

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND

ND_ReadCtrMaxValue

@ Description

Read the maximum counter value of counter0 or counter1 at a counter/frequency module in the specified COM port.

@ Modules Support

6080

@ Syntax

Visual C/C++, Borland C++

```
I16 ND_ReadCtrMaxValue (U16 com_port, U16 m_id, U16 counter, U32
        *max_value)
```

Visual Basic

ND_ReadCtrMaxValue (ByVal com_port As Integer, ByVal m_id As Integer, ByVal counter As Integer, max_value As Long) As Integer

Delphi

ND_ReadCtrMaxValue (com_port : Word; m_id : Word; counter : Word; var max_value : Cardinal) : Integer

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.
m_id : the module ID, the valid value must be within 0 and 255.
counter : which counter to read. Either 0 or 1.
max_value : the maximum counter value read.

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND
INVALID_COUNTER

ND_ReadCtrOverflowFlag

@ Description

Read the status of the overflow flag of counter0 or counter1 at a counter/frequency module in the specified COM port.

@ Modules Support

6080

@ Syntax

Visual C/C++, Borland C++

I16 ND_ReadCtrOverflowFlag (U16 com_port, U16 m_id, U16 counter, U16 *overflow_flag)

Visual Basic

ND_ReadCtrOverflowFlag (ByVal com_port As Integer, ByVal m_id As Integer, ByVal counter As Integer, overflow_flag As Integer) As Integer

Delphi

ND_ReadCtrOverflowFlag (com_port : Word; m_id : Word; counter : Word; var overflow_flag : Word) : Integer

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.
m_id : the module ID, the valid value must be within 0 and 255.
counter : which counter to read. Either 0 or 1.
overflow_flag : 0: the overflow flag has not been set
1: the counting value has exceeded the maximum count, the overflow flag has been set

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND

INVALID_COUNTER

ND_ReadCtrStatus

@ Description

Read the active status of counter0 or counter1 at a counter/frequency module in the specified COM port.

@ Modules Support

6080

@ Syntax

Visual C/C++, Borland C++

```
I16 ND_ReadCtrStatus (U16 com_port, U16 m_id, U16 counter, U16
        *active_status)
```

Visual Basic

```
ND_ReadCtrStatus (ByVal com_port As Integer, ByVal m_id As Integer, ByVal
        counter As Integer, active_status As Integer) As Integer
```

Delphi

```
ND_ReadCtrStatus (com_port : Word; m_id : Word; counter : Word; var
        active_status : Word) : Integer
```

@ Argument

com_port :	The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.
m_id :	the module ID, the valid value must be within 0 and 255.
counter :	which counter to read. Either 0 or 1.
active_status :	**0:** stop counting **1:** start counting

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND
INVALID_COUNTER

ND_ReadCtrValue

@ Description

Read the value of counter0 or counter1 at a counter/frequency module in the specified COM port.

@ Modules Support

6080

@ Syntax

Visual C/C++, Borland C++

```
I16 ND_ReadCtrValue (U16 com_port, U16 m_id, U16 counter, U32 *ctr_value)
```

Visual Basic

```
ND_ReadCtrValue (ByVal com_port As Integer, ByVal m_id As Integer, ByVal
        counter As Integer, ctr_value As Long) As Integer
```

Delphi

ND_ReadCtrValue (com_port : Word; m_id : Word; counter : Word; var ctr_value : Cardinal) : Integer

@ Argument

com_port :	The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.
m_id :	the module ID, the valid value must be within 0 and 255.
counter :	which counter to read. Either 0 or 1.
ctr_value :	the counter value read.

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND
INVALID_COUNTER

ND_ReadDI

@ Description

Read the digital input channel value from a module in the specified COM port.

@ Modules Support

6011, 6012, 6014D, 6024, 6050, 6052, 6053, 6054, 6058, 6060

@ Syntax

Visual C/C++, Borland C++

I16 ND_ReadDI (U16 com_port, U16 m_id, U16 *di_data, U16 module_type)

Visual Basic

ND_ReadDI (ByVal com_port As Integer, ByVal m_id As Integer, di_data As Integer, ByVal module_type As Integer) As Integer

Delphi

ND_ReadDI (com_port : Word; m_id : Word; var di_data : Word; module_type : Word) : Integer

@ Argument

com_port :	The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.
m_id :	the module ID, the valid value must be within 0 and 255.
di_data :	digital input channel value read.
module_type :	the module type of the specified module, e.g. 6011, 6012, 6014, or 6050, etc.

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND

ND_ReadEventCount

@ Description

Read the event counter value at the specified analog input module in the specified COM port.

@ Modules Support

6011, 6012, 6014D

@ Syntax

Visual C/C++, Borland C++

I16 ND_ReadEventCount (U16 com_port, U16 m_id, DATA data_str)

Visual Basic

ND_ReadEventCount (ByVal com_port As Integer, ByVal m_id As Integer, ByVal data_str As String) As Integer

Delphi

ND_ReadEventCount (com_port : Word; m_id : Word; data_str : PChar) : Integer

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.

m_id : the module ID, the valid value must be within 0 and 255.

data_str : the event counter data read. It should be 5-character string, range from '00000' to '65535'.

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND

ND_ReadFilterStatus

@ Description

Read the initial count value of counter0 or counter1 at a counter/frequency module in the specified COM port.

@ Modules Support

6080

@ Syntax

Visual C/C++, Borland C++

I16 ND_ReadFilterStatus (U16 com_port, U16 m_id, U16 *filter_status)

Visual Basic

ND_ReadFilterStatus (ByVal com_port As Integer, ByVal m_id As Integer, filter_status As Integer) As Integer

Delphi

ND_ReadFilterStatus (com_port : Word; m_id : Word; var filter_status : Word) : Integer

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.

m_id : the module ID, the valid value must be within 0 and 255.

filter_status : the digital filter status read.
0: filter disabled
1: filter enabled

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND

ND_ReadMappingSource

@ Description

Read the source high/low limit values from input for linear mapping. This function is only for ND-6014D.

@ Modules Support

6014D

@ Syntax

Visual C/C++, Borland C++

I16 ND_ReadMappingSource (U16 com_port, U16 m_id, DATA low_source, DATA high_source)

Visual Basic

ND_ReadMappingSource (ByVal com_port As Integer, ByVal m_id As Integer, ByVal low_source As String, ByVal high_source As String) As Integer

Delphi

ND_ReadMappingSource (com_port : Word; m_id : Word; port : Word; do_data : Word; low_source : Pchar; high_source : PChar) : Integer

@ Argument

com_port :	The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.
m_id :	the module ID, the valid value must be within 0 and 255.
low_source :	the string contained the low limit input value for linear mapping. Please refer to NuDAM‘s manual for the format of *low_source*.
high_source :	the string contained the high limit input value for linear mapping. Please refer to NuDAM‘s manual for the format of *high_source*.

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND

ND_WriteMappingTarget

@ Description

Read the target high/low limit values from input for linear mapping. This function is only for ND-6014D.

@ Modules Support

6014D

@ Syntax

Visual C/C++, Borland C++

I16 ND_ReadMappingTarget (U16 com_port, U16 m_id, DATA low_target, DATA high_target)

Visual Basic

ND_ReadMappingTarget (ByVal com_port As Integer, ByVal m_id As Integer, ByVal low_target As String, ByVal high_target As String) As Integer

Delphi

ND_ReadMappingTarget (com_port : Word; m_id : Word; port : Word; do_data : Word; low_target : Pchar; high_target : PChar) : Integer

@ Argument

com_port :	The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.
m_id :	the module ID, the valid value must be within 0 and 255.
low_target :	the string contained the mapped low limit input value for linear mapping. Please refer to NuDAM‘s manual for the format of *low_target*.
high_target :	the string contained the mapped high limit input value for linear mapping. Please refer to NuDAM‘s manual for the format of *high_target*.

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND

ND_ReadMinWidthAtHigh

@ Description

Read the minimum input signal width at high level at a counter/frequency module in the specified COM port.

@ Modules Support

6080

@ Syntax

Visual C/C++, Borland C++

I16 ND_ReadMinWidthAtHigh (U16 com_port, U16 m_id, U16 *min_width)

Visual Basic

ND_ReadMinWidthAtHigh (ByVal com_port As Integer, ByVal m_id As Integer, min_width As Integer) As Integer

Delphi

ND_ReadMinWidthAtHigh (com_port : Word; m_id : Word; var min_width : Word) : Integer

@ Argument

com_port :	The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.
m_id :	the module ID, the valid value must be within 0 and 255.
min_width :	The minimum width data at high level. The unit is μs.

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND

ND_ReadMinWidthAtLow

@ Description

Read the minimum input signal width at low level at a counter/frequency module in the specified COM port.

@ Modules Support

6080

@ Syntax

Visual C/C++, Borland C++

I16 ND_ReadMinWidthAtLow (U16 com_port, U16 m_id, U16 *min_width)

Visual Basic

ND_ReadMinWidthAtLow (ByVal com_port As Integer, ByVal m_id As Integer, min_width As Integer) As Integer

Delphi

ND_ReadMinWidthAtLow (com_port : Word; m_id : Word; var min_width : Word) : Integer

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.

m_id : the module ID, the valid value must be within 0 and 255.

min_width : The minimum width data at low level. The unit is μs.

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND

ND_ReadSyncData

@ Description

After a synchronized sampling function **ND_Sync** was called, you can read the sampled value that was stored in the register of a module in the specified COM port.

@ Modules Support

6011, 6012, 6013, 6014D, 6050, 6052, 6053, 6054, 6058, 6060

@ Syntax

Visual C/C++, Borland C++

I16 ND_ReadSyncData (U16 com_port, U16 m_id, DATA data, U16 module_type)

Visual Basic

ND_ReadSyncData (ByVal com_port As Integer, ByVal m_id As Integer, ByVal data_str As String, ByVal module_type As Integer) As Integer

Delphi

ND_ReadSyncData (com_port : Word; m_id : Word; data : PChar; module_type : Word) : Integer

@ Argument

com_port : the COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.

m_id : the module ID, the valid value must be within 0 and 255.

data : string that stored the sampled data read. The meaning of the content depends on the module type and data format this module was configured. Please refer to the chapter of "Data Format and Input Range" on NuDAM's manual.

module_type : the module type of the specified module, e.g. 6011, 6012, 6014, or 6050, etc.

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND

ND_ReadTTLInputHigh

@ Description

Read the TTL input high trigger level at a counter/frequency module in the specified COM port.

@ Modules Support

6080

@ Syntax

Visual C/C++, Borland C++

```
I16 ND_ReadTTLInputHigh (U16 com_port, U16 m_id, U16 *high_level)
```

Visual Basic

```
ND_ReadTTLInputHigh (ByVal com_port As Integer, ByVal m_id As Integer,
        high_level As Integer) As Integer
```

Delphi

```
ND_ReadTTLInputHigh (com_port : Word; m_id : Word; var high_level : Word) :
        Integer
```

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.

m_id : the module ID, the valid value must be within 0 and 255.

high_level : The high trigger level for TTL input. The unit is 0.1V.

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND

ND_ReadTTLInputLow

@ Description

Read the TTL input low trigger level at a counter/frequency module in the specified COM port.

@ Modules Support

6080

@ Syntax

Visual C/C++, Borland C++

I16 ND_ReadTTLInputLow (U16 com_port, U16 m_id, U16 *low_level)

Visual Basic

ND_ReadTTLInputLow (ByVal com_port As Integer, ByVal m_id As Integer, low_level As Integer) As Integer

Delphi

ND_ReadTTLInputLow (com_port : Word; m_id : Word; var low_level : Word) : Integer

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.

m_id : the module ID, the valid value must be within 0 and 255.

low_level : The low trigger level for TTL input. The unit is 0.1V.

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND

ND_ReleaseComm

@ Description

Release the specified COM port for NuDAM connection.

@ Syntax

Visual C/C++, Borland C++

I16 ND_ReleaseComm (U16 com_port)

Visual Basic

ND_ReleaseComm (ByVal com_port As Integer) As Integer

Delphi

ND_ReleaseComm (com_port : Word) : Integer

@ Argument

com_port : The COM device number to be released. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.

@ Return Code

NO_ERROR
DEVICE_NOT_OPEN

ND_ResetEventCount

@ Description

Reset the event counter to zero at an analog input module in the specified COM port.

@ Modules Support

6011, 6012, 6014D

@ Syntax

Visual C/C++, Borland C++

I16 ND_ResetEventCount (U16 com_port, U16 m_id)

Visual Basic

ND_ResetEventCount (ByVal com_port As Integer, ByVal m_id As Integer) As Integer

Delphi

ND_ResetEventCount (com_port : Word; m_id : Word) : Integer

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.

m_id : the module ID, the valid value must be within 0 and 255.

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND

ND_ResetStatusAO

@ Description

Read the reset status of the specified module connected to specified COM port. Its purpose is to check whether it has been reset since the last reset status command was issued to this module.

@ Modules Support

6021, 6024

@ Syntax

Visual C/C++, Borland C++

I16 ND_ResetStatusAO (U16 com_port, U16 m_id, U16 *status)

Visual Basic

ND_ ResetStatusAO (ByVal com_port As Integer, ByVal m_id As Integer, status As Integer) As Integer

Delphi

ND_ResetStatusAO (com_port : Word; m_id : Word; var status : Word) : Integer

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.

m_id : the module ID, the valid value must be within 0 and 255.

status : **0:** It has not been reset since the last reset status command was issued.
1: It has been reset since the last reset status command was issued.

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND

ND_SendCommand

@ Description

Host uses this function to send a command string and get the response string through a specified COM port.

@ Syntax

Visual C/C++, Borland C++

I16 ND_SendCommand (U16 com_port, STRING command_str, STRING result)

Visual Basic

ND_SendCommand (ByVal com_port As Integer, ByVal command_str As String, ByVal result As String) As Integer

Delphi

ND_SendCommand (com_port : Word; command : PChar; result : PChar) : Integer

@ Argument

com_port :	The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.
command_str :	the command string want to send through the specified COM port.
result :	the response string from the module the command string addressed.

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND

ND_SetConfig

@ Description

Configure the setting of a module in the specified COM port.

@ Modules Support

All of the ND-60XX modules

@ Syntax

Visual C/C++, Borland C++

I16 ND_SetConfig (U16 com_port, U16 m_id, NDCONFIG *config)

Visual Basic

ND_SetConfig (ByVal com_port As Integer, ByVal m_id As Integer, config As NDCONFIG) As Integer

Delphi

ND_SetConfig (com_port : Word; m_id : Word; var config : NDCONFIG) : Integer

@ Argument

com_port :	The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.
m_id :	the module ID, the valid value must be within 0 and 255.
config :	the configuration setting. Please refer to the description of **`ND_GetConfig`** for the meaning of configuration data.

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR

INVALID_COMMAND

ND_SetCtrAlarmLimit

@ Description

Set the alarm limit value of counter0 or counter1 at a counter/frequency module in the specified COM port.

@ Modules Support

6080

@ Syntax

Visual C/C++, Borland C++

I16 ND_SetCtrAlarmLimit (U16 com_port, U16 m_id, U16 counter, U32 alarm_value)

Visual Basic

ND_SetCtrAlarmLimit (ByVal com_port As Integer, ByVal m_id As Integer, ByVal counter As Integer, ByVal alarm_value As Long) As Integer

Delphi

ND_SetCtrAlarmLimit (com_port : Word; m_id : Word; counter : Word; alarm_value : Cardinal) : Integer

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.
m_id : the module ID, the valid value must be within 0 and 255.
counter : which counter's alarm limit value to set. Either 0 or 1.
alarm_value : The alarm limit value.

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND
INVALID_COUNTER

ND_SetCtrGateMode

@ Description

Set the gate control mode at a counter/frequency module in the specified COM port.

@ Modules Support

6080

@ Syntax

Visual C/C++, Borland C++

I16 ND_SetCtrGateMode (U16 com_port, U16 m_id, U16 gate_mode)

Visual Basic

ND_SetCtrGateMode (ByVal com_port As Integer, ByVal m_id As Integer, ByVal gate_mode As Integer) As Integer

Delphi

ND_SetCtrGateMode (com_port : Word; m_id : Word; gate_mode : Word) : Integer

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.
m_id : the module ID, the valid value must be within 0 and 255.
gate_mode : **0:** the gate is low
1: the gate is high
2: the gate is disable

@ Return Code
NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND

ND_SetCtrInitialValue

@ Description
Set the initial count value of counter0 or counter1 at a counter/frequency module in the specified COM port.

@ Modules Support
6080

@ Syntax
Visual C/C++, Borland C++
I16 ND_SetCtrInitialValue (U16 com_port, U16 m_id, U16 counter, U32 init_value)

Visual Basic
ND_SetCtrInitialValue (ByVal com_port As Integer, ByVal m_id As Integer, ByVal counter As Integer, ByVal init_value As Long) As Integer

Delphi
ND_SetCtrInitialValue (com_port : Word; m_id : Word; counter : Word; init_value : Cardinal) : Integer

@ Argument
com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.
m_id : the module ID, the valid value must be within 0 and 255.
counter : which counter to set. Either 0 or 1.
init_value : the initial count value.

@ Return Code
NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND
INVALID_COUNTER

ND_SetCtrInputMode

@ Description
Set the input signal mode at a counter/frequency module in the specified COM port.

@ Modules Support

6080

@ Syntax

Visual C/C++, Borland C++

I16 ND_SetCtrInputMode (U16 com_port, U16 m_id, U16 input_mode)

Visual Basic

ND_SetCtrInputMode (ByVal com_port As Integer, ByVal m_id As Integer, ByVal input_mode As Integer) As Integer

Delphi

ND_SetCtrInputMode (com_port : Word; m_id : Word; input_mode : Word) : Integer

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.

m_id : the module ID, the valid value must be within 0 and 255.

input_mode : **0:** TTL input
1: photo isolated input

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND

ND_SetCtrMaxValue

@ Description

Set the maximum counter value of counter0 or counter1 at a counter/frequency module in the specified COM port.

@ Modules Support

6080

@ Syntax

Visual C/C++, Borland C++

I16 ND_SetCtrMaxValue (U16 com_port, U16 m_id, U16 counter, U32 max_value)

Visual Basic

ND_SetCtrMaxValue (ByVal com_port As Integer, ByVal m_id As Integer, ByVal counter As Integer, ByVal max_value As Long) As Integer

Delphi

ND_SetCtrMaxValue (com_port : Word; m_id : Word; counter : Word; max_value : Cardinal) : Integer

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.

m_id : the module ID, the valid value must be within 0 and 255.

counter : which counter to set. Either 0 or 1.

max_value : the maximum counter value.

@ Return Code

NO_ERROR
COMMUNICATION_ERROR

COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND
INVALID_COUNTER

ND_SetHighAlarmLimit

@ Description
Set high alarm limit value at a module in the specified COM port.

@ Modules Support
6011, 6012, 6014D

@ Syntax

Visual C/C++, Borland C++
I16 ND_SetHighAlarmLimit (U16 com_port, U16 m_id, DATA hi_alarm)

Visual Basic
ND_SetHighAlarmLimit (ByVal com_port As Integer, ByVal m_id As Integer, ByVal hi_alarm As String) As Integer

Delphi
ND_SetHighAlarmLimit (com_port : Word; m_id : Word; hi_alarm : PChar) : Integer

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.
m_id : the module ID, the valid value must be within 0 and 255.
hi_alarm : the string contained the high alarm limit value to set. The content of *hi_alarm* is in engineering units format. Please refer to the chapter of "Data Format and Input Range" on NuDAM's manual.

@ Return Code
NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND

ND_SetLowAlarmLimit

@ Description
Set low alarm limit value at a module in the specified COM port.

@ Modules Support
6011, 6012, 6014D

@ Syntax

Visual C/C++, Borland C++
I16 ND_SetLowAlarmLimit (U16 com_port, U16 m_id, DATA low_alarm)

Visual Basic
ND_SetLowAlarmLimit (ByVal com_port As Integer, ByVal m_id As Integer, ByVal low_alarm As String) As Integer

Delphi
ND_SetLowAlarmLimit (com_port : Word; m_id : Word; low_alarm : PChar) : Integer

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.

m_id : the module ID, the valid value must be within 0 and 255.

low_alarm : the string contained the low alarm limit value to set. The content of *low_alarm* is in engineering units format. Please refer to the chapter of "Data Format and Input Range" on NuDAM's manual.

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND

ND_SetMinWidthAtHigh

@ Description

Set the minimum input signal width at high level at a counter/frequency module in the specified COM port.

@ Modules Support

6080

@ Syntax

Visual C/C++, Borland C++

I16 ND_SetMinWidthAtHigh (U16 com_port, U16 m_id, U16 min_width)

Visual Basic

ND_SetMinWidthAtHigh (ByVal com_port As Integer, ByVal m_id As Integer, ByVal min_width As Integer) As Integer

Delphi

ND_SetMinWidthAtHigh (com_port : Word; m_id : Word; min_width : Word) : Integer

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.

m_id : the module ID, the valid value must be within 0 and 255.

min_width : The minimum width data at high level. The unit is μs. The value should be within 4 and 1020.

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND

ND_SetMinWidthAtLow

@ Description

Set the minimum input signal width at low level at a counter/frequency module in the specified COM port.

@ Modules Support

6080

@ Syntax

Visual C/C++, Borland C++

I16 ND_SetMinWidthAtLow (U16 com_port, U16 m_id, U16 min_width)

Visual Basic

ND_SetMinWidthAtLow (ByVal com_port As Integer, ByVal m_id As Integer, ByVal min_width As Integer) As Integer

Delphi

ND_SetMinWidthAtLow (com_port : Word; m_id : Word; min_width : Word) : Integer

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.

m_id : the module ID, the valid value must be within 0 and 255.

min_width : The minimum width data at low level. The unit is μs. The value should be within 4 and 1020.

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND

ND_SetMUXChans

@ Description

Enable channels for multiplexing.

@ Modules Support

6013, 6017, 6018

@ Syntax

Visual C/C++, Borland C++

I16 ND_SetMUXChans (U16 com_port, U16 m_id, U16 enable_chans)

Visual Basic

ND_SetMUXChans (ByVal com_port As Integer, ByVal m_id As Integer, ByVal enable_chans As Integer) As Integer

Delphi

ND_SetMUXChans (com_port : Word; m_id : Word; enable_chans : Word) : Integer

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.

m_id : the module ID, the valid value must be within 0 and 255.

enable_chans : the channels to be enabled. This argument is an integer expression formed from one or more channel constants defined in DLL6.H (for C/C++), DLL6.BAS (for VB), and DLL6.PAS (for Delphi). If more than one channel want to be enabled, the constants are combined with the bitwise-OR operator (|) or addition operator (+). The DLL6.H, DLL6.BAS, or DLL6.PAS file defines the following channel constants: CHANNEL0, CHANNEL1,

CHANNEL2, CHANNEL3, CHANNEL4, CHANNEL5, CHANNEL6, and CHANNEL7.

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND

ND_SetTTLInputHigh

@ Description

Set the TTL input high trigger level at a counter/frequency module in the specified COM port.

@ Modules Support

6080

@ Syntax

Visual C/C++, Borland C++

```
I16 ND_SetTTLInputHigh (U16 com_port, U16 m_id, U16 high_level)
```

Visual Basic

```
ND_SetTTLInputHigh (ByVal com_port As Integer, ByVal m_id As Integer, ByVal
        high_level As Integer) As Integer
```

Delphi

```
ND_SetTTLInputHigh (com_port : Word; m_id : Word; high_level : Word) : Integer
```

@ Argument

com_port :	The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.
m_id :	the module ID, the valid value must be within 0 and 255.
high_level :	The high trigger level for TTL input. The unit is 0.1V. The value should be within 1 and 50 (0.1V to 5V).

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND

ND_SetTTLInputLow

@ Description

Set the TTL input low trigger level at a counter/frequency module in the specified COM port.

@ Modules Support

6080

@ Syntax

Visual C/C++, Borland C++

```
I16 ND_SetTTLInputLow (U16 com_port, U16 m_id, U16 low_level)
```

Visual Basic

ND_SetTTLInputLow (ByVal com_port As Integer, ByVal m_id As Integer, ByVal low_level As Integer) As Integer

Delphi

ND_SetTTLInputLow (com_port : Word; m_id : Word; low_level : Word) : Integer

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.

m_id : the module ID, the valid value must be within 0 and 255.

low_level : The low trigger level for TTL input. The unit is 0.1V. The value should be within 1 and 50 (0.1V to 5V).

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND

ND_SetWDT

@ Description

Enable/disable host watchdog timer, this module will change to safety state when host is failed, that is, host did not send ' Host is OK' command to this module before time-out.

@ Modules Support

All of the ND-60XX modules

@ Syntax

Visual C/C++, Borland C++

I16 ND_SetWDT (U16 com_port, U16 m_id, U16 enable, U16 timeout, U16 safe_state)

Visual Basic

ND_SetWDT (ByVal com_port As Integer, ByVal m_id As Integer, ByVal enable As Integer, ByVal timeout As Integer, ByVal safe_state As Integer) As Integer

Delphi

ND_SetWDT (com_port : Word; m_id : Word; enable : Word; timeout : Word; safe_state : Word) : Integer

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.

m_id : the module ID, the valid value must be within 0 and 255.

enable : 1: enable host watchdog timer
0: disable host watchdog timer

timeout : host time-out value, between this time period host must send ' Host is OK' command to this module, otherwise this module will change to safety state. The valid value must be within 0 and 255. One unit is 53.3 ms.

safe_state : 2-channel safety value of digital output channels when host is failed. The valid values are between 0 and 3.

@ Return Code

NO_ERROR

COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND

ND_StartCtr

@ Description

Start or stop the counting of counter0 or counter1 at a counter/frequency module in the specified COM port.

@ Modules Support

6080

@ Syntax

Visual C/C++, Borland C++

I16 ND_StartCtr (U16 com_port, U16 m_id, U16 counter, U16 status)

Visual Basic

ND_StartCtr (ByVal com_port As Integer, ByVal m_id As Integer, ByVal counter As Integer, ByVal status As Integer) As Integer

Delphi

ND_StartCtr (com_port : Word; m_id : Word; counter : Word; status : Word) : Integer

@ Argument

com_port :	The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.
m_id :	the module ID, the valid value must be within 0 and 255.
counter :	which counter to start/stop. Either 0 or 1.
status :	**0:** stop counting **1:** start counting

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND
INVALID_COUNTER

ND_Sync

@ Description

Synchronize all modules connected to the specified COM port to sample analog input or Digital I/O values and stored the values in the module‘s register at the same time. The sampled data can be read by **`ND_ReadSyncData`** function call.

@ Syntax

Visual C/C++, Borland C++

I16 ND_Sync (U16 com_port)

Visual Basic

ND_Sync (ByVal com_port As Integer) As Integer

Delphi

ND_Sync (com_port : Word) : Integer

@ Argument

com_port : COM device number. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.

@ Return Code

NO_ERROR
COMMUNICATION_ERROR

ND_TimeOut

@ Description

Set the time-out value of host waiting the modules in a specified COM port to response.

@ Syntax

Visual C/C++, Borland C++

I16 ND_TimeOut (U16 com_port, U16 m_sec)

Visual Basic

ND_TimeOut (ByVal com_port As Integer, ByVal m_sec As Integer) As Integer

Delphi

ND_TimeOut (com_port : Word; m_sec : Word) : Integer

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.

m_sec : time-out value in millisecond.

@ Return Code

NO_ERROR

ND_WriteAO

@ Description

Write the analog output value to an analog output module in the specified COM port.

@ Modules Support

6021

@ Syntax

Visual C/C++, Borland C++

I16 ND_WriteAO (U16 com_port, U16 m_id, DATA data)

Visual Basic

ND_WtiteAO (ByVal com_port As Integer, ByVal m_id As Integer, ByVal data_str As String) As Integer

Delphi

ND_WriteAO (com_port : Word; m_id : Word; data : PChar) : Integer

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.

m_id : the module ID, the valid value must be within 0 and 255.

data : the string contained the data to be written. The content of *data* is in the configured data format of this module. Please refer to the chapter of "Data Format and Input Range" on NuDAM's manual.

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND

ND_WriteChanAO

@ Description

Write the analog output value to a specified AO port at an analog output module in the specified COM port.

@ Modules Support

6024

@ Syntax

Visual C/C++, Borland C++

I16 ND_WriteAO (U16 com_port, U16 m_id, U16 ch_no, DATA data)

Visual Basic

ND_WtiteAO (ByVal com_port As Integer, ByVal m_id As Integer, ByVal ch_no As Integer, ByVal data_str As String) As Integer

Delphi

ND_WriteAO (com_port : Word; m_id : Word; ch_no : Word; data : PChar) : Integer

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.
m_id : the module ID, the valid value must be within 0 and 255.
ch_no : 0, 1, 2, and 3 indicates port A, B, C, and D respectively.
data : the string contained the data to be written. The content of *data* is in the configured data format of this module. Please refer to the chapter of "Data Format and Input Range" on NuDAM's manual.

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND

ND_WriteDOLine

@ Description

Set digital output line value at the specified module in the specified COM port.

@ Modules Support

6050, 6056, 6058, 6060, 6063

@ Syntax

Visual C/C++, Borland C++

I16 ND_WriteDOLine (U16 com_port, U16 m_id, U16 port, U16 line, U16 do_data, U16 module_type)

Visual Basic

ND_WriteDOLine (ByVal com_port As Integer, ByVal m_id As Integer, ByVal port As Integer, ByVal line As Integer, ByVal do_data As Integer, ByVal module_type As Integer) As Integer

Delphi

ND_WriteDOLine (com_port : Word; m_id : Word; port : Word; line : Word; do_data : Word; module_type : Word) : Integer

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.

m_id : the module ID, the valid value must be within 0 and 255.

port : for modules other than 6056 and 6058, you can ignore this argument.
6056: 0: low port; 1: high port
6058: 0: port A; 1: port B; 2: port C

line : digital output line (channel)
6050, 6056, 6058, 6063: 0 ~ 7
6060: 0 ~ 3

do_data : digital output data. For ND-6050 and ND-6060, this function will set the value to all channels. For example, if do_value is 0x03 (00000011), channel 0 and 1 are set ON, the other channels are set OFF.

module_type : the module type of the specified module, e.g. 6050, 6056, or 6060, etc.

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND
BAD_DATA_VALUE

ND_WriteDOPort

@ Description

Set digital output port value at the specified module in the specified COM port.

@ Modules Support

6011, 6012, 6014D, 6050, 6056, 6058, 6060, 6063, 6080

@ Syntax

Visual C/C++, Borland C++

I16 ND_WriteDOPort (U16 com_port, U16 m_id, U16 port, U16 do_data, U16 module_type)

Visual Basic

ND_WriteDOPort (ByVal com_port As Integer, ByVal m_id As Integer, ByVal port As Integer, ByVal do_data As Integer, ByVal module_type As Integer) As Integer

Delphi

ND_WriteDOPort (com_port : Word; m_id : Word; port : Word; do_data : Word; module_type : Word) : Integer

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.

m_id : the module ID, the valid value must be within 0 and 255.
port : for modules other than 6056 and 6058, you can ignore this argument.
6056: 0: low port; 1: high port
6058: 0: port A; 1: port B; 2: port C
do_data : digital output data. For ND-6050 and ND-6060, this function will set the value to all channels. For example, if do_value is 0x03 (00000011), channel 0 and 1 are set ON, the other channels are set OFF.
module_type : the module type of the specified module, e.g. 6011, 6012, 6014, or 6050, etc.

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND
BAD_DATA_VALUE

ND_WriteMappingSource

@ Description

Write the source high/low limit values from input for linear mapping. This function is only for ND-6014D.

@ Modules Support

6014D

@ Syntax

Visual C/C++, Borland C++

I16 ND_WriteMappingSource (U16 com_port, U16 m_id, DATA low_source, DATA high_source)

Visual Basic

ND_WriteMappingSource (ByVal com_port As Integer, ByVal m_id As Integer, ByVal low_source As String, ByVal high_source As String) As Integer

Delphi

ND_WriteMappingSource (com_port : Word; m_id : Word; port : Word; do_data : Word; low_source : Pchar; high_source : PChar) : Integer

@ Argument

com_port : The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.
m_id : the module ID, the valid value must be within 0 and 255.
low_source : the string contained the low limit input value for linear mapping. Please refer to NuDAM‘s manual for the format of *low_source*.
high_source : the string contained the high limit input value for linear mapping. Please refer to NuDAM‘s manual for the format of *high_source*.

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND

ND_WriteMappingTarget

@ Description

Write the target high/low limit values from input for linear mapping. This function is only for ND-6014D.

@ Modules Support

6014D

@ Syntax

Visual C/C++, Borland C++

I16 ND_WriteMappingTarget (U16 com_port, U16 m_id, DATA low_target, DATA high_target)

Visual Basic

ND_WriteMappingTarget (ByVal com_port As Integer, ByVal m_id As Integer, ByVal low_target As String, ByVal high_target As String) As Integer

Delphi

ND_WriteMappingTarget (com_port : Word; m_id : Word; port : Word; do_data : Word; low_target : Pchar; high_target : PChar) : Integer

@ Argument

com_port :	The COM port that the module connected. The valid values are within 1 and 4. 1, 2, 3, and 4 represent COM1, COM2, COM3, and COM4 respectively.
m_id :	the module ID, the valid value must be within 0 and 255.
low_target :	the string contained the mapped low limit input value for linear mapping. Please refer to NuDAM‘s manual for the format of *low_target*.
high_target :	the string contained the mapped high limit input value for linear mapping. Please refer to NuDAM‘s manual for the format of *high_target*.

@ Return Code

NO_ERROR
COMMUNICATION_ERROR
COMMUNICATION_TIMEOUT
CHECKSUM_ERROR
INVALID_COMMAND

Appendix Function Support

This appendix shows which modules each NDS-DLL6 function supports. The functions with an asterisk (*) are the functions for host or COM port.

Function \ Module	ND|6011	ND|6012	ND|6013	ND|6014D	ND|6017	ND|6018	ND|6021	ND|6024	ND|6050	ND|6052	ND|6053	ND|6054	ND|6056	ND|6058	ND|6060	ND|6063	ND|6080
ND_ClearCtr																	●
ND_ClearLatchAlarm	●	●															
ND_ConfigLeadingCode	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
ND_DisableAlarm	●	●		●													●
ND_DisableChecksum*																	
ND_DisableCtrAlarm																	●
ND_EnableAlarm	●	●		●													●
ND_EnableChecksum*																	
ND_EnableCtrAlarm																	●
ND_EnableDigitalFilter																	●
ND_EnableLinearMapping				●													
ND_GetAlarmStatus	●	●		●													
ND_GetConfig	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
ND_GetCtrAlarmStatus																	●
ND_GetHighAlarmLimit	●	●		●													
ND_GetLowAlarmLimit	●	●		●													
ND_GetModuleName	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
ND_GetVersion	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
ND_GetWDT	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
ND_HostOK*																	
ND_InitialComm*																	
ND_ReadAI	●	●	●	●													
ND_ReadAllAI			●														
ND_ReadBackAO							●										
ND_ReadBackDO	●	●		●					●						●		●
ND_ReadChanAI					●	●											
ND_ReadCJC	●					●											
ND_ReadCtrAlarmLimit																	●
ND_ReadCtrGateMode																	●
ND_ReadCtrInitialValue																	●
ND_ReadCtrInputMode																	●
ND_ReadCtrMaxValue																	●
ND_ReadCtrOverflowFlag																	●
ND_ReadCtrStatus																	●
ND_ReadCtrValue																	●
ND_ReadDI	●	●		●				●	●	●	●	●		●	●		
ND_ReadEventCount	●	●		●													

Function \ Module	ND|6011	ND|6012	ND|6013	ND|6014D	ND|6017	ND|6018	ND|6021	ND|6024	ND|6050	ND|6052	ND|6053	ND|6054	ND|6056	ND|6058	ND|6060	ND|6063	ND|6080
ND_ReadFilterStatus																	●
ND_ReadMappingSource				●													
ND_ReadMappingTarget				●													
ND_ReadMinWidthAtHigh																	●
ND_ReadMinWidthAtLow																	●
ND_ReadSyncData	●	●	●	●				●	●	●	●	●		●	●		
ND_ReadTTLInputHigh																	●
ND_ReadTTLInputLow																	●
ND_ReleaseComm*																	
ND_ResetEventCount	●	●		●													
ND_ResetStatusAO							●	●									
ND_SendCommand*																	
ND_SetConfig	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
ND_SetCtrAlarmLimit																	●
ND_SetCtrGateMode																	●
ND_SetCtrInitialValue																	●
ND_SetCtrInputMode																	●
ND_SetCtrMaxValue																	●
ND_SetHighAlarmLimit	●	●		●													
ND_SetLowAlarmLimit	●	●		●													
ND_SetMinWidthAtHigh																	●
ND_SetMinWidthAtLow																	●
ND_SetMUXChans			●		●	●											
ND_SetTTLInputHigh																	●
ND_SetTTLInputLow																	●
ND_SetWDT	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
ND_StartCtr																	●
ND_Sync*																	
ND_TimeOut*																	
ND_WriteAO							●										
ND_WriteChanAO								●									
ND_WriteDOLine									●				●	●	●	●	
ND_WriteDOPort	●	●		●					●				●	●	●	●	●
ND_WriteMappingSource				●													
ND_WriteMappingTarget				●													