I-87H17W
API Library Function

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.

Copyright

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.

Table of Contents

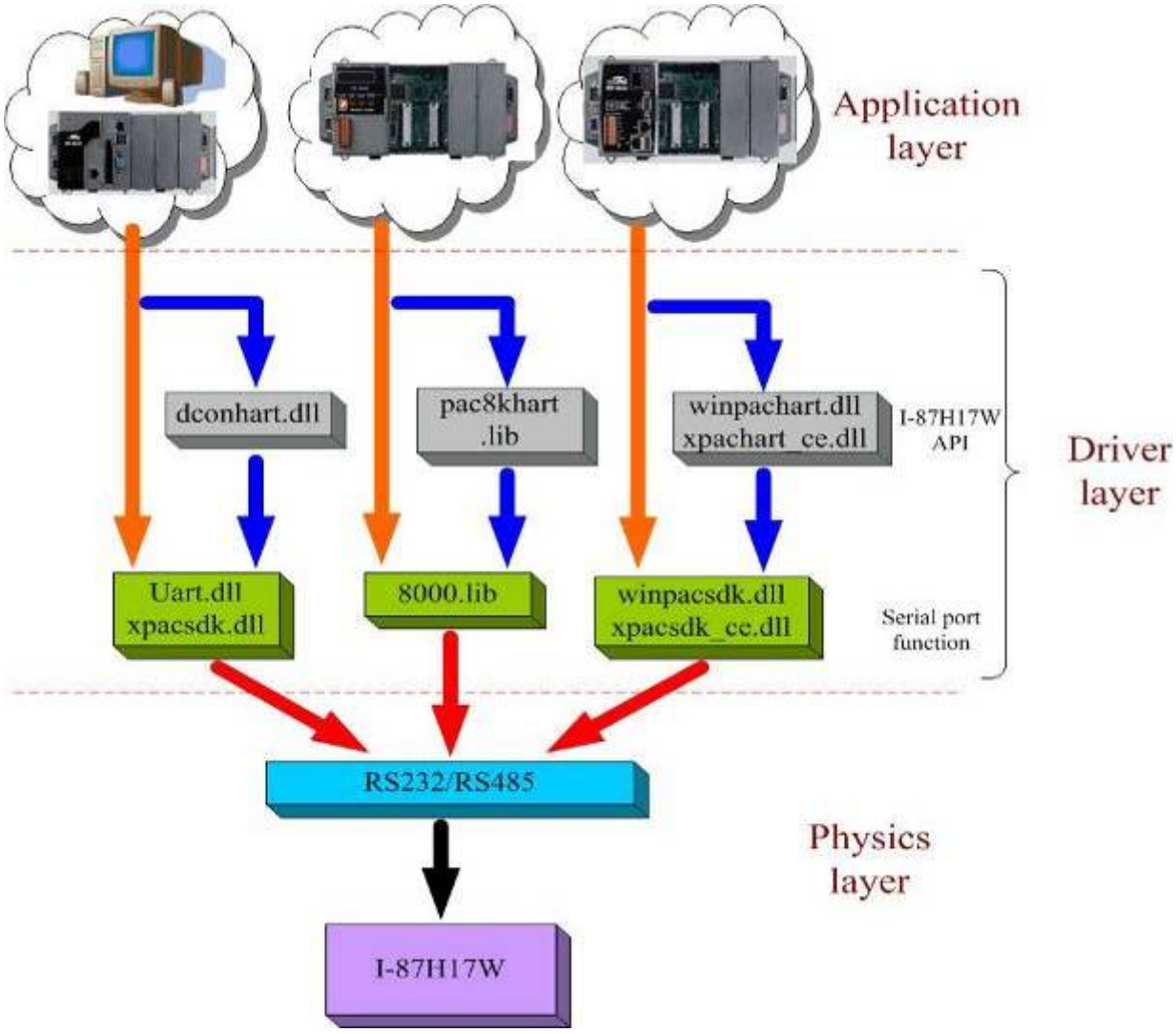
1. Introduction

The API library of I-87H17W provides various OS version for MiniOS7 / Windows 2000 / XP / XPe / Windows 7 and WinCE. Therefore, the users' programs developed by the API library can be ported easily from one platform to another one without any modification.

1.1 API Library Overview with various OS platform

OS	Platform	Library	Development Environment
MiniOS7 (Not Yet)	I-8000 iPAC-8000	pac8khart.lib 8000l.lib	BC 3.01 TC++ 1.2.1
Windows	PC	dconhart.dll dconhart_net.dll Uart.dll	VC++ 6.0 .Net 2005
WinCE	WinPAC (CE5)	winpachart.dll winpachart_net.dll winpacsdk.dll	EVC 4.0 .Net 2005
	XPAC (CE6)	xpachart_ce.dll xpachart_ce_net.dll xpacsdk_ce.dll	EVC 4.0 .Net 2005
XPe	XPAC	xpachart.dll xpachart_net.dll xpacsdk.dll	VC++ 6.0 .Net 2005

1.2 API Library Architecture



2. API Library Application

2.1 For I-8000 / iPAC-8000 (Not Yet)

[TC / BC development Tool]

- TC 2.01
- TC++ 1.01
- BC++ 3.1

From Borland website (http://community.borland.com/museum), the users can download the free TC v2.01 and TC++ v1.01 compilers.

[I-8000 / iPAC-8000]

- 8000e.lib, 8000e.h → i-8000 functions
  8000a.lib, 8000a.h → iPAC-8000 functions
- pac8khart.lib, dcon2hart.h → I-87H17W functions

2.2 For PC / XPAC (XPe)

[PC]

- Uart.dll, Uart.h → PC UART functions
- dconhart.dll, dconhart.lib, dcon2hart.h → I-87H17W functions (VC6.0)
- dconhart_net.dll → I-87H17W functions (.Net 2005)

[XPAC (XPe)]

- xpacsdk.dll, xpacsdk.lib, xpacsdk.h → XPAC functions
- xpachart.dll, xpachart.lib, dcon2hart.h → I-87H17W functions (VC6.0)
- xpachart_net.dll → I-87H17W functions (.Net 2005)

2.3 For XPAC (CE6)

[XPAC (CE6)]

- xpacsdk_ce.dll, xpacsdk_ce.lib, xpacsdk_ce.h → XPAC functions
- xpachart_ce.dll, xpachart_ce.lib, dcon2hart.h → I-87H17W functions (EVC)
- xpachart_ce_net.dll → I-87H17W functions (.Net 2005)

2.4 For WinPAC / ViewPAC (CE5)

[EVC Development Tool]

From Microsoft website (http://msdn.microsoft.com/downloads/Default.aspx), user

can download the free EVC++ 4.0.

How to create the new project of wince?

Step 1: Installing Embedded Visual C++ 4.0

Please refer to Microsoft website and look up related information.

Step 2: Installing EVC++4.0 Service Pack 4(SP4)

Please refer to Microsoft website and look up related information.

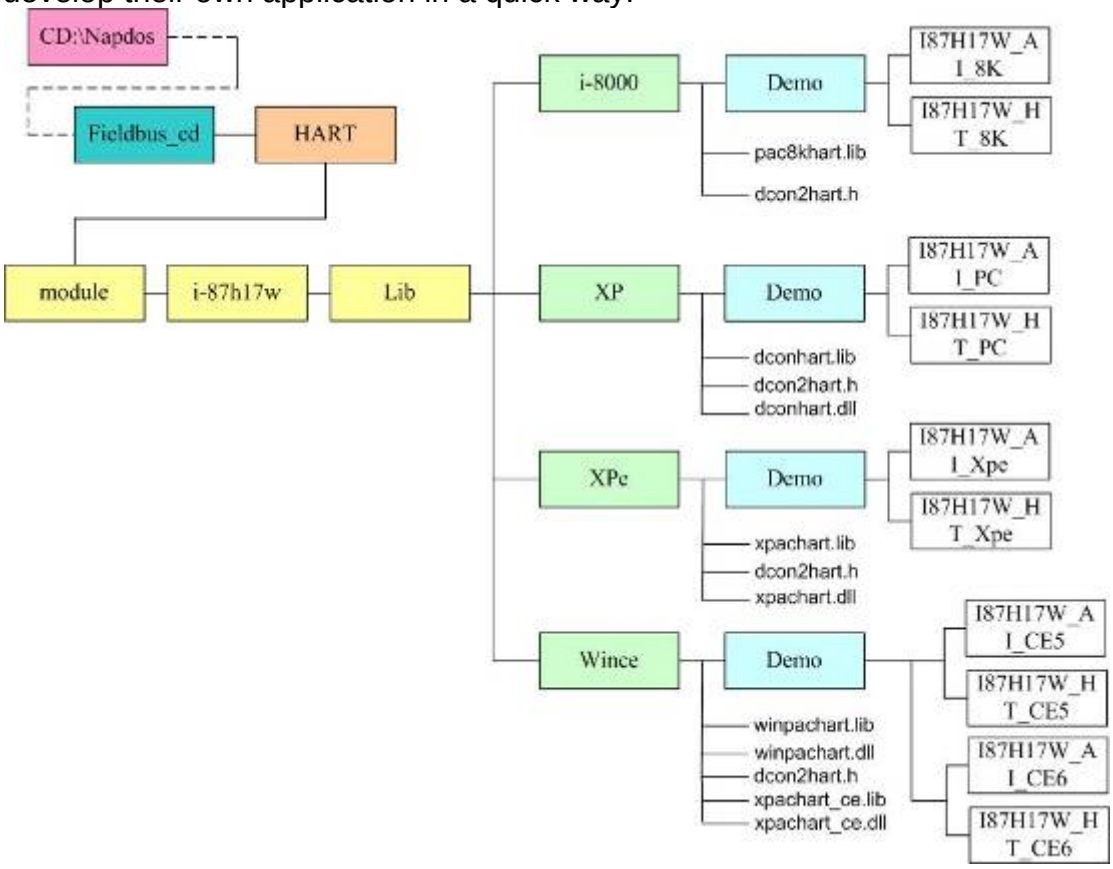
Step 3: Downloading winpacsdk to your PC. (Download from http://ftp.icpdas.com/pub/cd/winpac/napdos/wp-8x4x_ce50/sdk/winpacsdk/)

[WinPAC / VeiwPAC]

- winpacsdk.dll, winpacsdk.lib, winpacsdk.h → WinPA C/ ViewPAC functions
- winpachart.dll, winpachart.lib, dcon2hart.h → I-87H17W functions
- winpachart_net.dll → I-87H17W functions (.Net 2005)

2.5 API Demo List

Based on the demo programs, User can easily understand how to use the function and develop their own application in a quick way.



3. API Library Function

3.1 API Function List

Function	Description	Section
hart_GetModuleName	Get Module Name	3.2.1
hart_GetFwVersion	Get Module Firmware Version	3.2.2
hart_GetModRstSta	Get Module Reset Status	3.2.3
hart_GetModInitSta	Get Module Init Status	3.2.4
hart_GetModDBSta	Get Module Daughter Board Status	3.2.5
hart_Send	Send HART frame to HART device	3.2.6
hart_Recv	Read HART frame from HART device	3.2.7
hart_GetWDTStatus	Get Host Watchdog Status	3.2.8
hart_RstWDTStatus	Reset Host Watchdog Status	3.2.9
hart_GetWDTConfig	Get Host Watchdog Timeout Settings	3.2.10
hart_SetWDTConfig	Set Host Watchdog Timeout Settings	3.2.11
hart_SetZeroCal	Run Zero calibration of the specified ch.	3.2.12
hart_SetFullCal	Run Full calibration of the specified ch.	3.2.13
hart_ClrAllChHLVal	Clear max. or min. AI data of all channels	3.2.14
hart_ClrChHLVal	Clear max. or min. AI data of the specified ch.	3.2.15
hart_ReadChHLfVal	Get max. or min. float AI data of the specified ch.	3.2.16
hart_ReadChHLhVal	Get max. or min. hex AI data of the specified ch.	3.2.17
hart_ReadAllChHLfVal	Get max. or min. float AI data of all channels	3.2.18
hart_ReadAllChHLhVal	Get max. or min. hex AI data of all channels	3.2.19
hart_ReadChfVal	Get current float AI data of the specified ch.	3.2.20
hart_ReadChhVal	Get current hex AI data of the specified ch.	3.2.21
hart_ReadAllfVal	Get current float AI data of all channels	3.2.22
hart_ReadAllhVal	Get current hex AI data of all channels	3.2.23
hart_GetLibVer	Get API Library Version	3.2.24

3.2 API Function Description

3.2.1. hart_GetModuleName

Description:

Get the HART I/O module name.

Syntax:

```
short hart_GetModuleName (
    BYTE bComPort,
    short iAddr,
    char* InBuf,
    short iCheckSum,
    WORD wTimeout
);
```

Parameter:

bComPort **:**

[in] COM Port Number (0 to 255)

iAddr **:**

[in] Module Address (0 to 255)

InBuf **:**

[out] Return Module Name

iCheckSum **:**

[in] 0**:** Disable / 1**:** Enable

wTimeout **:**

[in] Timeout Value (Unit**:** ms; normal=100)

Return Value:

Return 0 means success, others means failure.

3.2.2. hart_GetFwVersion

Description:

Get the firmware version of HART I/O module.

Syntax:

```
short hart_GetFwVersion (
    BYTE bComPort,
    short iAddr,
    char* InBuf,
```

```
    short iCheckSum,
    WORD wTimeout
);
```

Parameter:

bComPort **:**

[in] COM Port Number (0 to 255)

iAddr **:**

[in] Module Address (0 to 255)

InBuf **:**

[out] Return Module Firmware Version (Like: A1.5)

iCheckSum **:**

[in] 0**:** Disable / 1**:** Enable

wTimeout **:**

[in] Timeout Value (Unit**:** ms)

Return Value:

Return 0 means success, others means failure.

3.2.3. hart_GetModRstSta

Description:

Get the reset status of HART I/O module.

Syntax:

```
short hart_GetFwVersion (
    BYTE bComPort,
    short iAddr,
    BYTE* status,
    short iCheckSum,
    WORD wTimeout
);
```

Parameter:

bComPort **:**

[in] COM Port Number (0 to 255)

iAddr **:**

[in] Module Address (0 to 255)

status **:**

[out] Return Module Reset Status.

1 = module has been reset and the value will be 0 after reading.

0 = the module has not been reset.

iCheckSum **:**
[in] 0**:** Disable / 1**:** Enable
wTimeout **:**
[in] Timeout Value (Unit**:** ms)

Return Value:

Return 0 means success, others means failure.

3.2.4. hart_GetModInitSta

Description:

Get the init status of HART I/O module.

Syntax:

```
short hart_GetFwVersion (
    BYTE bComPort,
    short iAddr,
    BYTE* status,
    short iCheckSum,
    WORD wTimeout
);
```

Parameter:

bComPort **:**
[in] COM Port Number (0 to 255)
iAddr **:**
[in] Module Address (0 to 255)
status **:**
[out] Return Module Init Status.
0**:** The INIT jumper is in the “INIT” position.
1**:** The INIT jumper is in the “Normal” position.
iCheckSum **:**
[in] 0**:** Disable / 1**:** Enable
wTimeout **:**
[in] Timeout Value (Unit**:** ms)

Return Value:

Return 0 means success, others means failure.

3.2.5. hart_GetModDBSta

Description:

Get the daughter board connection status of HART I/O module.

Syntax:

```
short hart_GetFwVersion (
    BYTE bComPort,
    short iAddr,
    BYTE* status,
    short iCheckSum,
    WORD wTimeout
);
```

Parameter:

bComPort **:**

[in] COM Port Number (0 to 255)

iAddr **:**

[in] Module Address (0 to 255)

status **:**

[out] Return Daughter Board Connection Status.

0**:** daughter board is connected.

1**:** daughter board is disconnected.

iCheckSum **:**

[in] 0**:** Disable / 1**:** Enable

wTimeout **:**

[in] Timeout Value (Unit**:** ms)

Return Value:

Return 0 means success, others means failure.

3.2.6. hart_Send

Description:

Send the HART frame to HART device.

Syntax:

```
short hart_Send (
    BYTE bComPort,
    short iAddr,
    HTSendFrame* HTFrame,
```

```
    BYTE ch,
    short iCheckSum,
    WORD wTimeout
);
```

Parameter:

bComPort **:**

[in] COM Port Number (0 to 255)

iAddr **:**

[in] Module Address (0 to 255)

HTFrame **:**

[in] HART Frame Data Consists of the “HTSendFrame” structure.

```
typedef struct {
    BYTE preamble;
    BYTE delimeter;
    BYTE addr[5];
    BYTE cmd;
    BYTE data[255];
    WORD SdataLen;
} HTSendFrame;
```

Preamble **:** Preamble Frame (0xFF => only 5 ~ 20)

Delimiter **:** Delimiter Frame (HEX)

addr[5] **:** HART Address
(1) The short frame only uses addr[0].
(2) The long frame uses addr[0] ~ addr[4].

Cmd **:** HART Command No. (Universal, Common-Practice and Transmitter-Specific)

data[255] **:** HART Data to Send

SdataLen **:** HART Data Length.

ch **:**

[in] The Channel of HART I/O Module (0~7 for I-87H17W)

iCheckSum **:**

[in] 0**:** Disable / 1**:** Enable

wTimeout **:**

[in] Timeout Value (Unit**:** ms)

Return Value:

Return 0 means success, others means failure.

3.2.7. hart_Recv

Description:

Receive the HART frame from HART device.

Syntax:

```
short hart_Recv (
    BYTE bComPort,
    short iAddr,
    HTRcvFrame* HTFrame,
    BYTE ch,
    short iCheckSum,
    WORD wTimeout
);
```

Parameter:

bComPort **:**

[in] COM Port Number (0 to 255)

iAddr **:**

[in] Module Address (0 to 255)

HTFrame **:**

[out] Return HART Frame Data Consists of the “HTRcvFrame” structure.

```
typedef struct {
    BYTE preamble;
    BYTE delimeter;
    BYTE addr[5];
    BYTE cmd;
    BYTE data[255];
    WORD RdataLen;
    WORD response_code;
} HTRcvFrame;
```

Preamble **:** Preamble Frame (0xFF => only 5 ~ 20)

Delimiter **:** Delimiter Frame (HEX)

addr[5] **:** HART Address
(1) The short frame only uses addr[0].
(2) The long frame uses addr[0] ~ addr[4].

Cmd **:** HART Command No. (Universal, Common-Practice and Transmitter-Specific)

data[255] **:** HART Data from HART device

RdataLen **:** HART Data Length.

response_code**:** Response Code

ch **:**

[in] The Channel of HART I/O Module (0~7 for I-87H17W)

iCheckSum **:**

[in] 0**:** Disable / 1**:** Enable

wTimeout **:**

[in] Timeout Value (Unit**:** ms)

Return Value:

Return 0 means success, others means failure.

3.2.8. hart_GetWDTStatus

Description:

Get the host WDT timeout status of HART I/O module.

Syntax:

```
short hart_GetWDTStatus (
    BYTE bComPort,
    short iAddr,
    WORD* status,
    short iCheckSum,
    WORD wTimeout
);
```

Parameter:

bComPort **:**

[in] COM Port Number (0 to 255)

iAddr **:**

[in] Module Address (0 to 255)

status **:**

[out] Return Host WDT Timeout Status of Module.

[Bit 7]

0**:** indicates that the host watchdog is disabled

1**:** indicates the host watchdog is enabled

[Bit 2]

0**:** indicates that no host watchdog timeout occurred

1**:** indicates that a host watchdog timeout occurred

iCheckSum **:**

[in] 0**:** Disable / 1**:** Enable

wTimeout **:**

[in] Timeout Value (Unit**:** ms)

Return Value:

Return 0 means success, others means failure.

3.2.9. hart_RstWDTStatus

Description:

Reset the host WDT timeout status of HART I/O module.

Syntax:

```
short hart_RstWDTStatus (
    BYTE bComPort,
    short iAddr,
    short iCheckSum,
    WORD wTimeout
);
```

Parameter:

bComPort **:**
    [in] COM Port Number (0 to 255)
iAddr **:**
    [in] Module Address (0 to 255)
iCheckSum **:**
    [in] 0**:** Disable / 1**:** Enable
wTimeout **:**
    [in] Timeout Value (Unit**:** ms)

Return Value:

Return 0 means success, others means failure.

3.2.10. hart_GetWDTConfig

Description:

Get the host WDT timeout setting of HART I/O module.

Syntax:

```
short hart_GetWDTConfig (
    BYTE bComPort,
    short iAddr,
    BYTE* Enwdt,
    BYTE* times,
    short iCheckSum,
    WORD wTimeout
);
```

Parameter:

bComPort **:**
    [in] COM Port Number (0 to 255)
iAddr **:**
    [in] Module Address (0 to 255)

Enwdt **:**
[out] Return Host WDT Function Enabled / Disabled Status of Module.
times **:**
[out] Return Host WDT Timeout Value of Module. (Unit**:** 0.1 sec)
iCheckSum **:**
[in] 0**:** Disable / 1**:** Enable
wTimeout **:**
[in] Timeout Value (Unit**:** ms)

Return Value:

Return 0 means success, others means failure.

3.2.11. hart_SetWDTConfig

Description:

Set the host WDT timeout setting of HART I/O module.

Syntax:

```
short hart_SetWDTConfig (
    BYTE bComPort,
    short iAddr,
    BYTE Enwdt,
    BYTE times,
    short iCheckSum,
    WORD wTimeout
);
```

Parameter:

bComPort **:**
[in] COM Port Number (0 to 255)
iAddr **:**
[in] Module Address (0 to 255)
Enwdt **:**
[in] Set Host WDT Enabled / Disabled Function of Module.
times **:**
[in] Set Host WDT Timeout Value of Module.
iCheckSum **:**
[in] 0**:** Disable / 1**:** Enable
wTimeout **:**
[in] Timeout Value (Unit**:** ms)

Return Value:

Return 0 means success, others means failure.

3.2.12. hart_SetZeroCal

Description:

Run the zero calibration of the specified channel.

Syntax:

```
short hart_SetZeroCal (
    BYTE bComPort,
    short iAddr,
    BYTE ch,
    short iCheckSum,
    WORD wTimeout
);
```

Parameter:

bComPort **:**
    [in] COM Port Number (0 to 255)
iAddr **:**
    [in] Module Address (0 to 255)
ch **:**
    [in] The Channel of HART I/O Module (0~7 for I-87H17W)
iCheckSum **:**
    [in] 0**:** Disable / 1**:** Enable
wTimeout **:**
    [in] Timeout Value (Unit**:** ms)

Return Value:

Return 0 means success, others means failure.

3.2.13. hart_SetFullCal

Description:

Run the full calibration of the specified channel.

Syntax:

```
short hart_SetFullCal (
    BYTE bComPort,
    short iAddr,
    BYTE ch,
```

```
    short iCheckSum,
    WORD wTimeout
);
```

Parameter:

bComPort **:**

[in] COM Port Number (0 to 255)

iAddr **:**

[in] Module Address (0 to 255)

ch **:**

[in] The Channel of HART I/O Module (0~7 for I-87H17W)

iCheckSum **:**

[in] 0**:** Disable / 1**:** Enable

wTimeout **:**

[in] Timeout Value (Unit**:** ms)

Return Value:

Return 0 means success, others means failure.

3.2.14. hart_ClrAllChHLVal

Description:

Clear the maximum / minimum analog input value of all channels.

Syntax:

```
short hart_ClrAllChHLVal (
    BYTE bComPort,
    short iAddr,
    BYTE HLState,
    short iCheckSum,
    WORD wTimeout
);
```

Parameter:

bComPort **:**

[in] COM Port Number (0 to 255)

iAddr **:**

[in] Module Address (0 to 255)

HLState **:**

[in] >0 **:** maximum **;** 0 **:** minimum

iCheckSum **:**

[in] 0**:** Disable / 1**:** Enable

wTimeout **:**

[in] Timeout Value (Unit**:** ms)

Return Value:

Return 0 means success, others means failure.

3.2.15. hart_ClrChHLVal

Description:

Clear the maximum or minimum analog input value of the specified channel.

Syntax:

```
short hart_ClrChHLVal (
    BYTE bComPort,
    short iAddr,
    BYTE HLState,
    BYTE ch,
    short iCheckSum,
    WORD wTimeout
);
```

Parameter:

bComPort **:**

[in] COM Port Number (0 to 255)

iAddr **:**

[in] Module Address (0 to 255)

HLState **:**

[in] >0 **:** maximum **;** 0 **:** minimum

ch **:**

[in] The Channel of HART I/O Module (0~7 for I-87H17W)

iCheckSum **:**

[in] 0**:** Disable / 1**:** Enable

wTimeout **:**

[in] Timeout Value (Unit**:** ms)

Return Value:

Return 0 means success, others means failure.

3.2.16. hart_ReadChHLfVal

Description:

Get the maximum or minimum analog input value with float format of the specified channel.

Syntax:

```
short hart_ReadChHLfVal (
    BYTE bComPort,
    short iAddr,
    BYTE HLState,
    BYTE ch,
    float* fVal,
    short iCheckSum,
    WORD wTimeout
);
```

Parameter:

bComPort **:**

[in] COM Port Number (0 to 255)

iAddr **:**

[in] Module Address (0 to 255)

HLState **:**

[in] 'H' **:** maximum **;** 'L' **:** minimum

ch **:**

[in] The Channel of HART I/O Module (0~7 for I-87H17W)

fVal **:**

[out] Return the maximum or minimum float Al value of the specified channel

iCheckSum **:**

[in] 0**:** Disable / 1**:** Enable

wTimeout **:**

[in] Timeout Value (Unit**:** ms)

Return Value:

Return 0 means success, others means failure.

3.2.17. hart_ReadChHLhVal

Description:

Get the maximum or minimum analog input value with hex format of the specified channel.

Syntax:

```
short hart_ReadChHLhVal (
    BYTE bComPort,
    short iAddr,
    BYTE HLState,
    BYTE ch,
    WORD* hVal,
    short iCheckSum,
    WORD wTimeout
);
```

Parameter:

bComPort **:**

[in] COM Port Number (0 to 255)

iAddr **:**

[in] Module Address (0 to 255)

HLState **:**

[in] 'H' **:** maximum **;** 'L' **:** minimum

ch **:**

[in] The Channel of HART I/O Module (0~7 for I-87H17W)

hVal **:**

[out] Return the maximum or minimum hex AI value of the specified channel

iCheckSum **:**

[in] 0**:** Disable / 1**:** Enable

wTimeout **:**

[in] Timeout Value (Unit**:** ms)

Return Value:

Return 0 means success, others means failure.

3.2.18. hart_ReadAllChHLfVal

Description:

Get the maximum or minimum analog input value with float format of all channels.

Syntax:

```
short hart_ReadAllChHLfVal (
    BYTE bComPort,
    short iAddr,
    BYTE HLState,
    float fVal[ ],
    short iCheckSum,
    WORD wTimeout
```

);

Parameter:

bComPort **:**

[in] COM Port Number (0 to 255)

iAddr **:**

[in] Module Address (0 to 255)

HLState **:**

[in] 'H' **:** maximum **;** 'L' **:** minimum

fVal[ ] **:**

[out] Return the maximum or minimum float AI value of all channels

iCheckSum **:**

[in] 0**:** Disable / 1**:** Enable

wTimeout **:**

[in] Timeout Value (Unit**:** ms)

Return Value:

Return 0 means success, others means failure.

3.2.19. hart_ReadAllChHLhVal

Description:

Get the maximum or minimum analog input value with hex format of all channels.

Syntax:

```
short hart_ReadAllChHLhVal (
    BYTE bComPort,
    short iAddr,
    BYTE HLState,
    WORD hVal[ ],
    short iCheckSum,
    WORD wTimeout
);
```

Parameter:

bComPort **:**

[in] COM Port Number (0 to 255)

iAddr **:**

[in] Module Address (0 to 255)

HLState **:**

[in] 'H' **:** maximum **;** 'L' **:** minimum

hVal[ ] **:**

[out] Return the maximum or minimum hex AI value of all channels

iCheckSum **:**
[in] 0**:** Disable / 1**:** Enable
wTimeout **:**
[in] Timeout Value (Unit**:** ms)

Return Value:

Return 0 means success, others means failure.

3.2.20. hart_ReadChfVal

Description:

Get the current analog input value with float format of the specified channel.

Syntax:

```
short hart_ReadChfVal (
    BYTE bComPort,
    short iAddr,
    BYTE ch,
    float* fVal,
    short iCheckSum,
    WORD wTimeout
);
```

Parameter:

bComPort **:**
[in] COM Port Number (0 to 255)
iAddr **:**
[in] Module Address (0 to 255)
ch **:**
[in] The Channel of HART I/O Module (0~7 for I-87H17W)
fVal **:**
[out] Return the current float Al value of the specified channel
iCheckSum **:**
[in] 0**:** Disable / 1**:** Enable
wTimeout **:**
[in] Timeout Value (Unit**:** ms)

Return Value:

Return 0 means success, others means failure.

3.2.21. hart_ReadChhVal

Description:

Get the current analog input value with hex format of the specified channel.

Syntax:

```
short hart_ReadChhVal (
    BYTE bComPort,
    short iAddr,
    BYTE ch,
    WORD* hVal,
    short iCheckSum,
    WORD wTimeout
);
```

Parameter:

bComPort **:**

[in] COM Port Number (0 to 255)

iAddr **:**

[in] Module Address (0 to 255)

ch **:**

[in] The Channel of HART I/O Module (0~7 for I-87H17W)

hVal **:**

[out] Return the current hex AI value of the specified channel

iCheckSum **:**

[in] 0**:** Disable / 1**:** Enable

wTimeout **:**

[in] Timeout Value (Unit**:** ms)

Return Value:

Return 0 means success, others means failure.

3.2.22. hart_ReadAllfVal

Description:

Get the current analog input value with float format of all channels.

Syntax:

```
short hart_ReadAllfVal (
    BYTE bComPort,
    short iAddr,
    float fVal[ ],
```

```
    short iCheckSum,
    WORD wTimeout
);
```

Parameter:

bComPort **:**

[in] COM Port Number (0 to 255)

iAddr **:**

[in] Module Address (0 to 255)

fVal[ ] **:**

[out] Return the current float AI value of all channels

iCheckSum **:**

[in] 0**:** Disable / 1**:** Enable

wTimeout **:**

[in] Timeout Value (Unit**:** ms)

Return Value:

Return 0 means success, others means failure.

3.2.23. hart_ReadAllhVal

Description:

Get the current analog input value with hex format of all channels.

Syntax:

```
short hart_ReadAllhVal (
    BYTE bComPort,
    short iAddr,
    WORD hVal[ ],
    short iCheckSum,
    WORD wTimeout
);
```

Parameter:

bComPort **:**

[in] COM Port Number (0 to 255)

iAddr **:**

[in] Module Address (0 to 255)

hVal[ ] **:**

[out] Return the current hex AI value of all channels

iCheckSum **:**

[in] 0**:** Disable / 1**:** Enable

wTimeout **:**

[in] Timeout Value (Unit**:** ms)

Return Value:

Return 0 means success, others means failure.

3.2.24. hart_GetLibVer

Description:

Get API library version.

Syntax:

```
short hart_GetLibVer (void);
```

Parameter:

None

Return Value:

Return the API library version value with hex format.

3.3 Return Code

The return value is used to show the result of executing HART I/O library functions. The following is the all return codes.

Error Code	Error ID
0	**_NOERR**
-99	**impossible_er**
-98	**para_mismatch_er**
-8	**response_er**
-7	**check_sum_er**
-6	**rece_cmd_er**
-5	**addr_burst_er**
-4	**addr_master_er**
-3	**delimiter_er**
-2	**rdata_tooshort_er**
-1	**timeout_er**
7	**_CheckSumError**
12	**_ResultStrCheckError**
13	**_CmdError**
19	**_UnderInputRange**
20	**_ExceedInputRange**
50	**_FORMATERROR**
51	**RECEIVEING**
52	**_FORMATNOTMATCH**
8	**_ComPortNotOpen**
15	**_TimeOut**
25	**ComInUse_Err**
65533	**COMInUseOrInvalid_Err**
0x14001	**_PAC_ERR_UART_CHECKSUM**
0x14002	**_PAC_ERR_UART_READ_TIMEOUT**
0x14003	**_PAC_ERR_UART_RESPONSE**
0x14004	**_PAC_ERR_UART_UNDER_INPUT_RANGE**
0x14005	**_PAC_ERR_UART_EXCEED_INPUT_RANGE**
0x14006	**_PAC_ERR_UART_OPEN_FAILED**
0x14007	**_PAC_ERR_UART_GET_COMM_STATUS_ERROR**
0x14008	**_PAC_ERR_UART_INVALID_VALUE**

4. History Version

Ver.	Author	Date	Description
1.0	Bill	2011/08/26	1. First version
2.0	Edward	2012/10/19	1. API Library v2.0 and add the below API. (1) hart_GetModRstSta( ) (2) hart_GetModInitSta( ) (3) hart_GetModDBSta( )