I-87H17W

HART Analog Input Module

User's Manual

Warranty

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

Warning

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

Trademark

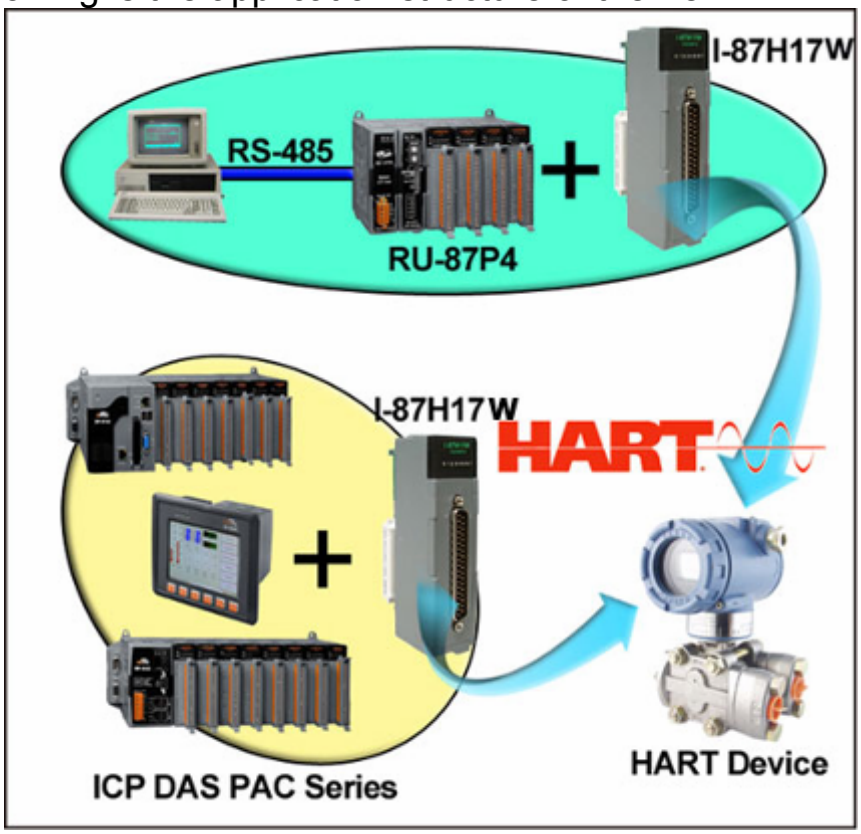
The names used for identification only may be registered trademarks of their respective companies.

Table of Contents

1. Introduction

The I-87H17W is a HART analog input module with 8 channels. It can measures 4~20mA current and acts as a HART master to access HART slave devices simultaneously. Its user-friendly design saves a lot of work when measuring current by using a built-in resistor without any external resistor. The communication baud rate of I-87H17W is adjustable and supports DCON protocol and DCON utility (Download from http://ftp.icpdas.com/pub/cd/8000cd/napdos/driver/dcon_utility/). Otherwise, ICP DAS also provides API library for users to develop their own program by VC / VB / C# / VB.net environment.

The following is the application structure of the I-87H17W module.



1.1 Features

- Support 4 ~ 20 mA current input capability with or without HART
- Support just point-to-point mode (analog / digital) for every HART channel

- Support 2 or 4 wire transmitters HART device
- With a built-in resistor for current measurement easily
- Open wire detection for every channel
- 4 kV ESD protection
- 2500 VDC intra-module isolation
- Built-in Dual Watchdog
- RoHS compliance
- Support DCON protocol and DCON utility
- Provide API library for VC / VB / C# and VB.net
- Provide PWR and Channel Indication LED
- Support in WinPAC / ViewPAC / XPAC / iPAC series PACs
- Support I-87Kn / RU-87Pn Remote I/O Expansion Unit

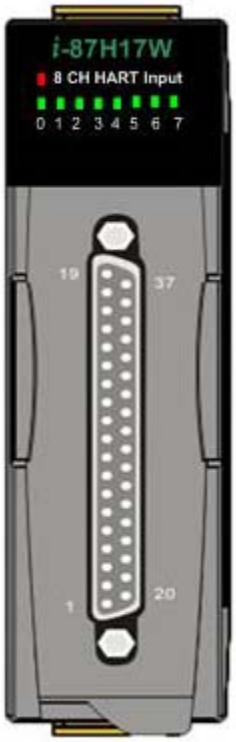
1.2 Specifications

Analog Input	
Input Channels	8 Differential 4-20 mA input channels
HART Wiring	2 or 4 wire transmitters
HART Response Time	1s per single channel
Resolution	16-bit
Zero Drift	+/- 20μV/ °C
Span Drift	+/- 25μV / °C
Common Mode Rejection	86 dB
Normal Mode Rejection	100 dB
Input Impedance	400 Ohms
Common Voltage	-200V to +200V
Open Wire Detection	Yes
4KV ESD Protection	Yes, Contact for each terminal.
Intra-module Isolation (Field to Logic)	2500V_{DC}
Watchdog	
Dual Watchdog	Yes, Module(1.6 sec) and Comm.(programmable)

LED Display	
1 LED as Power Indicator 8 LED as HART Communication Indicator	
Power	
Power Consumption	Maximum **:** 1.8W
Environment	
Operating Temperature	-25 to 75 °C
Storage Temperature	-30 to 75 °C
Humidity	5 to 95% RH, non-condensing
Dimensions	
30mm x 102mm x 115mm(W x L x H) Detail	

Note: A warm up period of 30 minutes is recommended in order to achieve the complete performance results described in the specifications.

2.Hardware



2.1 LED Indication

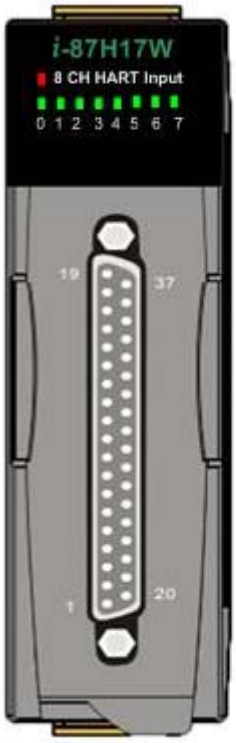
(1) **PWR LED :**

Provide one red LED as Power indicator. When the module is powered on, the PWR LED will be on. Otherwise, it will be off.

(2) **HART Channel LED :**

Provide eight green LEDs as HART channel communication indicators. When the specific HART channel is communicating with HART device, the corresponding HART channel LED will be on. Otherwise it will be off. Note if users just read the AI value of channel without any HART communication, then the channel LED will not turn on.

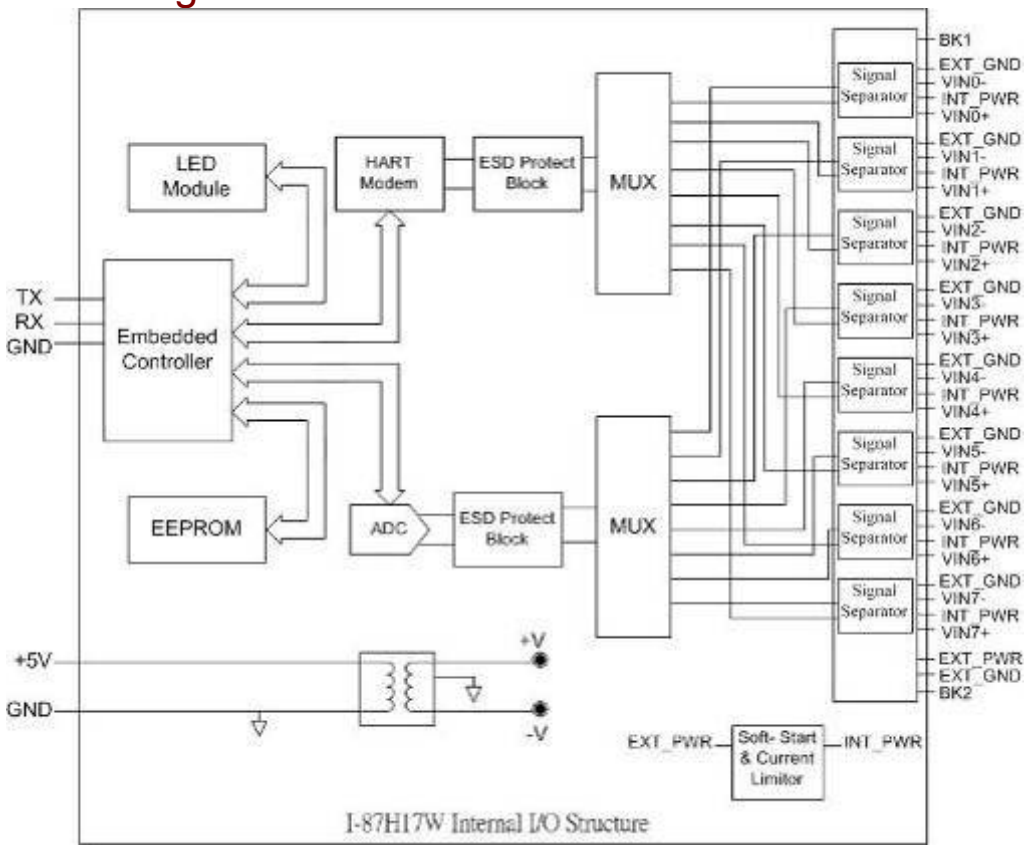
2.2 Terminal Assignment



Pin Assignment Name	Terminal No.		Pin Assignment Name
X	19	37	BK2
EXT_PWR	18	36	EXT_GND
VIN7-	17	35	INT_PWR7
VIN7+	16	34	EXT_GND
VIN6-	15	33	INT_PWR6
VIN6+	14	32	EXT_GND
VIN5-	13	31	INT_PWR5
VIN5+	12	30	EXT_GND
VIN4-	11	29	INT_PWR4
VIN4+	10	28	EXT_GND
VIN3-	09	27	INT_PWR3
VIN3+	08	26	EXT_GND
VIN2-	07	25	INT_PWR2
VIN2+	06	24	EXT_GND
VIN1-	05	23	INT_PWR1
VIN1+	04	22	EXT_GND
VIN0-	03	21	INT_PWR0
VIN0+	02	20	EXT_GND
BK1	01		

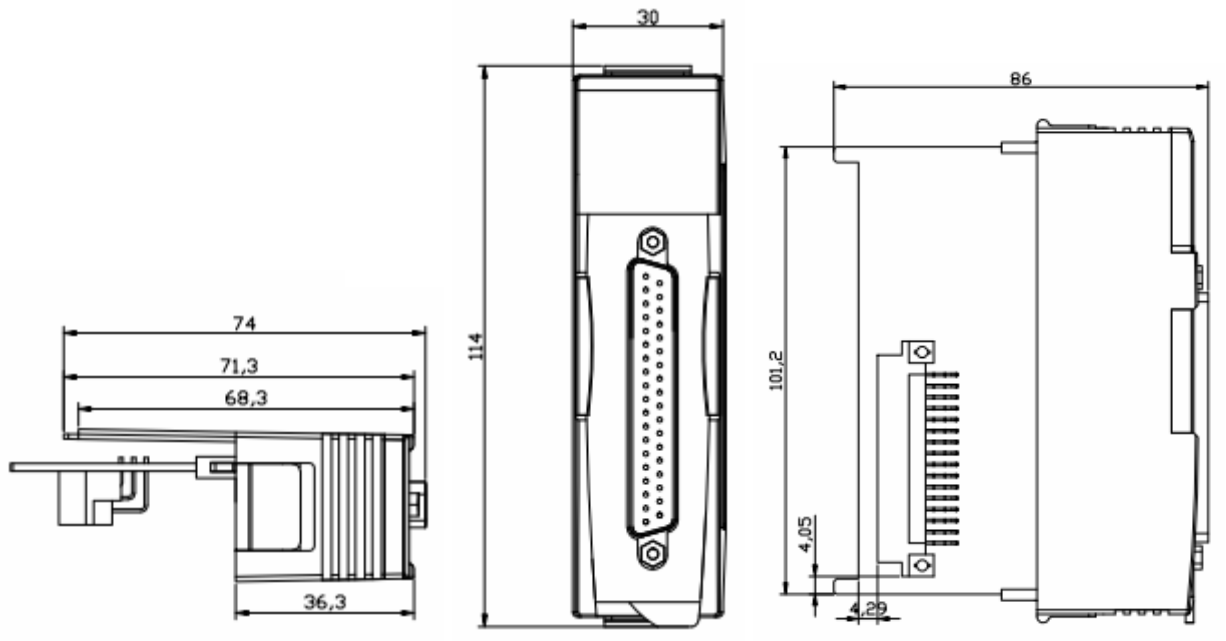
37-pin male D-Sub Connector

2.3 Block Diagrams



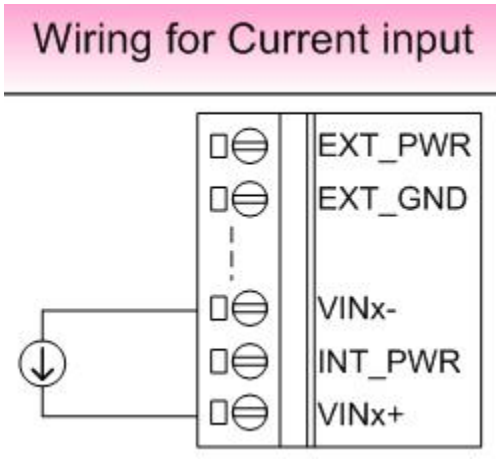
I-87H17W Internal I/O Structure

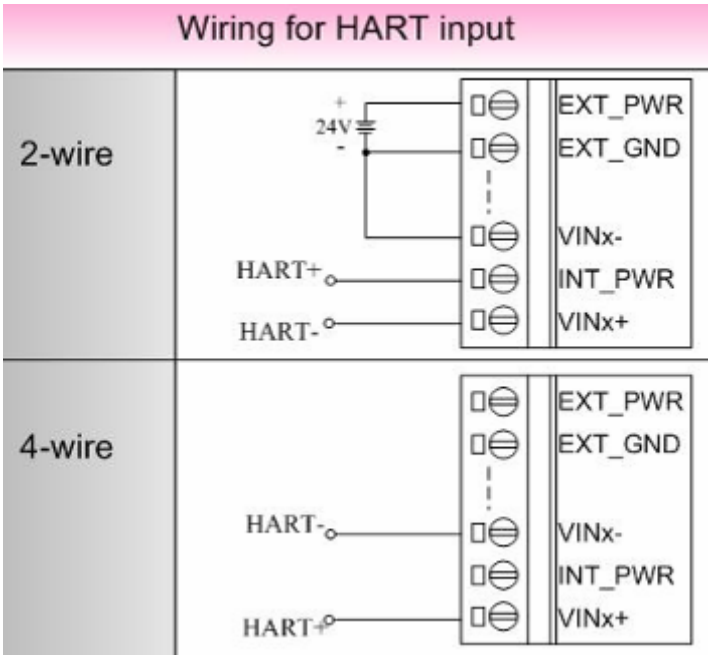
2.4 Dimensions



High Profile (unit : mm)

2.5 Wiring Diagrams





2.6 Default Settings

The factory default setting of the I-87H17W module is as below.

Module Param.	Setting
Module address	0x01
Analog input type	4mA to 20mA
Baudrate	115200, N, 8, 1
Checksum	Disabled
Unit format	Engineering

2.7 Configuration Tables

(1) Analog Input Type Setting (TT)

Type Code: 07 (4~20 mA Input)

(2) Baudrate Setting (CC)

Code	03	04	05	06	07	08	09	0A
Baud Rate	1200	2400	4800	9600	19200	38400	57600	115200

Note: The communication data format is fixed to be 8, N, 1 (eight data bits, no parity and one stop bit)

(3) Data Format Setting (FF)

7	6	5	4	3	2	1	0
FS	CS	MS	Reserved			DF	

Key	Description
DF (Data format)	00: Engineering unit 01: % of FSR (full scale range) 10: 2's complement hexadecimal
MS	none
CS (Checksum)	0: Disabled 1: Enabled
FS	none

Note : The reserved bits should be zero.

(4) The "+/- Full Scale" table

Data Format	+F.S.	-F.S.
Engineering Unit	+20.000	+04.000
% of FSR	+100.00	+000.00

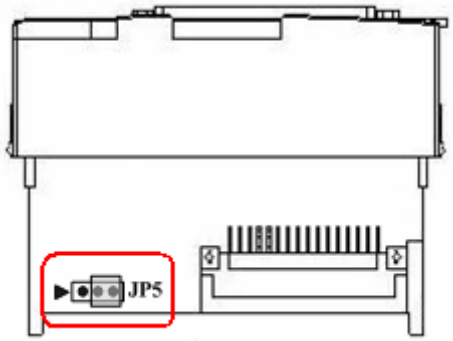
2's Complement HEX	7FFF	0000

(5) The "Over/Under Range" table.

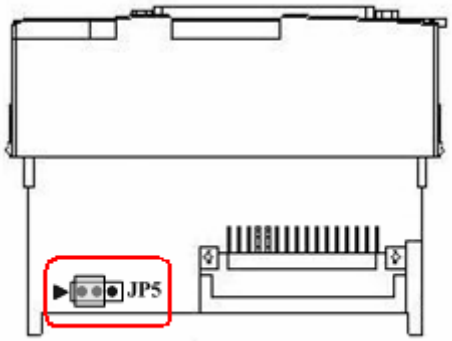
Data Format	Over Range	Under Range
Engineering Unit	+9999.9	-9999.9
% of FSR	+999.99	-999.99
2's Complement HEX	7FFF	8000

2.8 Init / Normal Mode

There is a jumper – JP5 (supported in hardware v1.3 or newer) in the I-87H17 module shown as the below figure.



"**Init**" Mode



"**Normal**" Mode

(1) "**Init**" Mode **:**
Connect the pin 2&3 of JP5 together.
(2) "**Normal**" Mode **:**
Connect the pin 1&2 of JP5 together.

2.8.1 **Init Mode**

In "Init" mode, the module settings are in the default value and it is useful when users forget the module settings. The module address is "**0**" and the baud rate parameters is "**115200, N, 8, 1**" without checksum. After the module settings are configured successfully, it will take effect when I-87H17W runs in the "Normal" mode.

Module Parameters	Default Setting
Module address	0x00
Analog input type	4mA to 20mA
Baud rate	115200, N, 8, 1

Checksum	Disabled
Data format	Engineering

2.8.2 **Normal Mode**

In “Normal” mode, users can just configure **address** and **data format** parameters of module and operate module normally.

2.9 Quick Start

To install the module, please follow the steps below **:**

1. Insert the I-87H17W to the I-87Kn or RU-87Pn.
2. Power on the I-87Kn or RU-87Pn and user can read analog input via the DCON utility.
3. If users need to test HART communication, please refer to the “I87H17W_HT_PC” demo. (Download from ftp://ftp.icpdas.com/pub/cd/fieldbus_cd/hart/module/i-87h17w/lib/xp/demo/i87h17w_ht_pc/release/)

2.10 Module Calibration

The module calibration procedure is as below.

1. Warm up the module for 30 minutes.
2. Enable calibration. Refer to Section 2.13 for details.
3. Apply 0mA to the zero calibration.
4. Send the zero calibration command. Refer to Sections 2.4 for details.
5. Apply 25mA to the span calibration.
6. Send the span calibration command. Refer to Sections 2.5 details.
7. Repeat steps 3 to 7 three times.

Warning: *It is not recommended that calibration be performed until the process is fully understood.*

2.11 Technical Support

When users encounter problems while using the I-87H17W module, and unable to find the help you need in this manual or on our website, please contact ICP DAS Product Support.

Email : support@icpdas.com
Website: http://www.icpdas.com/service/support.htm

When requesting technical support, be prepared to provide the following information about your system:

1. Module name and serial number. The serial number can be found printed on the barcode label attached to the cover of the module.
2. Firmware version. See Section 2.8 for information regarding the command used to identify the firmware version.
3. Host configuration (type and operating system)
4. If the problem is reproducible, please give the full details describing the procedure used to reproduce the problem.
5. Specific error messages displayed. If a dialog box with an error message is displayed, please include the full text of the dialog box, including the text in the title bar.
6. If the problem involves other programs or hardware devices, please describe the details of the problem in full.
7. Any comments and suggestions related to the problem are welcome.

ICP DAS will reply to your request by email within three business days.

3. Module DCON Protocol

Each I-87H17W module has a unique ID number that is used for addressing purposes and stored in non-volatile memory. The ID is 01 by default and can be changed by user command. All commands to the modules contain the ID address, meaning that only the addressed module will respond. The only exception to this is commands #** (Section 3.2) and ~** (Section 3.18), which are sent to all modules, but in these cases, the modules do not reply to the command.

SendCmd Format:

Leading Character	Module Address	Command	[CHKSUM]	CR

Response Format:

Leading Character	Module Address	Data	[CHKSUM]	CR

Note **:**

(1) [CHKSUM] :

It is a 2-character checksum that is present when the checksum setting is enabled (See Section 2.7 and 3.1 for details). The following is the checksum Calculation description.

[1] Calculate the ASCII code sum of all the characters in the send /response command string except the carriage return character (CR).

[2] The checksum is equal to the sum masked by 0xFF.

Example**:**

Send Command string**:** $012(CR)

1. Sum of the string = “$”+”0”+”1”+”2” = 24h+30h+31h+32h = B7h
2. Therefore the checksum is B7h, and so CHKSUM = “B7”
3. The command string with the checksum = $012B7(CR)

Response string**:** !01200600(CR)

1. Sum of the string = “!”+”0”+”1”+”2”+”0”+”0”+”6”+”0”+”0” = 21h+30h+31h+32h+30h+30h+36h+30h+30h = 1AAh
2. Therefore the checksum is AAh, and so CHKSUM = “AA”
3. The response string with the checksum = !01200600AA(CR)

Note: All characters should be in upper case.

(2) CR :

End of command character, carriage return (0x0D)

3.1 Module Command Sets

3.1.1 General Command Sets

General Command Sets			
Command	**Response**	**Description**	**Section**
%AANNTTCCFF	!AA	Set module configuration	3.2.1
$AA2	!AATTCCFF	Get module configuration	3.2.2
#AA	>(data)	Get AI data of all channels	3.2.3
#AAN	>(data)	Get AI data of the specified channel	3.2.4
$AA0Ci	!AA	Run span calibration of the specified channel	3.2.5
$AA1Ci	!AA	Run zero calibration of the specified channel	3.2.6
$AA5	!AAS	Get reset status	3.2.7
$AAI	!AAS	Get INIT status	3.2.8
$AAF	!AA(data)	Get Firmware Version	3.2.9
$AAM	!AA(data)	Get module name	3.2.10
@AACS	!AA	Clear max. / min. AI data	3.2.11
@AACSN	!AA	Clear max. / min. AI data of the specified channel	3.2.12
@AAOD	!AAN	Get the connection status of daughter board	3.2.13
@AARS	!(data)	Get max. / min. AI data	3.2.14
@AARSN	!(data)	Get max. / min. AI data of the specified channel	3.2.15
~AAEV	!AA	Enable / Disable the calibration	3.2.16

3.1.2 HART Command Sets

HART Command Sets			
Command	**Response**	**Description**	**Section**
$AAHTNppdd(aa)cc	!AAN	Set HART Frame Format	3.3.1

#AAHTSN(data)	>AAN	Send HART Data	3.3.2
#AAHTRN	>AA(len)(data)	Receive HART Data	3.3.3

3.1.3 Host Watchdog Command Sets

Host Watchdog Command Sets			
Command	**Response**	**Description**	**Section**
~**	No Response	Host is OK	3.4.1
~AA0	!AASS	Get Host WDT Timeout Status of Module	3.4.2
~AA1	!AA	Reset Host WDT Timeout Status of Module	3.4.3
~AA2	!AAEVV	Get Host WDT Timeout Setting	3.4.4
~AA3EVV	!AA	Set Host WDT Timeout Setting	3.4.5
~AARD	!AATT	Get Response Delay Time.	3.4.6
~AARDTT	!AA	Set Response Delay Time.	3.4.7

3.1.4 General Error Condition

1. No response error condition **:**
   (1) The command syntax is incorrect
   (2) The communication is failed
   (3) No module with the specified address

3.2 General Command Sets

3.2.1 %AANNTTCCFF (Set Configuration)

(1) Description:

Set the module configuration.

(2) Syntax:

%AANNTTCCFF[CHKSUM](CR)

% **:** Delimiter character

AA **:** Original module address in hex format (00 to FF)

NN **:** New module address in hex format (00 to FF)

TT : Module type (07 => 4~20mA Input)
CC : New Baudrate code (Section 2.7 for details). To change the Baud rate, the module must be in the INIT mode.
FF : Used to set the data format, checksum, and filter settings (Section 2.7 for details). To change the checksum setting, the module must be in the INIT mode.

(3) Response:

Valid Response : !AA[CHKSUM](CR)
Invalid Response : ?AA[CHKSUM](CR)
! : Delimiter character for a valid response
? : Delimiter character for an invalid response. If changing the **Baud Rate** or **checksum** settings without in the INIT mode, the module will return an invalid command.
AA : Module address in hex format (00 to FF)

(4) Examples:

[1] Change module (01) address to 02.
Send: %0102070600 Resp.: !02
[2] Set module (02) data format to be 02 (2's complement).
Send: %0202070602 Resp.: !02
[3] Change module (01) baudrate to 115200bps. The module returns an invalid command, because module is not in INIT mode.
Send: %0101070A00 Resp.: ?01
[4] Change module (01) baudrate to 115200bps.
Send: %0101070A00 Resp.: !01

Notes:

1. When the **address**, **data format**, **type code** and **filter** settings are modified in "Normal" mode successfully, they will take effect immediately.
2. The **baud rate**, **checksum** and **parity** settings can just be modified in "INIT" mode. When they are modified successfully, please run the module in "Normal" mode to take effect the setting.
3. Users can use the "DCON Utility" to modify these setting easily and quickly. (ftp://ftp.icpdas.com/pub/cd/8000cd/napdos/driver/dcon_utility/)

3.2.2 $AA2 (Get Configuration)

(1) Description:

Get the module configuration.

(2) Syntax:

$AA2[CHKSUM](CR)

$: Delimiter character
AA : Module address in hex format (00 to FF)
2 : Command to get the module configuration

(3) Response:

Valid Response : !AATTCCFF[CHKSUM](CR)
Invalid Response : ?AA[CHKSUM](CR)
! : Delimiter character for a valid response
? : Delimiter character for an invalid response.
AA : Module address in hex format (00 to FF)
TT : Module type (07 => 4~20mA Input)
CC : Baud Rate code of the module (Section 2.7 for details).
FF : Data format, checksum and filter settings of the module (Section 2.7 for details).

(4) Examples:

[1] Get module (01) configuration.
Send: $012 Resp.: !01070A00

Notes:

1. Users can use the “DCON Utility” to modify these setting easily and quickly. (ftp://ftp.icpdas.com/pub/cd/8000cd/napdos/driver/dcon_utility/)

3.2.3 #AA (hart_ReadAllfVal / hart_ReadAllhVal)

(1) Description:

Get the AI data of all channels.

(2) Syntax:

#AA[CHKSUM](CR)

: Delimiter character

AA : Module address in hex format (00 to FF)

(3) Response:
Valid Response : **>(Data)**[CHKSUM](CR)
Invalid Response : ?AA[CHKSUM](CR)
> : Delimiter character for a valid response
? : Delimiter character for an invalid response.
AA : Module address in hex format (00 to FF)
(Data) : AI Data of all channels. (Section 2.7 for details)

(4) Examples:
[1] Get all channel AI data (engineering format) of module (01).
Send: #01
Resp.: >+025.12+020.45+012.78+018.97+003.24+015.35+008.07
+014.79
[2] Get all channel AI data (hex format) of module (02).
Send: #02
Resp.: >4C532628E2D683A20F2ADBA16284BA71
[3] Get module (03) and the data is under range.
Send: #03
Resp.: >-9999.9-9999.9-9999.9-9999.9-9999.9-9999.9-9999.9
-9999.9

3.2.4 #AAN (hart_ReadChfVal / hart_ReadChhVal)

(1) Description:
Get the AI data of the specified channel.

(2) Syntax:
#AAN[CHKSUM](CR)
: Delimiter character
AA : Module address in hex format (00 to FF)
N : Specify the channel to be read

(3) Response:
Valid Response : **>(Data)**[CHKSUM](CR)
Invalid Response : ?AA[CHKSUM](CR)

> : Delimiter character for a valid response
? : Delimiter character for an invalid response.
AA : Module address in hex format (00 to FF)
(Data) : AI data of the specified channel. (Section 2.7 for details)

(4) Examples:
[1] Get AI data from channel 2 of module (03).
Send: #032 Resp.: >+025.13
[2] Get AI data from channel 9 of module (02) and an error is returned because channel 9 is invalid.
Send: ##029 Resp.: ?02

3.2.5 $AA0Ci (Run Span Calibration of Ch.)

(1) Description:
Run the span calibration of the specified channel.

(2) Syntax:
$AA0Ci[CHKSUM](CR)
$: Delimiter character
AA : Module address in hex format (00 to FF)
0 : Command for the span calibration
Ci : Specify the channel to be calibrated

(3) Response:
Valid Response : !AA[CHKSUM](CR)
Invalid Response : ?AA[CHKSUM](CR)
! : Delimiter character for a valid response
? : Delimiter character for an invalid response. An invalid command is returned if the specified channel is incorrect.
AA : Module address in hex format (00 to FF)

(4) Examples:
[1] Run the span calibration on channel 0 of module (01).
Send: $010C0 Resp.: !01
[2] Run the span calibration on channel 1 of module (03). An invalid command is returned because the calibration function is disabled.

Send: $030C1 Resp.: ?03

3.2.6 $AA1Ci (Run Zero Calibration of Ch.)

(1) Description:

Run the zero calibration of the specified channel.

(2) Syntax:

$AA1Ci[CHKSUM](CR)

$: Delimiter character
AA : Module address in hex format (00 to FF)
1 : Command for the zero calibration
Ci : Specify the channel to be calibrated

(3) Response:

Valid Response : !AA[CHKSUM](CR)
Invalid Response : ?AA[CHKSUM](CR)
! : Delimiter character for a valid response
? : Delimiter character for an invalid response. An invalid command is returned if the specified channel is incorrect.
AA : Module address in hex format (00 to FF)

(4) Examples:

[1] Run the zero calibration on channel 0 of module (01).
Send: $011C0 Resp.: !01
[2] Run the zero calibration on channel 1 of module (03). An invalid command is returned because the calibration function is disabled.
Send: $031C1 Resp.: ?03

3.2.7 $AA5 (hart_GetModRstSta)

(1) Description:

Get module reset status.

(2) Syntax:

$AA5[CHKSUM](CR)

$: Delimiter character
AA : Module address in hex format (00 to FF)
5 : Command to get module reset status

(3) Response:
Valid Response : !AAS[CHKSUM](CR)
Invalid Response : ?AA[CHKSUM](CR)
! : Delimiter character for a valid response
? : Delimiter character for an invalid response.
AA : Module address in hex format (00 to FF)
S : Reset status
1 = module has been reset and the value will be 0 after reading.
0 = the module has not been reset.

(4) Examples:
[1] Get the reset status of module (01) and the module has been reset before reading.
Send: $015 Resp.: !011
[2] Get the reset status of module (01) and the module has not been reset before reading.
Send: $015 Resp.: !010

3.2.8 $AAI (hart_GetModInitSta)

(1) Description:
Get module INIT status.

(2) Syntax:
$AAI[CHKSUM](CR)
$: Delimiter character
AA : Module address in hex format (00 to FF)
I : Command to get the module INIT status

(3) Response:
Valid Response : !AAS[CHKSUM](CR)
Invalid Response : ?AA[CHKSUM](CR)
! : Delimiter character for a valid response
? : Delimiter character for an invalid response.

AA : Module address in hex format (00 to FF)
S : INIT jumper status of module
0: The INIT jumper is in the “INIT” position
1: The INIT jumper is in the “Normal” position

(4) Examples:
[1] Get the INIT status of module (01) and module is in the “INIT” mode.
Send: $01I Resp.: !010

3.2.9 $AAF (hart_GetFwVersion)

(1) Description:
Get module firmware version.

(2) Syntax:
$AAF[CHKSUM](CR)
$: Delimiter character
AA : Module address in hex format (00 to FF)
F : Command to get the module firmware version

(3) Response:
Valid Response : !AA(Data)[CHKSUM](CR)
Invalid Response : ?AA[CHKSUM](CR)
! : Delimiter character for a valid response
? : Delimiter character for an invalid response.
AA : Module address in hex format (00 to FF)
(Data) : A string indicates the module firmware version.

(4) Examples:
[1] Get the firmware version of module (01) and the version is A1.5.
Send: $01F Resp.: !01A1.5

3.2.10 $AAM (hart_GetModuleName)

(1) Description:
Get module name.

(2) Syntax:
$AAM[CHKSUM](CR)
$: Delimiter character
AA : Module address in hex format (00 to FF)
M : Command to get the module name

(3) Response:
Valid Response : !AA(Data)[CHKSUM](CR)
Invalid Response : ?AA[CHKSUM](CR)
! : Delimiter character for a valid response
? : Delimiter character for an invalid response.
AA : Module address in hex format (00 to FF)
(Data) : A string indicates the module name.

(4) Examples:
[1] Get the firmware version of module (01) and the module is I-87H17W.
Send: $01M Resp.: !0187H17

3.2.11 @AACS (hart_ClrAllChHLVal)

(1) Description:
Clear the maximum / minimum AI data of all channels.

(2) Syntax:
@AACS[CHK](cr)
@ : Delimiter character
AA : Module address in hex format (00 to FF)
C : Command to clear maximum or minimum AI data
S : S = H for clear maximum analog inputs
S = L for clear minimum analog inputs

(3) Response:
Valid Response : !AA[CHKSUM](CR)
Invalid Response : ?AA[CHKSUM](CR)
! : Delimiter character for a valid response
? : Delimiter character for an invalid response.

AA : Module address in hex format (00 to FF)

(4) Examples:

[1] Clear all the maximum AI data of module (01).
Send: @01CH Resp.: !01

[2] Clear all the minimum AI data of module (02).
Send: @02CL Resp.: !02

3.2.12 @AACSN (hart_ClrChHLVal)

(1) Description:

Clear maximum / minimum AI data of the specified channel.

(2) Syntax:

@AACSN[CHK](cr)

@ : Delimiter character
AA : Module address in hex format (00 to FF)
C : Command to clear maximum or minimum AI data
S : S = H for clear the maximum AI data
S = L for clear the minimum AI data
N : Specify the channel to be cleared

(3) Response:

Valid Response : !AA[CHKSUM](CR)
Invalid Response : ?AA[CHKSUM](CR)
! : Delimiter character for a valid response
? : Delimiter character for an invalid response.
AA : Module address in hex format (00 to FF)

(4) Examples:

[1] Clear the maximum AI data on channel 0 of module (01).
Send: @01CH0 Resp.: !01

[2] Clear the minimum AI data on channel 1 of module (02).
Send: @02CL1 Resp.: !02

3.2.13 @AAOD (hart_GetModDBSta)

(1) Description:

Get the connection status of daughter board.

(2) Syntax:

@AAOD[CHK](cr)

@ : Delimiter character
AA : Module address in hex format (00 to FF)
OD : Command to get open-wired detection status

(3) Response:

Valid Response : !AAN[CHKSUM](CR)
Invalid Response : ?AA[CHKSUM](CR)
! : Delimiter character for a valid response
? : Delimiter character for an invalid response.
AA : Module address in hex format (00 to FF)
N : 0: daughter board is connected
1: daughter board is disconnected

(4) Examples:

[1] Get the open wire status of module (01) and shows that it is open wire status.

Send: @01OD Resp.: !011

3.2.14 @AARS (hart_ReadAllChHLfVal / hart_ReadAllChHLhVal)

(1) Description:

Get the maximum/minimum AI data of all channels.

(2) Syntax:

@AARS[CHK](cr)

@ : Delimiter character
AA : Module address in hex format (00 to FF)
R : Command to get the maximum or minimum AI data
S : S = H for get the maximum AI data
S = L for get the minimum AI data

(3) Response:

Valid Response : **!(Data)**[CHKSUM](CR)
Invalid Response : ?AA[CHKSUM](CR)
! : Delimiter character for a valid response
? : Delimiter character for an invalid response.
AA : Module address in hex format (00 to FF)
(Data): The maximum or minimum AI data of all channels

(4) Examples:

[1] Get all the minimum AI data of module (01).
Send: @01RL
Resp.: !+04.000+05.000+06.123+05.134+09.123+05.345+07.145
+08.145

3.2.15 @AARSN (hart_ReadChHLfVal / hart_ReadChHLhVal)

(1) Description:

Get the maximum/minimum AI data of the specified channel.

(2) Syntax:

@AARSN[CHK](cr)
@ : Delimiter character
AA : Module address in hex format (00 to FF)
R : Command to get the maximum or minimum AI data
S : S = H for get the maximum AI data
S = L for get the minimum AI data
N : Specify the channel to read

(3) Response:

Valid Response : **!(Data)**[CHKSUM](CR)
Invalid Response : ?AA[CHKSUM](CR)
! : Delimiter character for a valid response
? : Delimiter character for an invalid response.
AA : Module address in hex format (00 to FF)
(Data): the maximum or minimum AI data of the specified channel

(4) Examples:

[1] Get the minimum AI data on channel 3 of module (01).

Send**:** @01RL3 Resp.**:** !-05.134

3.2.16 ~AAEV (hart_SetZeroCal / hart_SetFullCal)

(1) Description:

Enable / Disable module calibration.

(2) Syntax:

~AAEV[CHK](cr)

~ **:** Delimiter character

AA **:** Module address in hex format (00 to FF)

E **:** Command to enable/disable calibration

V **:** 1: enable calibration

0: disable calibration

(3) Response:

Valid Response **:** !AA(Data)[CHKSUM](CR)

Invalid Response **:** ?AA[CHKSUM](CR)

! **:** Delimiter character for a valid response

? **:** Delimiter character for an invalid response.

AA **:** module address in hex format (00 to FF)

(4) Examples:

[1] Run the span calibration on channel 0 of module (01). It returns an invalid response because the calibration function is disabled.

Send**:** $010C0 Resp.**:** ?01

[2] Enables calibration of module (01).

Send**:** ~01E1 Resp.**:** !01

[3] Run the span calibration on channel 0 of module (01). It returns a valid response because the calibration function is enabled.

Send**:** $010C0 Resp.**:** !01

3.3 HART Command Sets

3.3.1 $AAHTNppdd(aa)cc (hart_Send-1)

(1) Description:

Set HART Frame Format.

(2) Syntax:

$AAHTNppdd(aa)cc[CHKSUM](CR)

$: Delimiter character
AA : Original module address in hex format (00 to FF)
HT : HART frame
N : Specify the HART channel to be set
pp : HART Preamble (0xFF) Number (5 ~ 20)
dd : HART Delimiter data
aa : HART Address data (short or long address)
cc : HART Command data (Universal, Common-Practice and Transmitter-Specific)

(3) Response:

Valid Response : !AAN[CHKSUM](CR)
Invalid Response : ?AA[CHKSUM](CR)
! : Delimiter character for a valid response
? : Delimiter character for an invalid response.
AA : Module address in hex format (00 to FF)
N : HART channel

(4) Examples:

[1] Set HART frame format on channel 1 of module (01).
Send: $01HT105028000 Resp.: !011

3.3.2 #AAHTSN(data) (hart_Send-2)

(1) Description:

Send HART data to HART device.

(2) Syntax:

#AAHTSN(data)[CHKSUM](CR)
: Delimiter character
AA : Original module address in hex format (00 to FF)
HT : HART frame
S : Send HART data
N : Specify the HART channel
data : HART data

(3) Response:
Valid Response : >AAN[CHKSUM](CR)
Invalid Response : ?AA[CHKSUM](CR)
> : Delimiter character for a valid response
? : Delimiter character for an invalid response.
AA : Module address in hex format (00 to FF)
N : HART channel

(4) Examples:
[1] Send the HART data from channel 1 of module (01). The HART data length is zero.
Set HART frame on channel 0 of module (01).
Send: #01HTS1 Resp.: >011

3.3.3 #AAHTRN (hart_Recv)

(1) Description:
Receive HART data from HART device.

(2) Syntax:
#AAHTRN[CHKSUM](CR)
: Delimiter character
AA : Original module address in hex format (00 to FF)
HT : HART frame
R : Receive HART data
N : Specify the HART channel

(3) Response:
Valid Response : >AA(len)(data)[CHKSUM](CR)

Invalid Response : ?AA[CHKSUM](CR)
> : Delimiter character for a valid response
? : Delimiter character for an invalid response.
AA : Module address in hex format (00 to FF)
(len) : Response data length from HART device
(data) : Response data from HART device

(4) Examples:
[1] Receive the HART data from channel 1 of module (01).
Send: #01HTR1
Resp.: >0032FFFFFFFFFF0680000E0000FE16850705020B08020B0A42A7

3.4 Host Watchdog Command Sets

3.4.1 ~** (Host is OK)

(1) Description:
Inform all modules that the host software is OK.

(2) Syntax:
~[CHKSUM](CR)**
~ : Delimiter character
** : Host OK command

(3) Response:
No response.

(4) Examples:
[1] Sends a "Host is OK" command to all modules.
Send: ~** Resp.: No Response

3.4.2 ~AA0 (hart_GetWDTStatus)

(1) Description:
Get the host watchdog time out status of a module.

(2) Syntax:

~AA0[CHKSUM](CR)

~ : Delimiter character

AA : Module address in hex format (00 to FF)

0 : Command to get the host WDT time out status of module

(3) Response:

Valid Response : !AASS[CHKSUM](CR)

Invalid Response : ?AA[CHKSUM](CR)

! : Delimiter character for a valid response

? : Delimiter character for an invalid response.

AA : Module address in hex format (00 to FF)

SS : Two bits represent the host WDT status.

[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

Note : The host watchdog status of module is stored in EEPROM and can be only reset by using the **~AA1** command.

(4) Examples:

[1] Get the host watchdog status of module (01) and returns 00, meaning that the host watchdog is disabled and no host watchdog time out has occurred.

Send: ~010 Resp.: !0100

[1] Get the host watchdog status of module (02) and returns 04, meaning that the host watchdog is disabled and the host watchdog time out has occurred.

Send: ~020 Resp.: !0204

3.4.3 ~AA1 (hart_RstWDTStatus)

(1) Description:

Reset the host watchdog time out status of a module.

(2) Syntax:
~AA1[CHKSUM](CR)
~ : Delimiter character
AA : Module address in hex format (00 to FF)
1 : Command to reset the host WDT time out status of module

(3) Response:
Valid Response : !AA[CHKSUM](CR)
Invalid Response : ?AA[CHKSUM](CR)
! : Delimiter character for a valid response
? : Delimiter character for an invalid response.
AA : Module address in hex format (00 to FF)

(4) Examples:
[1] Reset the host watchdog time out status of module (01).
Send: ~011 Resp.: !01

3.4.4 ~AA2 (hart_GetWDTConfig)

(1) Description:
Get the host watchdog time out setting of a module.

(2) Syntax:
~AA2[CHKSUM](CR)
~ : Delimiter character
AA : Module address in hex format (00 to FF)
2 : Command to get the host WDT time out setting of module

(3) Response:
Valid Response : !AAEVV[CHKSUM](CR)
Invalid Response : ?AA[CHKSUM](CR)
! : Delimiter character for a valid response
? : Delimiter character for an invalid response.
AA : Module address in hex format (00 to FF)
E : 0: the host watchdog is disabled
1: the host watchdog is enabled

VV : The host WDT time out value of module (unit: 0.1sec). For example: 01 means 0.1 seconds and FF means 25.5 seconds

(4) Examples:

[1] Get the host WDT time out value of module (01) and returns 0xFF, meaning that the host WDT is enabled and the host WDT time out value is 25.5 seconds.

Send: ~012 Resp.: !011FF

3.4.5 ~AA3EVV (hart_SetWDTConfig)

(1) Description:

Set the host watchdog time out setting of a module.

(2) Syntax:

~AA3EVV[CHKSUM](CR)

~ : Delimiter character

AA : Module address in hex format (00 to FF)

3 : Command to set the host WDT time out setting of module

E : 0: the host watchdog is disabled
1: the host watchdog is enabled

VV : The host WDT time out value of module (unit: 0.1sec). For example: 01 means 0.1 seconds and FF means 25.5 seconds

(3) Response:

Valid Response : !AA[CHKSUM](CR)

Invalid Response : ?AA[CHKSUM](CR)

! : Delimiter character for a valid response

? : Delimiter character for an invalid response.

AA : Module address in hex format (00 to FF)

(4) Examples:

[1] Enables the host WDT of module (01) and sets the host WDT time out value to be 10.0 seconds.

Send: ~013164 Resp.: !01

3.4.6 ~AARD (Get Response Delay Time)

(1) Description:

Get the response delay time of HART I/O module.

(2) Syntax:

~AARD[CHKSUM](CR)

~ : Delimiter character
AA : Module address in hex format (00 to FF)
RD : Command to get the response delay time

(3) Response:

Valid Response : !AATT[CHKSUM](CR)
Invalid Response : ?AA[CHKSUM](CR)
! : Delimiter character for a valid response
? : Delimiter character for an invalid response.
AA : Module address in hex format (00 to FF)
TT : The value must be equal or less than 0x1E. For example, 0x01 means 1ms and 0x1A denotes 26 ms.

(4) Examples:

[1] Get the response delay time and returns 16. It means that when the module receives the host command and it will send back the response after 16 ms have elapsed.

Send: ~01RD Resp.: !0110

Notes:

1. Users can use the “DCON Utility” to modify the setting easily and quickly. (ftp://ftp.icpdas.com/pub/cd/8000cd/napdos/driver/dcon_utility/)

3.4.7 ~AARDTT (Set Response Delay Time)

(1) Description:

Set the response delay time of HART I/O module.

(2) Syntax:

~AARDTT[CHKSUM](CR)

~ : Delimiter character
AA : Module address in hex format (00 to FF)
RD : Command to set the response delay time
TT : The value must be equal or less than 0x1E. For example, 0x01 means 1ms and 0x1A denotes 26 ms.

(3) Response:

Valid Response : !AA[CHKSUM](CR)
Invalid Response : ?AA[CHKSUM](CR)
! : Delimiter character for a valid response
? : Delimiter character for an invalid response.
AA : Module address in hex format (00 to FF)

(4) Examples:

[1] Set the response time of module (01) to be 16 ms.
Send: ~01RD10 Resp.: !01

Notes:

1. Users can use the “DCON Utility” to modify the setting easily and quickly. (ftp://ftp.icpdas.com/pub/cd/8000cd/napdos/driver/dcon_utility/)

4. Appendix

4.1 Dual Watchdog Operation

The I-87H17W module includes an Dual Watchdog (Module Watchdog + Host Watchdog) function, making the control system more reliable and stable.

(1) Module Watchdog :

The Module Watchdog is a hardware reset circuit that monitors the operating status of the module. While working in harsh or noisy environments, the module may be shut down by external signals. The circuit allows the module to work continuously without disruption.

(2) Host Watchdog :

The Host Watchdog is a software function that monitors the operating status of the host. Its purpose is to prevent problems due to network/communication errors or host malfunctions. When a host watchdog time out occurs, the module will reset all outputs to a safe state in order to prevent any erroneous operations of the controlled target.

5. History Version

Ver.	Author	Date	Description
1.0	Bill	2011/08/25	1. First version
2.0	Edward	2012/10/19	1. Update Content. 2. Add “Init / Normal” Mode in HW v1.3 or newer.