



I-752N Series User Manual

Version: 3.0.0, Aug. 2022



Written by Tim Tsai
Edited by Sunny Chiu

Warranty

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

Warning

ICP DAS assumes no liability for damages consequent to the use of this product.

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

Copyright

Copyright © 2022 by ICP DAS. All rights are reserved.

Contact Us

If you have any questions, please feel free to contact us via email at:

Service@icpdas.com

Contents

1. Introduction	6
2. Hardware Information.....	9
2.1. Specifications.....	9
2.2. Appearance.....	11
2.3. Pin Assignments.....	13
2.4. Wiring Connections.....	19
2.5. Dimensions	23
2.6. Mounting the Hardware.....	24
2.7. LED Indicator	25
2.8. INIT Mode.....	26
3. Getting Started	27
3.1. Connecting to Host PC and Power Supply	27
3.2. Searching Module	29
3.3. Setting Configuration	32
3.4. Communication Testing.....	34
3.5. Command Line Tool.....	34
4. Function Description.....	36
4.1. RS-232 Port Address.....	36
4.2. End Character (CrLfmode)	38
4.3. Modbus ASCII/Modbus RTU Support (CrLfmode).....	42
4.4. RS-232 Device Communication	42
4.5. Bypass Data to COM2	43
4.6. DI/DO	44
4.7. Dual Watchdog	46
5. INIT Mode and Firmware Update	48
5.1. INIT Mode.....	48
5.2. 7188xw	49
5.3. Firmware Update.....	54
5.4. Getting Communication Parameters of a Module with Unknown Settings	58
6. Applications	63
6.1. Connecting I-7522 to One Agilent 34401A	63
6.2. Connecting I-7523 to Two Agilent 34401A.....	65
6.3. Connecting Four Agilent 34401A with Two I-7523.....	67
6.4. Reading Data from Multiple Barcode Scanner.....	69
6.5. Bypassing data from devices to COM2 (host PC)	70

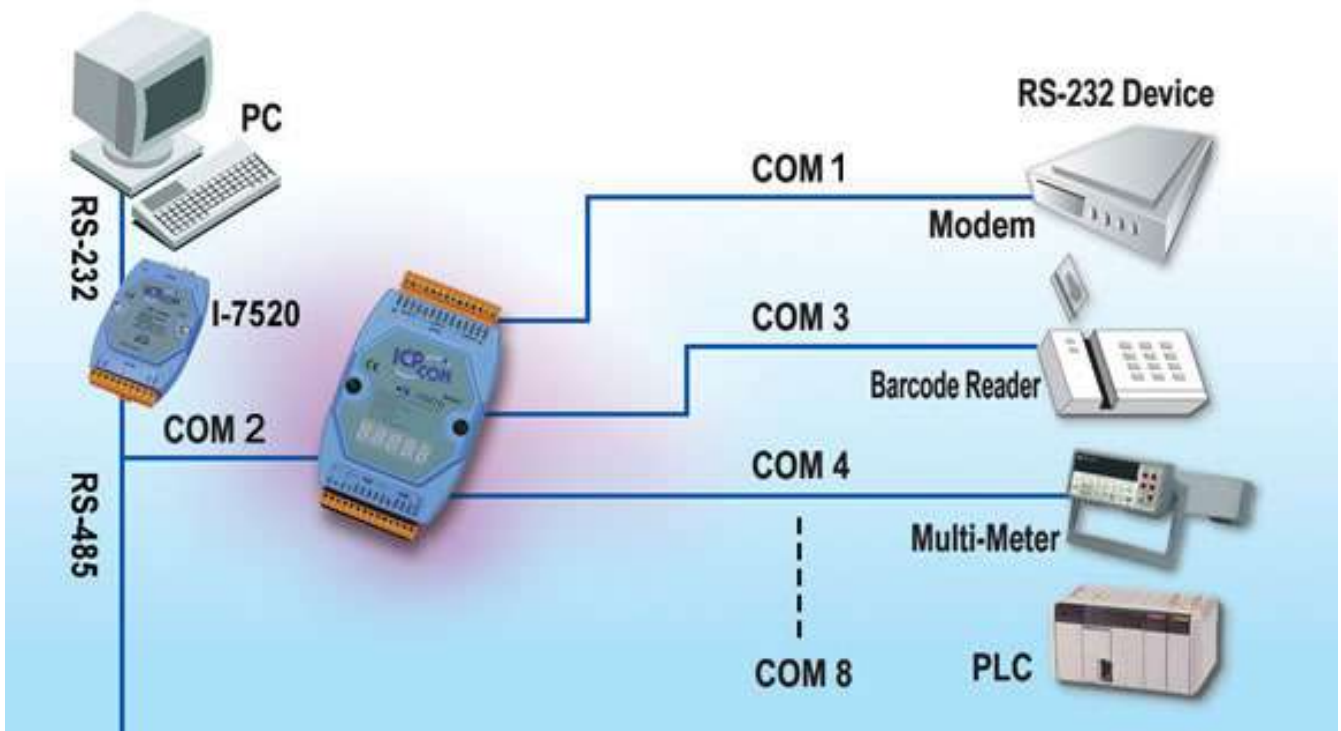
6.6.	Real-time DI Monitoring and DO Control	71
6.7.	Real time A/D Monitoring and D/A Control.....	72
6.8.	8-channel Long Term Event Counters	73
6.9.	Signal Monitor and Alarm Control.....	74
6.10.	Electronic Scale Application	75
7.	Command Sets.....	81
7.1.	Command Format	81
7.2.	COM Port Address.....	81
7.3.	Command Set Table	82
7.4.	Commands.....	87
7.4.1.	\$AAA[addr].....	87
7.4.2.	\$ AABN[baud-rate].....	88
7.4.3.	\$AADN[data-bit]	89
7.4.4.	\$AAPN[parity-bit].....	91
7.4.5.	\$AAON[stop-bit]	93
7.4.6.	\$AA6[ID]	95
7.4.7.	\$AA7	96
7.4.8.	\$AAKV	97
7.4.9.	\$AATN[CrLfmode].....	98
7.4.10.	\$AAW	100
7.4.11.	\$AAXV	101
7.4.12.	\$AA2	102
7.4.13.	\$AAIV	103
7.4.14.	\$AA5	105
7.4.15.	\$AAF	106
7.4.16.	\$AAM	107
7.4.17.	\$AAC[delimiter]	108
7.4.18.	\$AAD.....	109
7.4.19.	[delimiter]AA[bypass].....	110
7.4.20.	\$AAJN[timeout]	111
7.4.21.	\$AAGN[trigger-level].....	114
7.4.22.	\$AAEV	116
7.4.23.	\$AAHV	118
7.4.24.	\$AAU.....	120
7.4.25.	\$AAUR	123
7.4.26.	\$AAUA	124
7.4.27.	\$AAUC	126
7.4.28.	\$AAUN	127
7.4.29.	\$AAUL[minlength].....	128
7.4.30.	\$AAUD[padchar]	129
7.4.31.	\$AAUS[seperator]	130
7.4.32.	\$AAN[buffermode]	132
7.4.33.	\$AAS[lastdatahdl]	133
7.4.34.	DI/DO Data Bit Mapping	134
7.4.35.	\$AAYN	135
7.4.36.	\$AAZNV.....	136
7.4.37.	#**	138

7.4.38.	\$AA4	139
7.4.39.	\$AAL[data]	140
7.4.40.	\$AAR.....	142
7.4.41.	@AA[data].....	143
7.4.42.	#AABBHH.....	145
7.4.43.	#AABCDD	146
7.4.44.	~**	147
7.4.45.	~AA0	148
7.4.46.	~AA1	149
7.4.47.	~AA2	150
7.4.48.	~AA3ETT.....	151
7.4.49.	~AA4P / ~AA4S.....	152
7.4.50.	~AA5P / ~AA5S.....	154
Revision History		156

1. Introduction

With the rapid advancement of communication technology in recent years, automation has changed our daily life at various levels, such as the access control and security system inside intelligent buildings, the parking lot management systems that can be seen everywhere, the self-checkout system of the stores and so on. Using automation to tedious and highly repetitive tasks can boost efficiency for management and service. In addition, importing the data collected by these automation systems into big data analytics can extend the application scope to production or sales information analysis and aggregation.

The I-752N series addressable RS-485 to RS-232 converter provides one RS-485 interface and multiple RS-232/422 ports. It can assign a unique address to every RS-232/422 port for communicating with multiple RS-232 devices in different locations through one RS-485 port. Besides, using the I-752N series module allows the communication distance between host and RS-232 devices beyond the maximum distance of about 15 meters.



Features

■ Addressable RS-232 converter

Most RS-232 devices don't support individual device address. To overcome this limitation, the I-752N series module assigns a unique address to every RS-232/422 port for communicating with multiple RS-232 devices through one RS-485 port. The host can send commands to multiple RS-232 devices and receive responses from these devices by applying the I-752N series module.

■ Connecting multiple RS-232 devices through one RS-232/RS-485/USB port on PC

By using one RS-232/USB to RS-485 converter or one RS-485 port on the PC or PAC, the host program can connect dozens to hundreds of RS-232 devices through I-752N series module. With different baud rate, data bit, parity bit and stop bit settings, the number of connecting devices can be expanded to thousands or more.

■ Support various communication format

1. Baud Rate: 115200, 57600, 38400, 19200, 9600, 4800, 2400, 1200 (bps)
2. Protocol: DCON, Modbus RTU, Modbus ASCII
3. Checksum: Disabled, Enabled
4. Data format: N81, E81, O81, N82, E82, O82, N71, E71, O71, N72, E72, O72
(E82 and O82 are not supported for COM1 and COM2)

■ Built-in queue buffer

Every RS-232 port of the I-752N series module comes with a queue buffer, ranging in size from 12KB to 50KB. Any data from RS-232 devices that are not received during a waiting period will be temporarily stored in the queue buffer to avoid data loss.

■ DI/DO Control

The I-752N series module provides 1/2/5 channels of DI or 1/3/5 channels of DO, which can be used for ON/OFF control and monitoring. For example, DI can be used to connect to sensors or confirm the status of switches, while DO can directly drive relays, LED lights, or turn on/ turn off equipments when an emergency situation occurs.

■ Programmable firmware for being a Master-type converter

The firmware of the I-752N series module can be modified to be Master-type converter. In real industrial applications, many users are not satisfied with Slave-type converters because they cannot be adapted to individual requirement. The powerful I-752N series module can analyze the DI /DO of local RS-232 devices without the need for a Host PC.

I-752N Series Comparison Table

Model	CPU	SRAM	RS-232	RS-422 /RS-485	RS-485	RS-232 /RS-485	DI	DO
I-7521	80188 20 MHz	128KB	-	-	1	1	2	3
I-7522			1	-			2	1
I-7523			2	-			1	-
I-7522A	80188, 40 MHz	512KB	-	1	1	1	5	5
I-7525			3	-			1	1
I-7527			6	-			1	1

What's New in Firmware V4

1. Automatically changes the end character of COM2 according to the end character in received string from host.
2. Added support for Modbus RTU and Modbus ASCII protocols ([section 4.3](#))
3. Automatically converts protocol between Modbus RTU and Modbus ASCII. It means that the I-752N series module can also be used as a gateway between Modbus ASCII and Modbus RTU.
4. Within 30 seconds after powered on, connecting the GND pin (ground) and the INIT* pin for more than 2 seconds enables the I-752N series module to execute the firmware in INIT mode. (Module address = 00, baud rate = 9600, data format = 8N1).

ODM option

The I-752N series module provides ODM options, allowing customers to enhance the protection of the module.

- ✧ COM2: 3000V isolation protection. (available for I-7522A/I-7524/I-7527 only)

2. Hardware Information

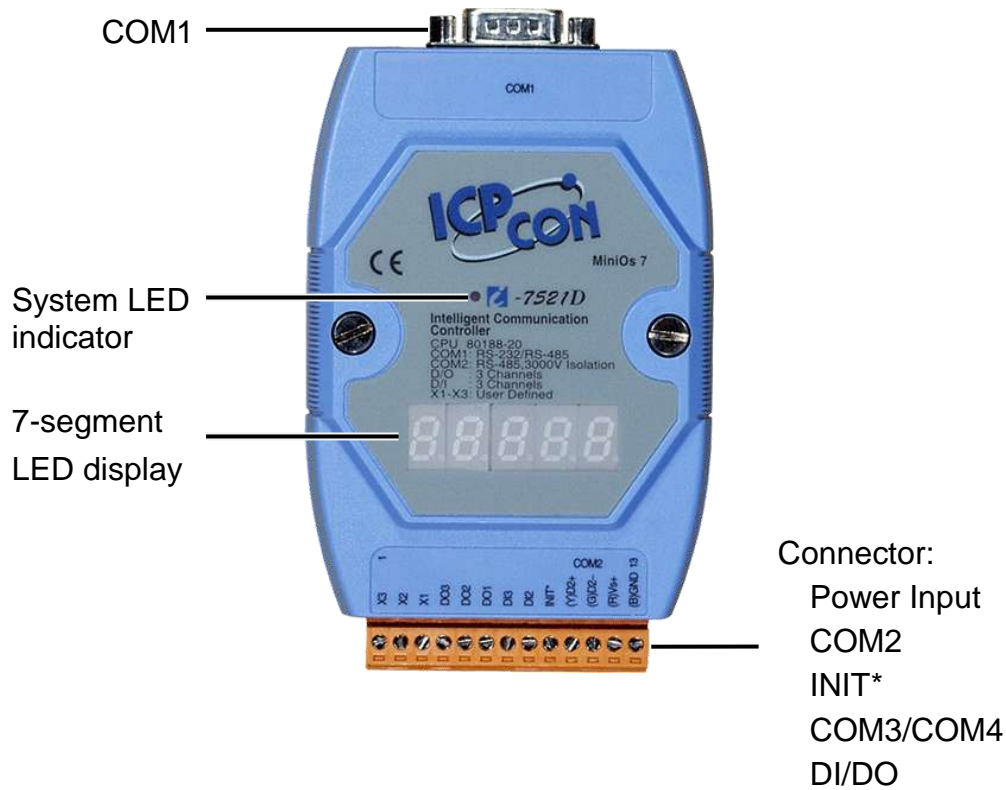
2.1. Specifications

Model	I-7521	I-7521D	I-7522	I-7522D	I-7523	I-7523D
CPU Module						
CPU	80188 or compatible, 20 MHz					
SRAM	128 KB					
Flash	512 KB					
Watchdog Timer	Yes					
Display						
7-segment LED Display	-	Yes	-	Yes	-	Yes
LED Indicators	1 x System					
Communication Port						
Baud Rate	115200 bps Max.					
COM1	5-wire RS-232/RS-485 (DB-9 male)					
COM2	RS-485 (Data+, Data-) with 3000 V _{DC} Isolation					
COM3	-	5-wire RS-232				
COM4	-				3-wire RS-232	
Digital Input						
Channels	2			1		
Type	Dry Contact, Non-isolated					
ON Voltage Level	Close to GND					
OFF Voltage Level	Open					
Digital Output						
Channels	3	1		-		
Type	Open Collector					
Load Voltage	+30 V _{DC}					
Load Current	100 mA					
Power						
Reverse Polarity Protection	Yes					
Input Range	+10 ~ 30 V _{DC}					
Consumption	2.0 W	3.0 W	2.0 W	3.0 W	2.0 W	3.0 W
Mechanical						
Dimension	72 mm x 119 mm x 33 mm (W x H x D)					
Installation	DIN-Rail, Wall mounting					
Environmental						
Operating Temperature	-25 ~ +75 °C					
Storage Temperature	-40 ~ +80 °C					
Humidity	5 ~ 90 % RH, Non-condensing					

Model	I-7522A	I-7522AD	I-7524	I-7524D	I-75237	I-7527D
CPU Module						
CPU	80188 or compatible, 40 MHz					
SRAM	512 KB					
Flash	512 KB					
Watchdog Timer	Yes					
Display						
7-segment LED Display	-	Yes	-	Yes	-	Yes
LED Indicators	1 x System					
Communication Port						
Baud Rate	115200 bps Max.					
COM1	5-wire RS-232/RS-485					
COM2	RS-485 (Data+, Data-)					
COM3	RS-422/RS-485			5-wire RS-232	3-wire RS-232	
COM4						
COM5			-			
COM6						
COM7						
COM8						
Digital Input						
Channels	5		1			
Type	Dry Contact, Non-isolated					
ON Voltage Level	Close to GND					
OFF Voltage Level	Open					
Digital Output						
Channels	5		1			
Type	Open Collector					
Load Voltage	+30 V _{DC}					
Load Current	100 mA					
Power						
Reverse Polarity Protection	Yes					
Input Range	+10 ~ 30 V _{DC}					
Consumption	2.0 W	3.0 W	2.0 W	3.0 W	2.0 W	3.0 W
Mechanical						
Dimension	72 mm x 123 mm x 33 mm (W x H x D)					
Installation	DIN-Rail, Wall mounting					
Environmental						
Operating Temperature	-25 ~ +75 °C					
Storage Temperature	-40 ~ +80 °C					
Humidity	5 ~ 90 % RH, Non-condensing					

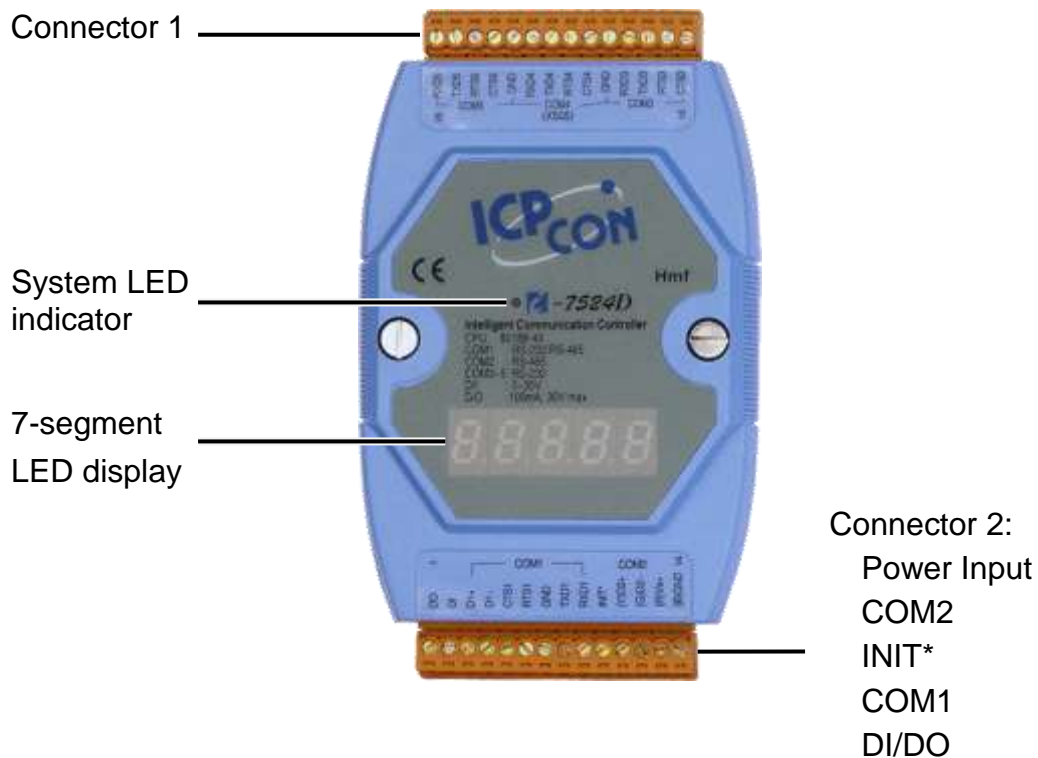
2.2. Appearance

■ I-7521(D)/I-7522(D)/I-7523(D)



Item	Description
COM1	In INIT mode: used for updating firmware or OS image. In Operating mode: used for connecting to an RS-232 device.
System LED indicator	In INIT mode: blinking red. In Operating mode: solid red with firmware V3; blinking red with firmware V4.
7-segment LED display	Displays communication configuration, programmable, available for D versions only.
Connector	Connector for power supply, COM2/3/4 and DI/DO.

■ I-7522A(D)/I-7524(D)/I-7527(D)



Item	Description
Connector1	Connector for COM3 ~ 8 and DI/DO.
System LED indicator	In INIT mode: blinking red. In Operating mode: solid red with firmware V3; blinking red with firmware V4.
7-segment LED display	Displays communication configuration, programmable, available for D versions only.
Connector	Connector for power supply, COM1/2 and DI/DO.

2.3. Pin Assignments

■ I-7521(D)



Terminal No.	Pin Assignment	
COM1	1	DATA+
	2	TXD
	3	RXD
	4	N/C
	5	GND
	6	N/C
	7	CTS
	8	RTS
	9	DATA-

Terminal No.	Pin Assignment	
1	X3	
2	X2	
3	X1	
4	DO3	
5	DO2	
6	DO1	
7	DI3	
8	DI2	
9	INIT*	
COM2	10	D2+
	11	D2-
12	Vs+	
13	GND	

■ I-7522(D)



Terminal No.	Pin Assignment
COM1	1 DATA+
	2 TXD
	3 RXD
	4 N/C
	5 GND
	6 N/C
	7 CTS
	8 RTS
	9 DATA-

Terminal No.	Pin Assignment
COM3	1 CTS3
	2 RTS3
	3 RXD3
	4 TXD3
	5 GND
6 DO1	
7 DI3	
8 DI2	
9 INIT*	
COM2	10 D2+
	11 D2-
12 Vs+	
13 GND	

■ I-7522A(D)



Terminal No.	Pin Assignment	
1	DO (0)	
2	DI (0)	
COM1	3	D1+
	4	D1-
	5	CTS1
	6	RTS1
	7	GND
	8	TXD1
	9	RXD1
	10	INIT*
COM2	11	D2+
	12	D2-
	13	Vs+
	14	GND

Terminal No.	Pin Assignment	
COM3	15	TXD3+
	16	TXD3-
	17	RXD3+
	18	RXD3-
19	DI1	
20	DI2	
21	DI3	
22	DI4	
23	GND	
24	PWR	
25	DO1	
26	DO2	
27	DO3	
28	DO4	

■ I-7523(D)



Terminal No.	Pin Assignment
COM1	1 DATA+
	2 TxD
	3 RxD
	4 N/C
	5 GND
	6 N/C
	7 CTS
	8 RTS
	9 DATA-

Terminal No.	Pin Assignment
COM3	1 CTS3
	2 RTS3
	3 RXD3
	4 TXD3
5 GND	
COM4	6 TXD4
	7 RXD4
8 DI2	
9 INIT*	
COM2	10 D2+
	11 D2-
12 Vs+	
13 GND	

■ I-7524(D)



Terminal No.	Pin Assignment	
1	DO	
2	DI	
COM1	3	D1+
	4	D1-
	5	CTS1
	6	RTS1
	7	GND
	8	TXD1
	9	RXD1
10	INIT*	
COM2	11	D2+
	12	D2-
13	Vs+	
14	GND	

Terminal No.	Pin Assignment	
COM3	15	CTS3
	16	RTS3
	17	TXD3
	18	RXD3
19	GND	
COM4	20	CTS4
	21	RTS4
	22	TXD4
	23	RXD4
24	GND	
COM5	25	CTS5
	26	RTS5
	27	TXD5
	28	RXD5

■ I-7527(D)

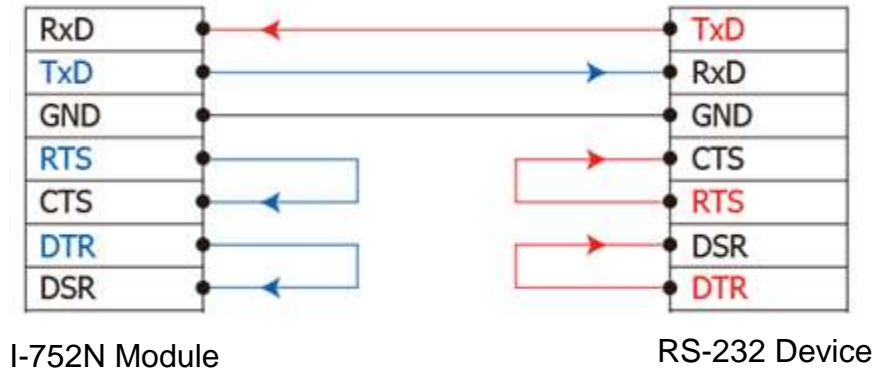


Terminal No.	Pin Assignment	
1	DO	
2	DI	
COM1	3	D1+
	4	D1-
	5	CTS1
	6	RTS1
	7	GND
	8	TXD1
	9	RXD1
10	INIT*	
COM2	11	D2+
	12	D2-
13	Vs+	
14	GND	

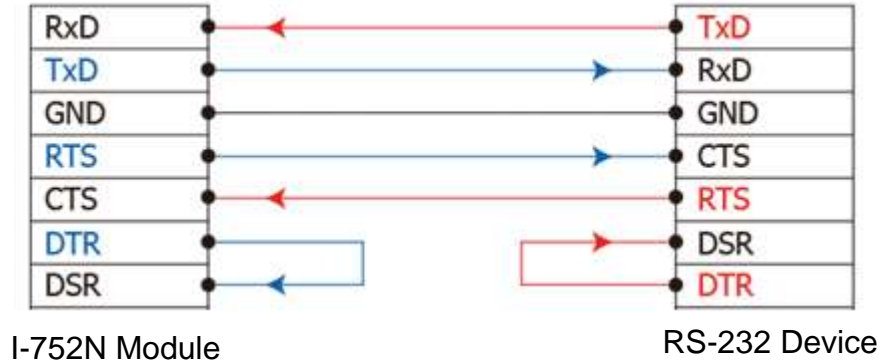
Terminal No.	Pin Assignment	
COM3	15	RXD3
	16	TXD3
COM4	17	RXD4
	18	TXD4
	19	GND
COM5	20	RXD5
	21	TXD5
COM6	22	RXD6
	23	TXD6
	24	GND
COM7	25	RXD7
	26	TXD7
COM8	27	RXD8
	28	TXD8

2.4. Wiring Connections

■ 3-wire RS-232 Wiring

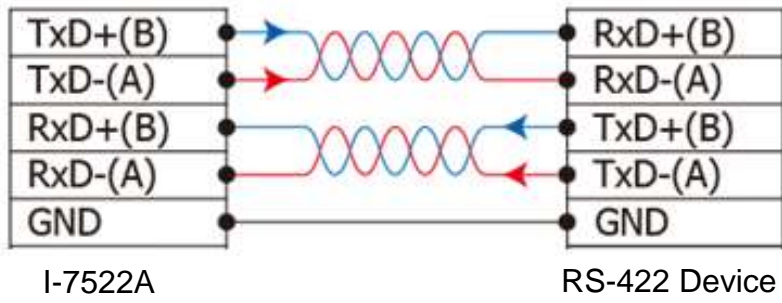


■ 5-wire RS-232 Wiring

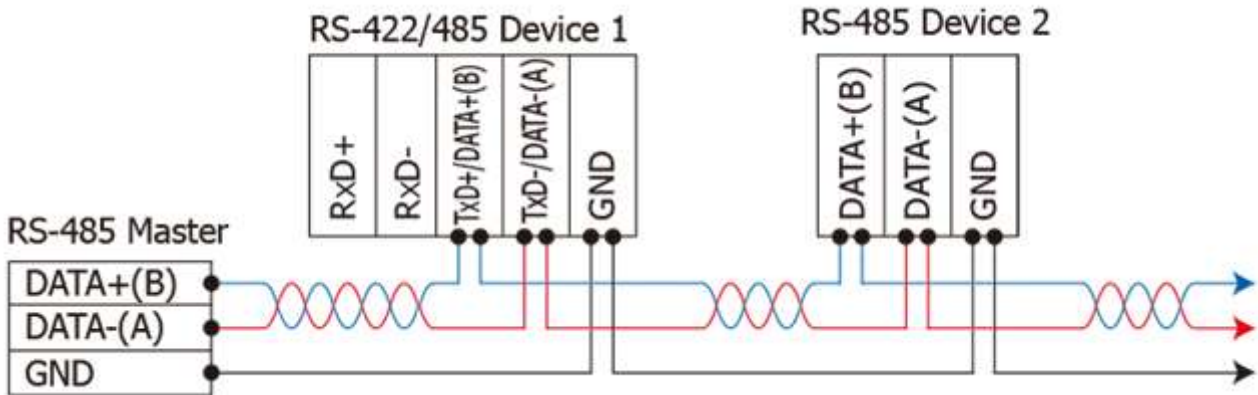


For 3-wire RS-232 connections, it is recommended to short unused signals such as RTS/CTS and DTR/DSR, since some system may still check the CTS and DSR status.

■ RS-422 Wiring

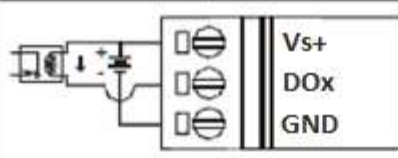
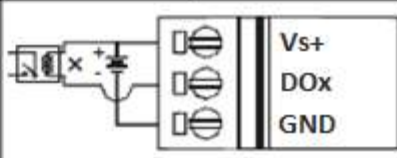
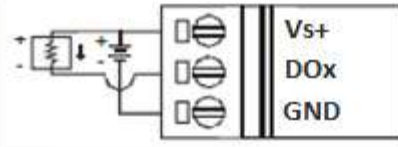
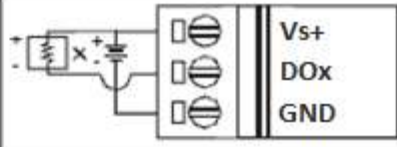


■ RS-485 Wiring

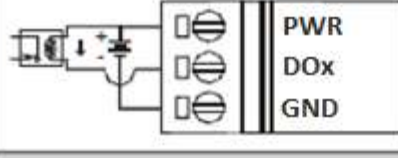
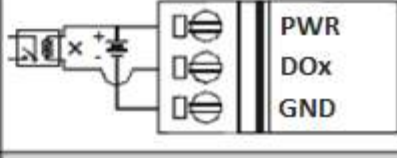
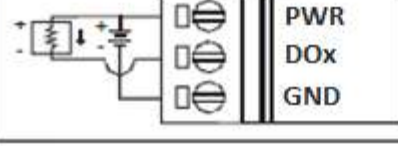



■ DO Wiring

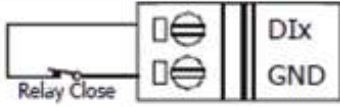
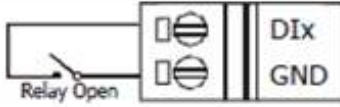
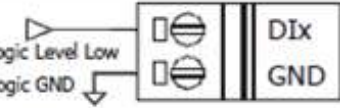
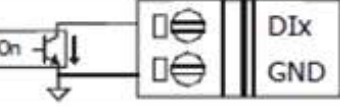
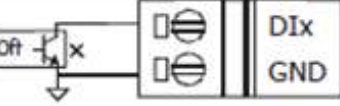
◇ I-7521/I-7522/7524/I-7527

Output Type	DO Command as 1	DO Command as 0
	Relay ON	Relay Off
Drive Relay		
Resistance Load		

◇ I-7522A

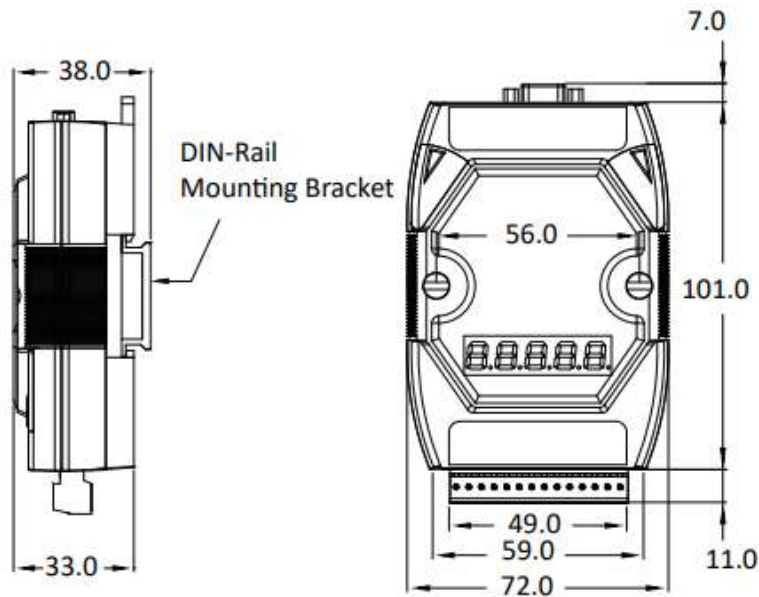
Output Type	DO Command as 1	DO Command as 0
	Relay ON	Relay Off
Drive Relay		
Resistance Load		

■ DI Wiring

Input Type	DI Value as 0	DI Value as 1
Relay Contact	Relay ON 	Relay Off 
	TTL/CMOS Logic	Voltage < 1V 
Open Collector	Open Collector On 	Open Collector Off 

2.5. Dimensions

I-7521(D)/I-7522(D)/I-7523(D)

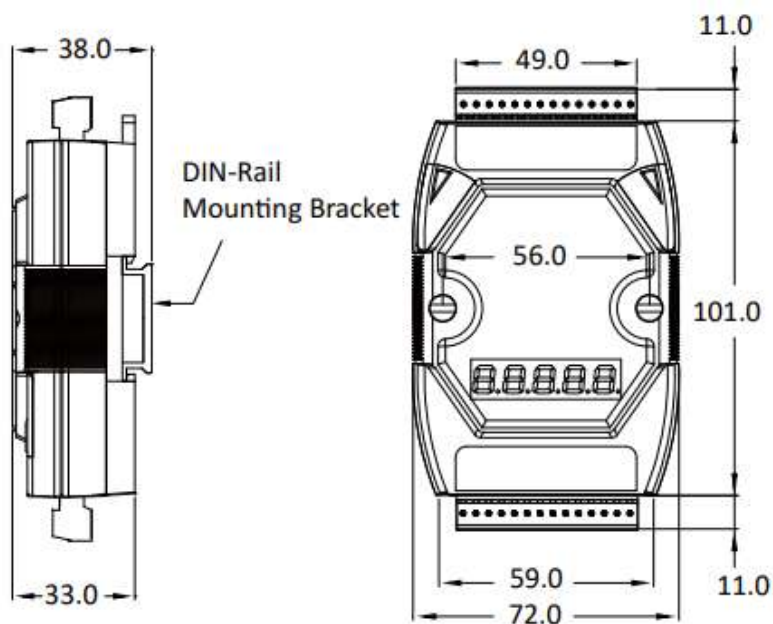


Side View

Front View

Back View

I-7522A(D)/I-7524(D)/I-7527(D)



Side View

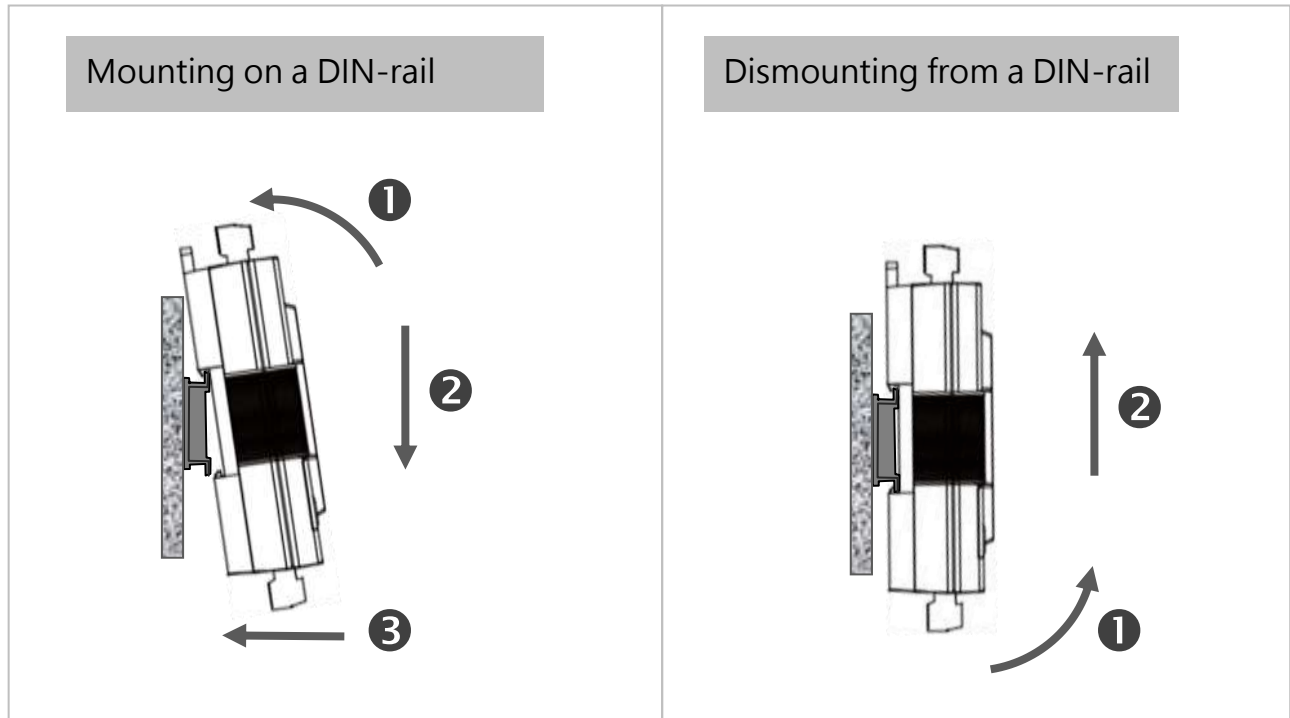
Front View

Back View

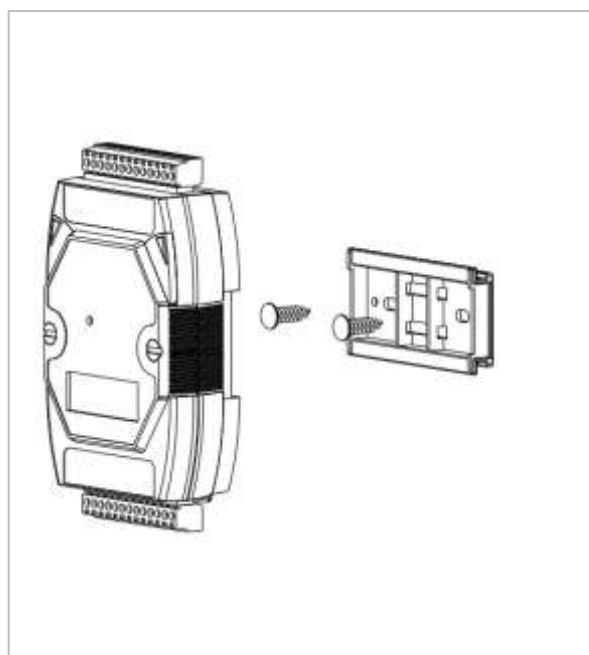
2.6. Mounting the Hardware

The I-752N series module contains simple rail clips to enable it to be reliably mounted on a standard 35-mm DIN rail.

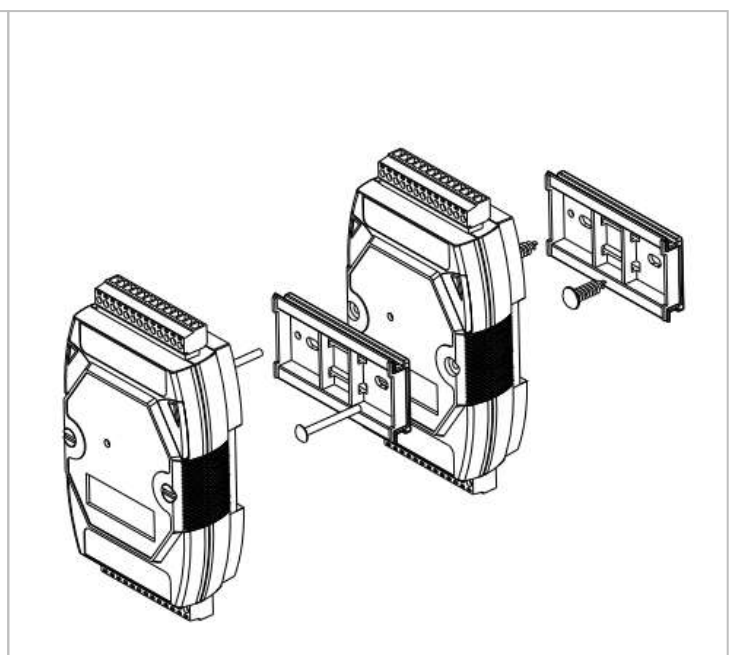
■ DIN-rail Mounting



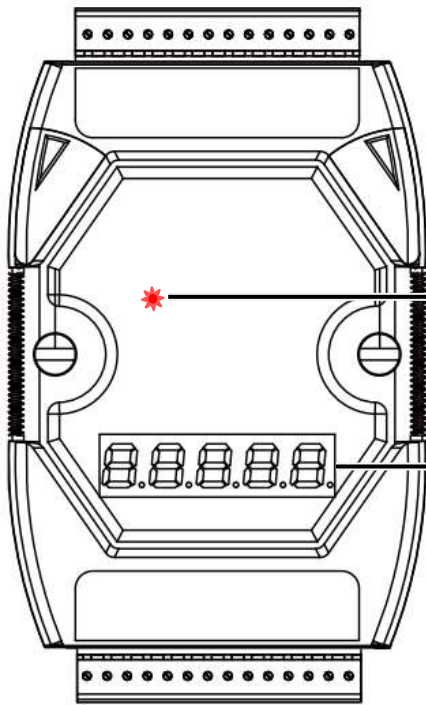
■ Wall Mounting



■ Piggyback Mounting



2.7. LED Indicator



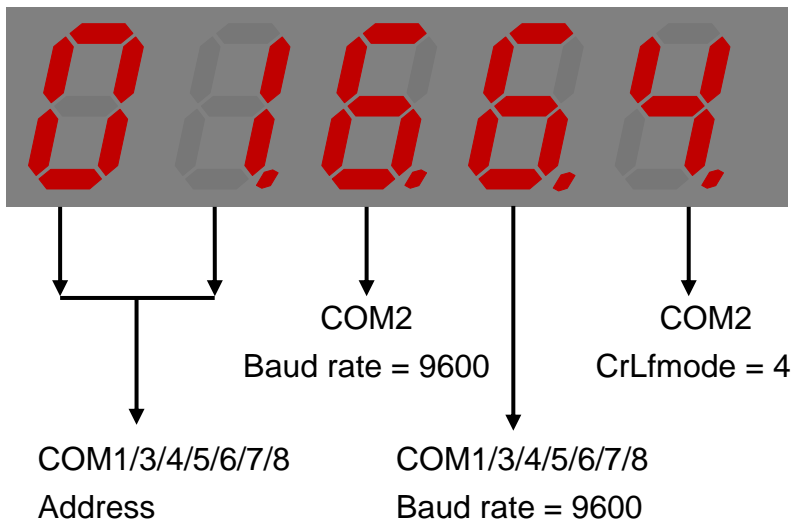
■ System LED indicator

Solid red with firmware V3

Blinking red with firmware V4

■ 7-segment LED display

For displaying communication settings of every COM port, available for D version only.

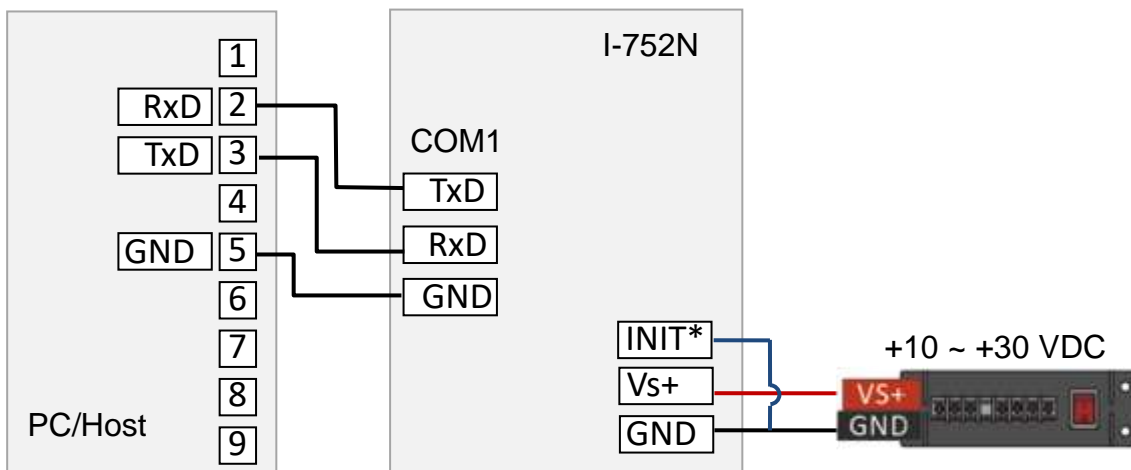


CrLfmode	
No.	End character
0	0x0D (CR)
1	0x0D+0x0A (CR+LF)
2	0x0A (LF)
3	0x0A+0x0D (LF+CR)
4	No end character
5	Modbus ASCII
6	Modbus RTU

Baud Rate (Unit: bps)							
No.	Baud Rate	No.	Baud Rate	No.	Baud Rate	No.	Baud Rate
1	300	4	2400	7	19200	A	115200
2	600	5	4800	8	38400		
3	1200	6	9600	9	57600		

2.8. INIT Mode

The I-752N series module functions in normal mode. When it is necessary to update firmware or operating system, the I-752N series module must start up in INIT mode (powering on the module with INIT* pin and GND pin shorted). At this time, the I-752N series module will not execute the firmware. After the update process is completed, remember to remove the lead wire between INIT* pin and GND pin, and then power cycle the I-752N series module to restore it to normal mode.



Refer to the wiring diagram above for using COM1 to communicate to the I-752N series module which is in INIT mode. When the module is in INIT mode, the 7-segment LED display on D-version module shows sequence numbers in 1 increment from 1 as below.



3. Getting Started

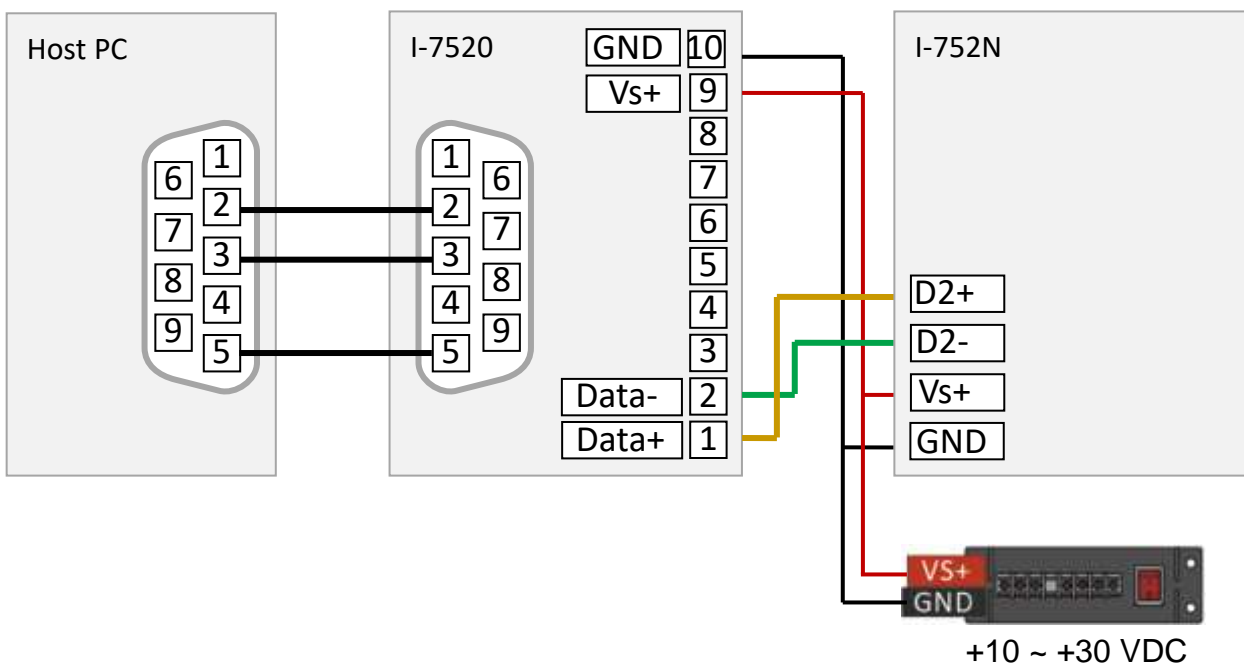
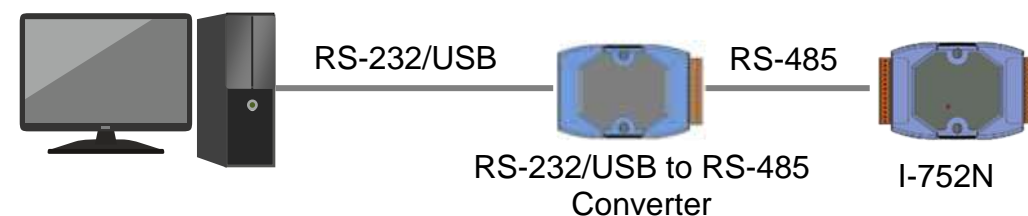
3.1. Connecting to Host PC and Power Supply

If your host PC/controller has RS-232 port(s), an RS-232 to RS-485 converter is needed to connect with the I-752N series module. Otherwise, a USB to RS-485 converter is useful, too.

Factory settings for I-752N series module

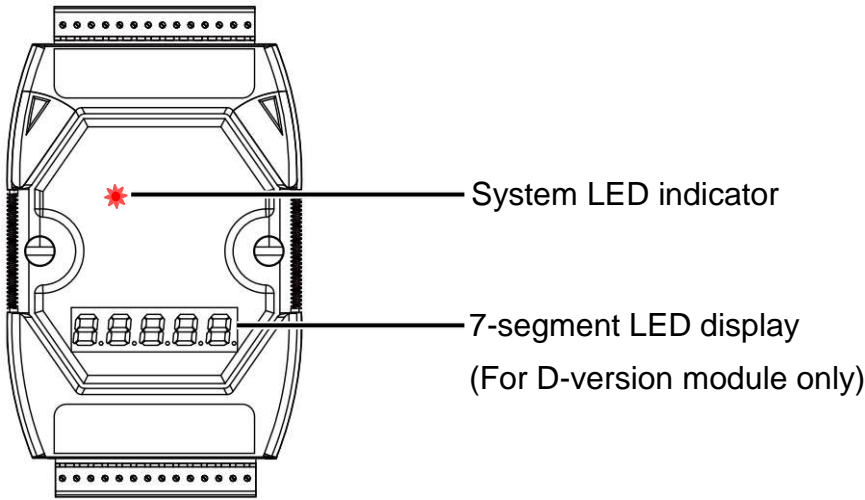
Address	1
RS-485/RS-232	
Baud Rate	9600
Data Format	N,8,1
Protocol	DCON
CrLf Mode (End Character)	0 (CR) in firmware V3.x 4 (no end character in firmware V4.x)

Step1. Connect the COM2 on the I-752N series module and RS-232/USB to RS-485 converter to the host PC. (If a USB to RS-485 converter is used, make sure that the driver for the converter is correctly installed.)

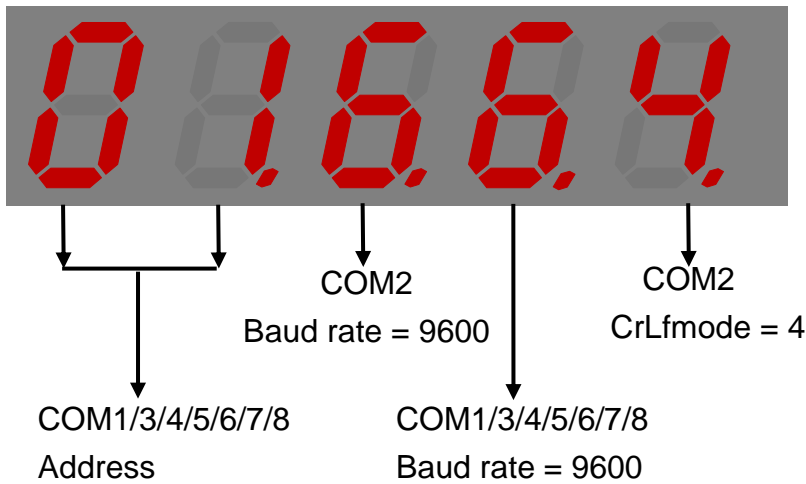


Step2. Confirm the status of system LED indicator.

(It is solid red with firmware V3.X and blinking red at about 1 Hz with V4.x.)



The 7-segment LED display on D version module shows the configuration for COM ports as



CrLfmode	
Code	End character
0	0x0D (CR)
1	0x0D+0x0A (CR+LF)
2	0x0A (LF)
3	0x0A+0x0D (LF+CR)
4	No end character
5	Modbus ASCII
6	Modbus RTU





Baud Rate (bps)							
Code	Baud Rate	Code	Baud Rate	Code	Baud Rate	Code	Baud Rate
1	300	4	2400	7	19200	A	115200
2	600	5	4800	8	38400		
3	1200	6	9600	9	57600		

3.2. Searching Module

Step3. Download DCON Utility Pro, unzip it and execute DCON_UTILITY_Pro.exe

The page for downloading the DCON Utility Pro:

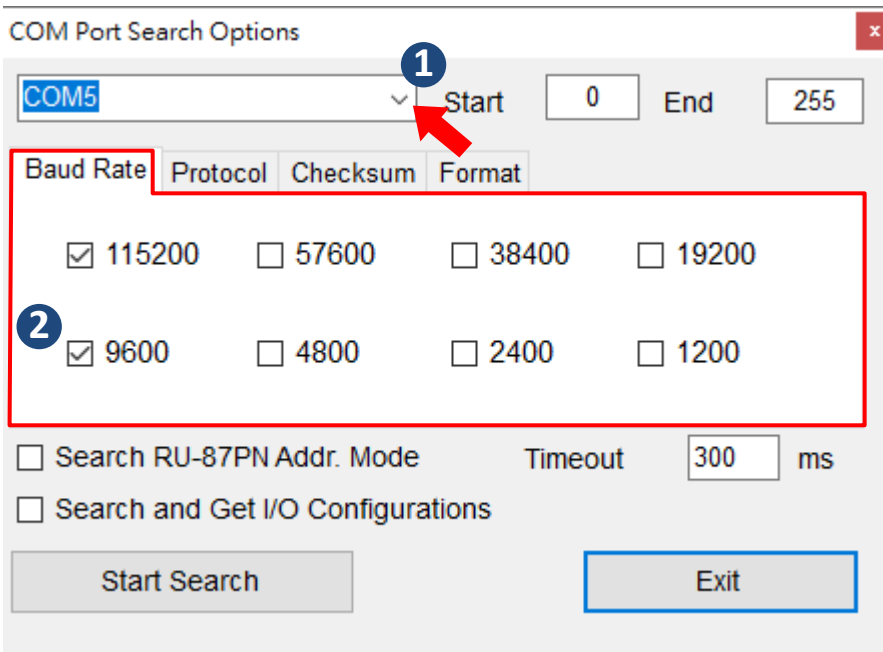
 <https://www.icpdas.com/en/download/show.php?num=1046>

FILE NAME	VERSION	FILE DATE	SIZE	NOTE	
DCON_UTILITY_Pro_PC_V4200	v4.2	2022-03-16	80.8 MB		
What's New					
Reversion History					
DCON_UTILITY_Pro_PC.zip	v3.1		28 MB		

Step4. Click the **Connection Options** icon on the toolbar to set communication configuration

The configuration for communicating to the COM2 on a brand new I-752N series module is module address 1, baud rate 9600 bps, data format 8/N/1.

1. Select the COM Port used to communicate to the I-752N series module.
2. Tick the checkbox for 9600 in the Baud Rate tab.



COM Port Search Options

COM5 Start 0 End 255

Baud Rate Protocol Checksum Format

115200 57600 38400 19200

9600 4800 2400 1200

Search RU-87PN Addr. Mode Timeout 300 ms

Search and Get I/O Configurations

Start Search Exit

3. Tick the checkbox for DCON in the Protocol tab.

Baud Rate Protocol Checksum Format

3

DCON Modbus RTU Modbus ASCII



4. Tick the checkbox for Checksum Disabled in the Checksum tab.

Baud Rate Protocol Checksum Format

4

Checksum Disabled Checksum Enabled



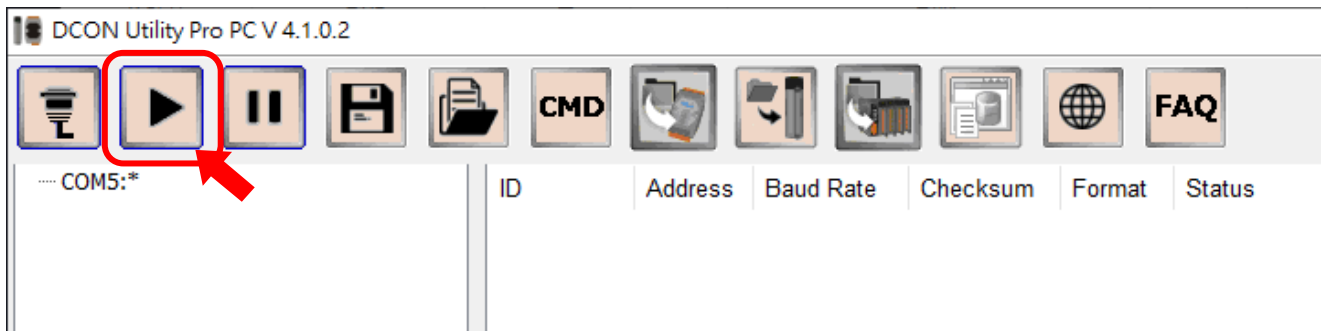
5. Tick the checkbox for N,8,1 in the Format tab.

Baud Rate Protocol Checksum Format

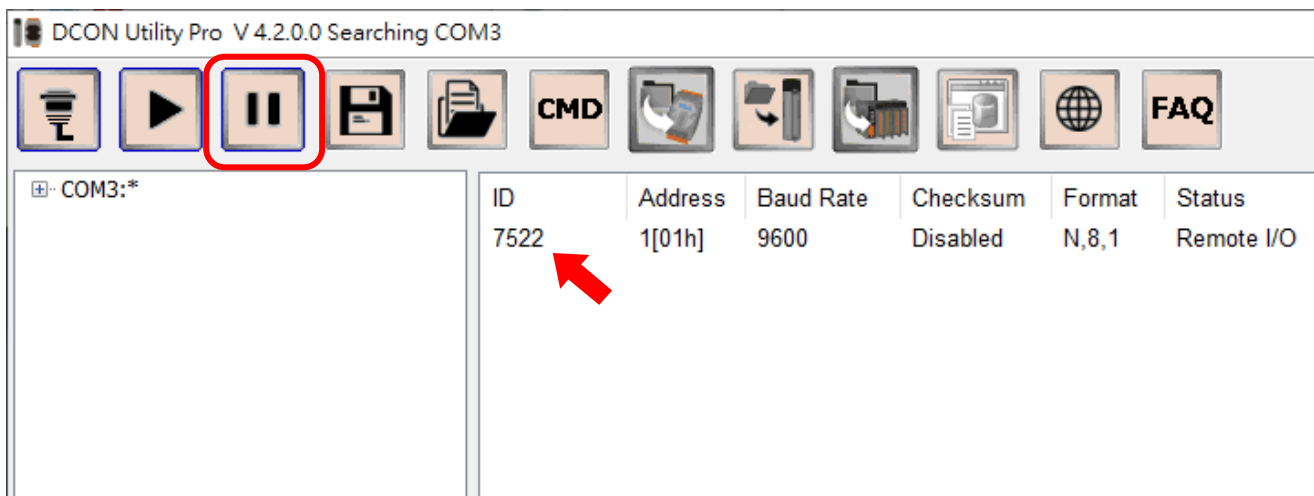
5

N,8,1 N,8,2 E,8,1 O,8,1

Step5. Click the **Start Search** icon on the toolbar to search module.

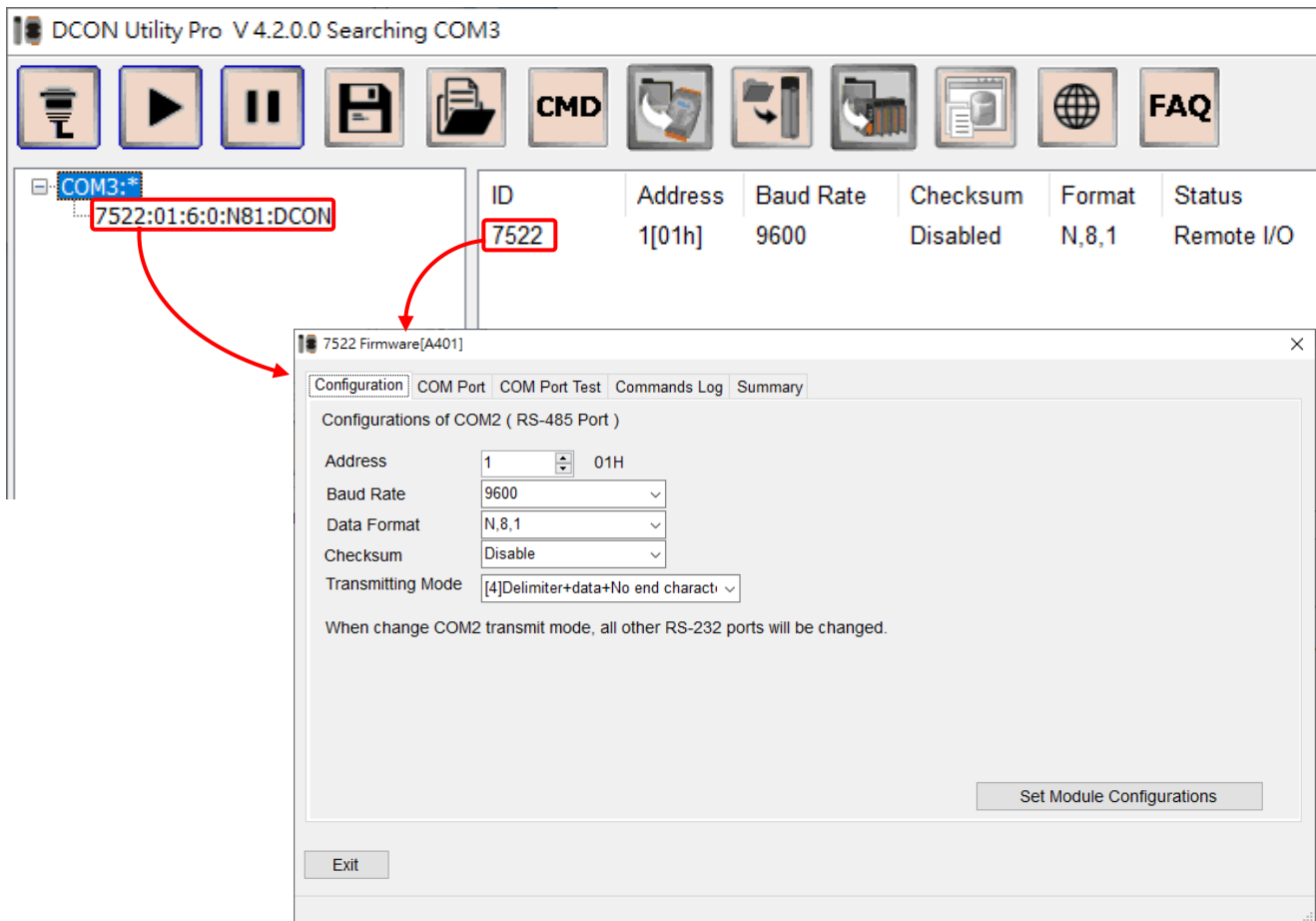


Click the **Stop Search** icon to stop search process if the module is found.

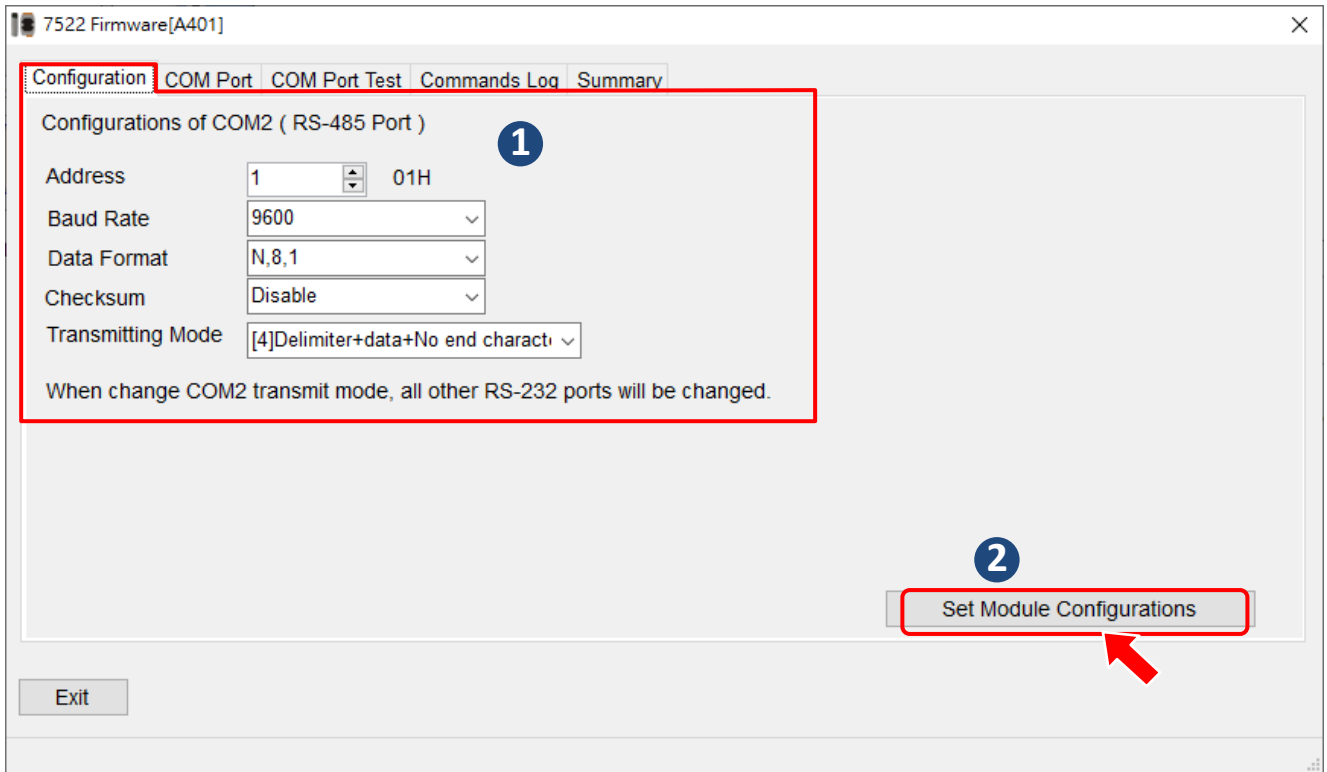


3.3. Setting Configuration

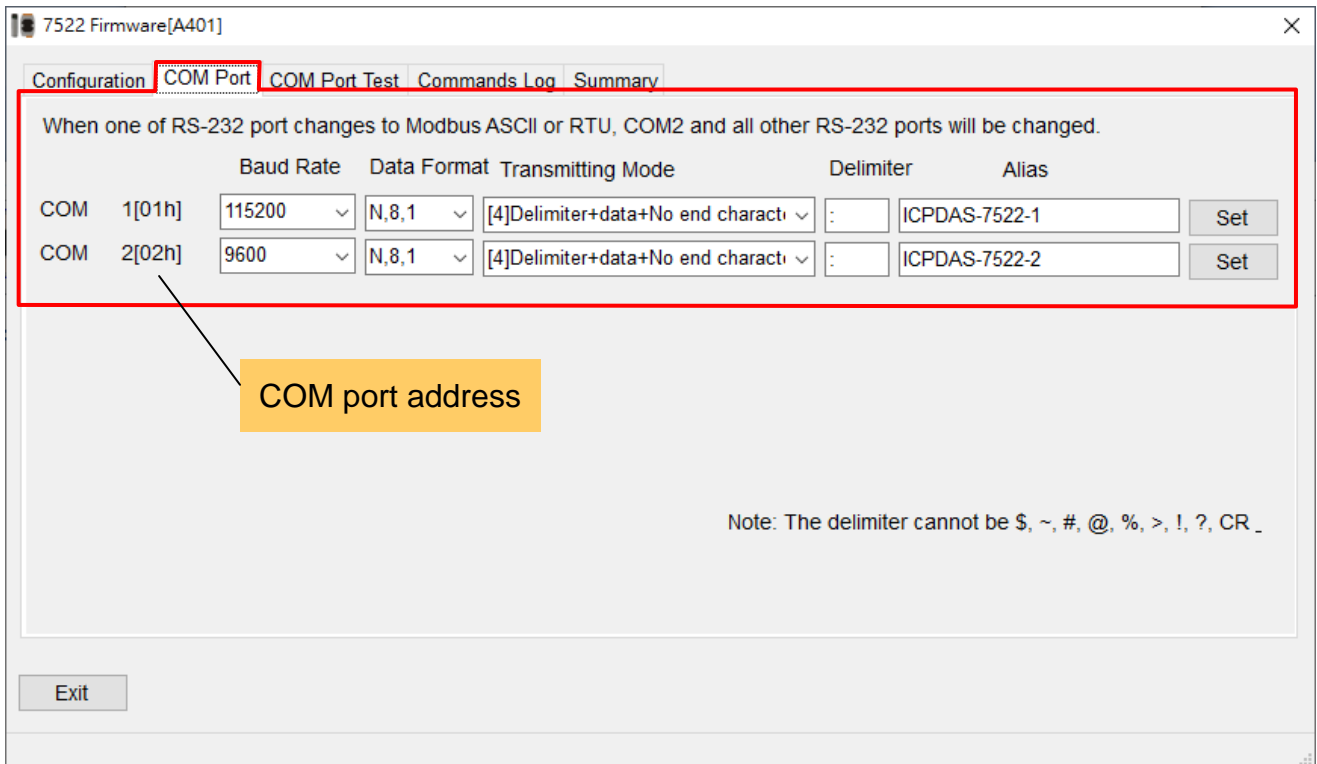
Step6. Click the module name to open the configuration window.



- ✧ Set the configuration and end character for COM2 and click the “**Set Module Configuration**” button in the Configuration tab to complete the operation.

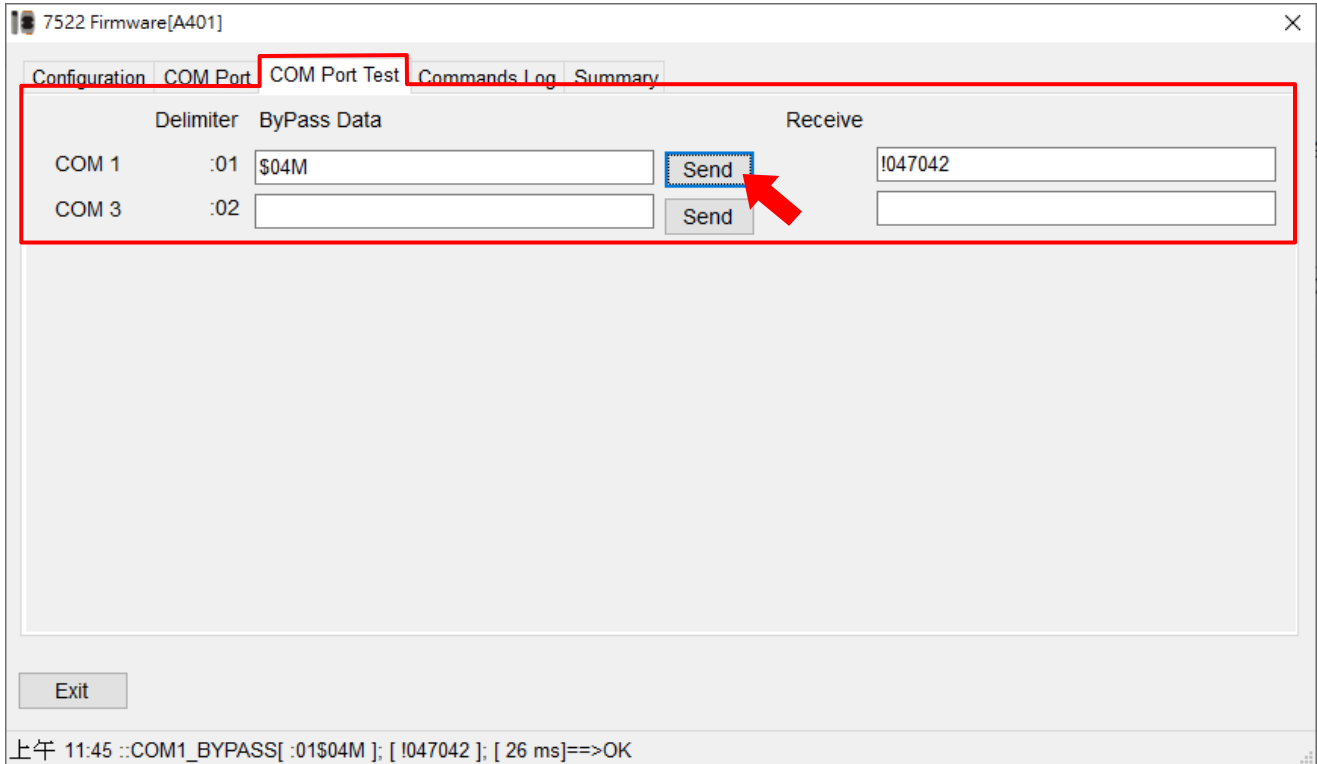


- ✧ Set the configuration and end character for COM1/3/4/5/6/7/8 in the COM port tab and click the “**Set**” button for each COM port to complete the operation.



3.4. Communication Testing

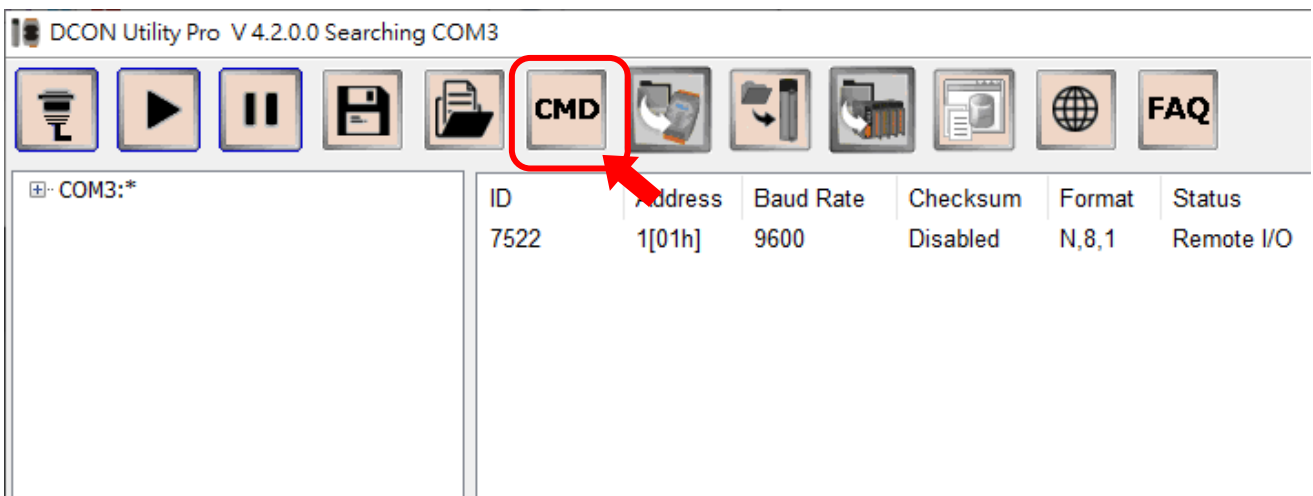
Enter commands for a device connected to a COM port on the I-752N series module, and then click the “**Send**” button to send the command to the device. Any response from the device will be displayed in the **Receive** text box.



3.5. Command Line Tool

The Command Line function can be used as a tool for module testing and debugging.

Step1. Click the “**Command Line**” icon to open the Command Line window.



Step2. Select correct Baud Rate, device address and the COM port that the I-752N series module is connected to.

Step3. Enter the command for the I-752N series module and then click the “**Send**” button to send the command. Any response from the module will be displayed in the Response text box.

The screenshot shows the 'Terminal Command Line Tool' window with the following configuration and actions:

- 1** COM Port: COM3
- 2** Baud Rate: 9600
- Checksum: Disabled
- Timeout: 100 ms
- Protocol: DCON
- Format: N,8,1-None Parity
- 3** Address: 1
- Select ID: 7522
- 4** Command: \$01M
- 5** Send button (highlighted)
- 6** Response: !017522

The response area shows the following log entry: 下午 01:21 :: [\$01M]; [!017522]; [20 ms]==>OK

Buttons at the bottom: Export Commands, Clear, Save to path\log_report\

Code	Description
1	Selects the COM port on the PC to connect the I-752N series module.
2	Selects the Baud Rate for communicating to the module.
3	Selects the address for the module
4	Enters command.
5	Clicks the “Send” button to send command.
6	Displays the response from the device

4. Function Description

4.1. RS-232 Port Address

The I-752N series module enables the host to communicate to several RS-232 devices through one RS-232 or USB port by connecting these RS-232 devices to an RS-485 network and assigning a unique address for each. A multi-port I-752N series module can be regarded as a combination of multiple I-7521 modules. Each one has a unique address and connects one RS-232 device to the RS-485 network.

Based on the module address AA:

the connection between COM2 and COM1 functions like the first I-7521 of address AA
the connection between COM2 and COM3 functions like the second I-7521 of address AA + 1,
the connection between COM2 and COM4 functions like the third I-7521 of address AA + 2,
and so on.

In other words,

I-7521 occupies one address, which is the module address AA

I-7522 occupies two addresses, which are AA and AA+1,

I-7523 occupies three addresses, which are AA, AA+1 and AA+2

I-7524 occupies four addresses, which are AA, AA+1, AA+2 and AA+3

I-7527 occupies seven addresses, which are from AA to AA+6

The COM port addresses definition based on the module address AA:

	I-7521	I-7522	I-7523	I-7522A	I-7524	I-7527
COM1 (RS-232)/ (RS-485)	AA	AA	AA	AA	AA	AA
COM2(RS-485)	AA	AA	AA	AA	AA	AA
COM3 (RS-232)/(RS-422)	-	AA+1	AA+1	AA+1	AA+1	AA+1
COM4(RS-232)	-	-	AA+2	-	AA+2	AA+2
COM5(RS-232)	-	-	-	-	AA+3	AA+3
COM6(RS-232)	-	-	-	-	-	AA+4
COM7(RS-232)	-	-	-	-	-	AA+5
COM8(RS-232)	-	-	-	-	-	AA+6

The host PC sends commands to a specified RS-232 device by assigning the unique address for COM port which the device is connected to. If multiple I-752N series modules are used on the same RS-485 network, the address of each RS-232 port needs be unique.

Example: if three I-7527 modules are used on the same RS-485 network, each RS-232 port of them must be set with a unique address.

	Address	I-7527#1	I-7527#2	I-7527#3
COM1 (RS-232)/(RS-485)	AA	01	08	15
COM2(RS-485)	AA	01	08	15
COM3(RS-232)	AA+1	02	09	16
COM4(RS-232)	AA+2	03	10	17
COM5(RS-232)	AA+3	04	11	18
COM6(RS-232)	AA+4	05	12	19
COM7(RS-232)	AA+5	06	13	20
COM8(RS-232)	AA+6	07	14	21

4.2. End Character (CrLfmode)

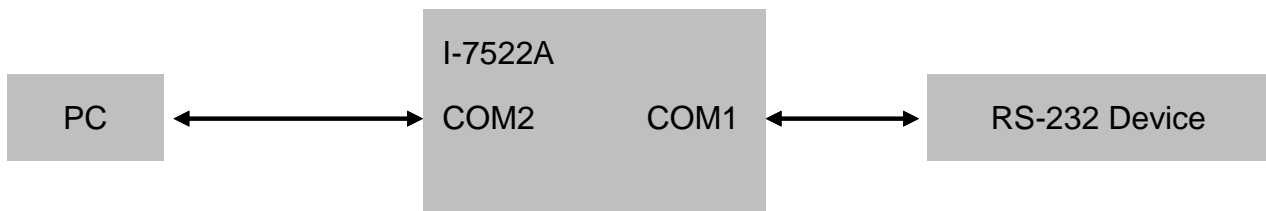
The I-752N series module provides 5 end character options; you can set different end character for each COM port according to the end character used in the protocol of connected device. The default end character is **0 (Cr) in firmware V3.x, but 4 (None) in firmware V4.x.**

CrLfmode	End character	Description
0	0x0D (Cr)	In firmware V3.x, a message is ended while the end character is received. In firmware V4.x, regardless of the ending character setting, a message is ended by timeout setting as CrLfmode = 4.
1	0x0D+0x0A (Cr+Lf)	
2	0x0A (Lf)	
3	0x0A+0x0D (Lf+Cr)	
4	No end character	If there is no data to be transferred or received before timeout expired, the communication is over.

When the CrLfmode is set to 4, (no end character), the I-752N series module will end a data transfer process if there is no data to be sent or received before timeout expired.

For example, If a string ":01T" is sent by enter ":0" first, and then "1T" after 20 ms, but timeout is set to 10 ms, the I-752N series module will treat ":0" as one message and "1T" as another one. Because the timeout interval is too small, using communication software is much better than key in the command string manually.

The end character for a COM port needs be set to match the end character used in protocol of the connected device. The following examples illustrate the process flow of the CrLfmode setting in I-752N series module.



Command: Sends ":01ABCD<Cr>" from PC

Response: Device returns "EFGH<Cr>" after it received "ABCD<Cr>"

■ Timeout Setting

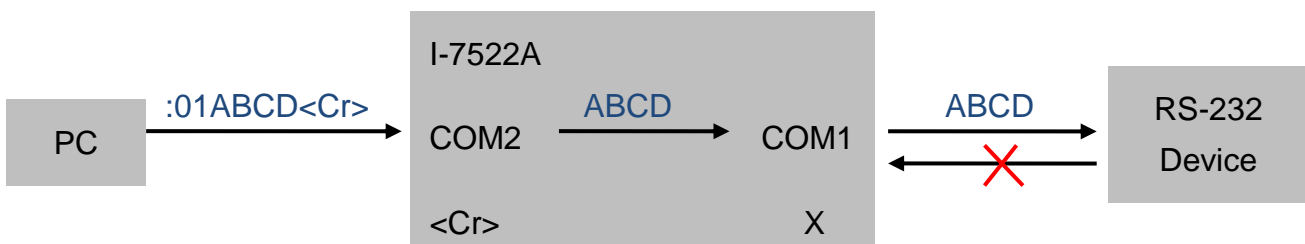
With firmware version v3.x and later, the command “\$AAJN[timeout]” is used to set timeout period. If the length of response string is longer, or the transfer speed of the response is slower, the timeout period can be extended to receive complete message. When a response is received within the timeout period, the data will be store in the COM port buffer, and “\$AAU” is used for reading data in the buffer.



When the I-752N series module receives a “:AAxxxxx” command from COM2 (PC), it will send the command “xxxxx” to the device by passing the string to the COM port (with address “AA”) to which that the device is connected, wait for a response from the device, and return the response to the host PC. Any data that is **not** received during the waiting time will be stored in the COM port buffer. “\$AAU” command is used to read data in the buffer.

■ Scenario 1

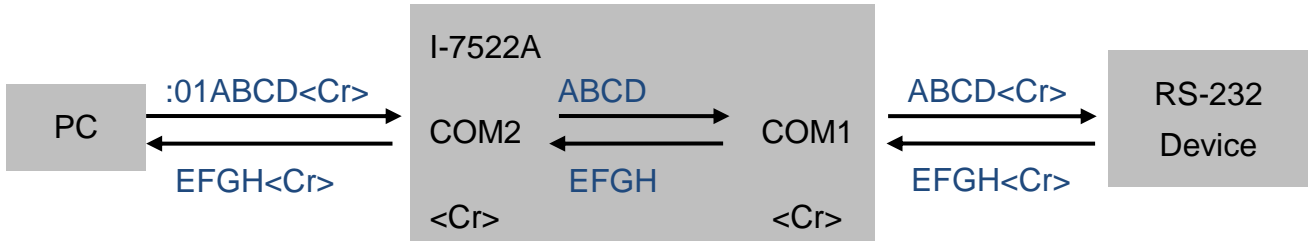
The CrLfmode for COM2 of I-7522A is set to 0 (Cr end character), for COM1, it is set to 4 (no end character). In addition, <Cr> is the end character in protocol of the RS-232 device.



When a command “:01ABCD<Cr>” is received from COM2 (PC), only valid string “ABCD” will be passed to COM1 to the RS-232 device without any modification, because COM1 is set to no end character. At this time, the RS-232 device will **not** response to the command because it does not receive a <Cr> end character.

■ Scenario 2

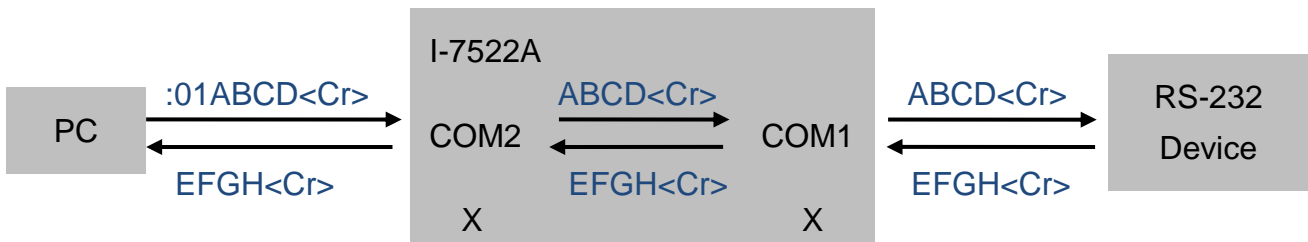
The CrLfmode for COM2 and COM1 of I-7522A are both set to 0 (Cr end character). Same as scenario 1, <Cr> is the end character in protocol of the RS-232 device.



Same as scenario 1, only string "ABCD" will be passed to COM1. It will be sent to the RS-232 device with <Cr> end character because Cr end character is set for COM1. The device receives the string "ABCD<Cr>", and gives the response "EFGH<Cr>".

■ Scenario 3

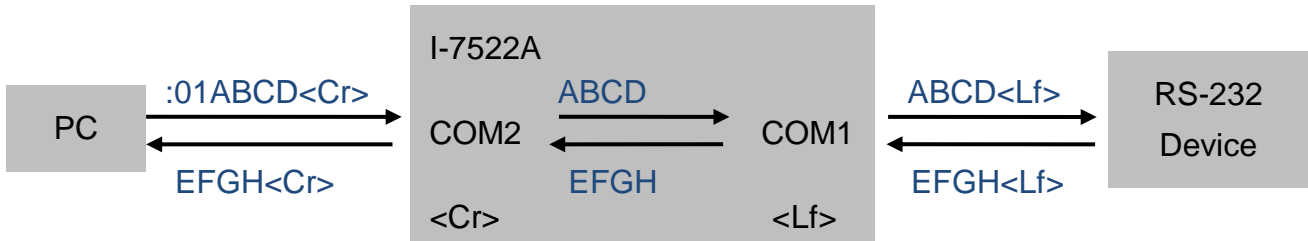
The CrLfmode for COM2 and COM1 of I-7522A are both set to 4 (no end character). Same as scenario 1, <Cr> is the end character in protocol of the RS-232 device.



Because the CrLfmode of COM2 is set to 4 (no end character), when it receives ":01ABCD<Cr>", it will bypass the whole string "ABCD<Cr>" to COM1. Likewise, the whole string "ABCD<Cr>" will be sent to the device because CrLfmode is set to 4 for COM1. The RS-232 device receives the string "ABCD<Cr>", and gives the response "EFGH<Cr>".

■ Scenario 4

The CrLfmodes for COM2 is set to 0 (Cr end character), while it is set to 2 (Lf end character) for COM1. In this scenario, <Lf> is the end character in protocol of the RS-232 device.



Because the Cr end character is removed at COM2, only valid string "ABCD" is passed to COM1. Then "ABCD<Lf>" will be sent to the RS-232 device because Lf end character is set for COM1. The device receives the string "ABCD<Lf>", and gives the response "EFGH<Lf>". For the same reason, the end character of response "EFGH<Lf>" will be removed and only string "EFGH" is sent to COM2. Then the <Cr> end character set for COM2 will be added to "EFGH" string when it is transmitted to the host PC.

4.3. Modbus ASCII/Modbus RTU Support (CrLfmode)

With the firmware V4.x and later, the I-752N series module supports Modbus ASCII and Modbus RTU protocols which are enabled by setting the CrLfmode. If any COM port is set to support Modbus ASCII or Modbus RTU protocol, all other COM port(s) on the module are limited to support Modbus ASCII or Modbus RTU protocol only.

When the CrLfmode for COM2 is set to 5, all other COM port(s) will be set to 5, too. Users can set these COM port to mode 6 only, mode 0 to mode 4 are not support since mode 5 is enable. Likewise, if the CrLfmode for COM2 is set to 6, all other COM port(s) will be set to 6, too. Users can set these COM port to mode5, mode 0 ~ 4 are not supported.

CrLfmode	End Character	Description
0	0x0D (CR)	In firmware V3.x, a message is ended while the end character is received. In firmware V4.x, regardless of the ending character setting, a message is ended by timeout setting as CrLfmode = 4.
1	0x0D+0x0A (CR+LF)	
2	0x0A (LF)	
3	0x0A+0x0D (LF+CR)	
4	No end character (Default)	If there is no data to be transferred or received before timeout expired, the communication is over.
5	Modbus ASCII	Forwards the message to COM Port without checking the LRC. The end character is Cr + Lf.
6	Modbus RTU	If there is no data received before timeout expired, the communication is over.



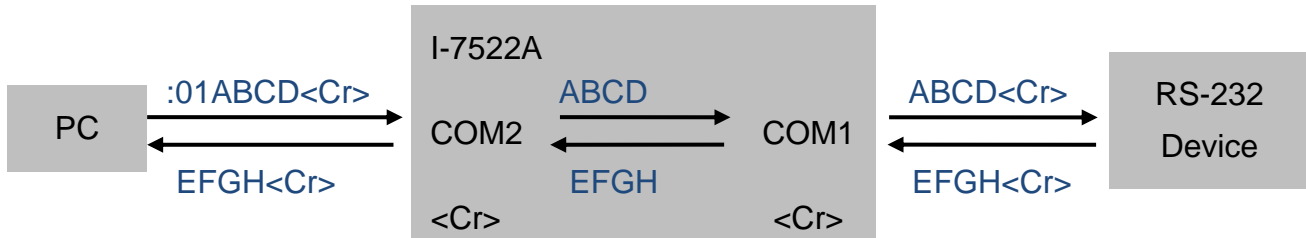
Even the CrLfmode for COM2 is set to mode 5 or 6, it can execute and reply the setting/reading commands of the I-752N series module.

4.4. RS-232 Device Communication

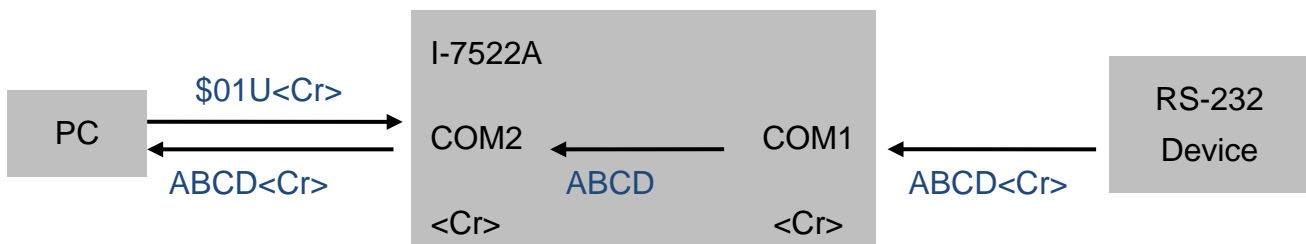
[delimiter]AA[bypass] command (section 7.4.10) is used to send data to a serial device connected to the COM port with address "AA". It will wait for a response from the device, and return the response to the host PC. Any data that is **not** received during the waiting time will be stored in COM port buffer. **“\$AAU”** command (section 7.4.30) can read data in buffer.

4.5. Bypass Data to COM2

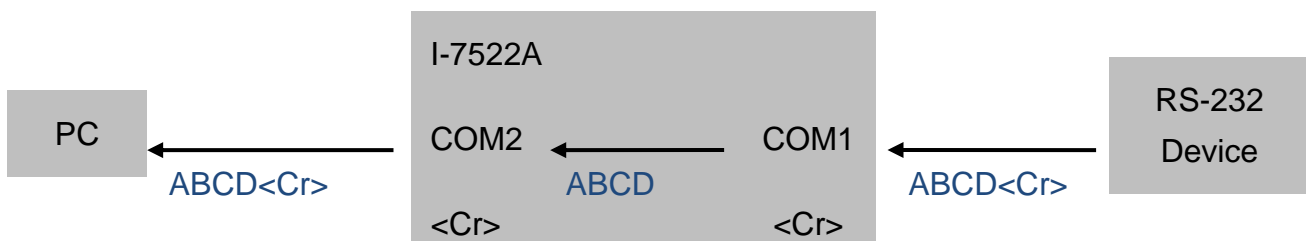
Since RS-485 is a half-duplex communication network, it cannot transmit and receive simultaneously. The communication is mainly done by sending commands to the device from host/PC, and then waiting the response from the device.



Any data that is not received during the waiting time for a response will be placed in COM port buffer. “\$AAU” command is used to read data in buffer.



“\$AAHV” command (section 7.4.23) is used to enable or disable the function of directly bypassing the data received from the RS-232 port to COM2. Some devices such as barcode scanners and electronic scales may transmit scanned (read) data one by one without commands. In such applications, the I-752N series module can be set to bypass data from connected devices to COM2 (PC) without sending commands.



In such applications, all the COM ports on a module should be set to enable the bypass function at a time. Besides, only one I-752N series module can be used on a RS-485 network, and PC cannot send data to the RS-485 network.

4.6. DI/DO

The I-752N series module provides DI/DO function, which can be used to turn on/off devices or alarm (DO), or obtain the status of buttons or switches. The DO/DI channels vary on the I-752N series modules as listed in the table below.

	I-7521	I-7522	I-7523	I-7522A	I-7524	I-7527
DI	2	2	1	5	1	1
DO	3	1	-	5	1	1

The data bit used to represent channel number varies from module to module. Refer to the following tables to confirm the data bit information for accessing status of DI/DO channels.

The corresponding table for DI channel number and the reading value

	Bit0	Bit1	Bit2	Bit3	Bit4
I-7521	-	DI2	DI3	-	-
I-7522	-	DI2	DI3	-	-
I-7523	-	DI2	-	-	-
I-7522A	DI	DI1	DI2	DI3	DI4
I-7524	DI	-	-	-	-
I-7527	DI	-	-	-	-

The corresponding table for DO channel number and the setting value

	Bit0	Bit1	Bit2	Bit3	Bit4
I-7521	DO1	DO2	DO3	-	-
I-7522	DO1	-	-	-	-
I-7523	-	-	-	-	-
I-7522A	DO	DO1	DO2	DO3	DO4
I-7524	DO	-	-	-	-
I-7527	DO	-	-	-	-

- ✧ The power-on value of DO channels are set by “~AA5P” command DO (section 7.4.50). “~AA4P” is used to read the power-on value (section 7.4.49).
- ✧ “\$AAZNV” command (section 7.4.36) can be used to read/set one DO channel.
- ✧ Since the module is powered on or restarted by the event of Module Watchdog timeout, DO channels stay with the status of power-on value, until a DO command is received.
- ✧ The DO channels will stay with the status of save value when Host Watchdog timeout occurs. At this time, DO commands will not be executed until the timeout status is cleared by “AA1” command (section 7.4.46).
- ✧ “#AABBHH” command (section 7.4.42) is used to control multiple DO channels, while “#AABCDD” command (section 7.4.43) is used to control one DO channel.
- ✧ “@AA[data]” command (section 7.4.41) can be used to read multiple DO and DI channels, or set the status of multiple DO channels.
- ✧ “\$AAYN” command (section 7.4.35) reads the status of one channel DI.
- ✧ In order to synchronously capture the DI status from multiple modules on the same RS-485 network, “#**” command (section 7.4.37) is provided. All modules connected to the RS-485 network will sample its DI status at the same while “#**” command is transmitted. “\$AA4” command (section 7.4.38) is used to read the synchronized DI data.

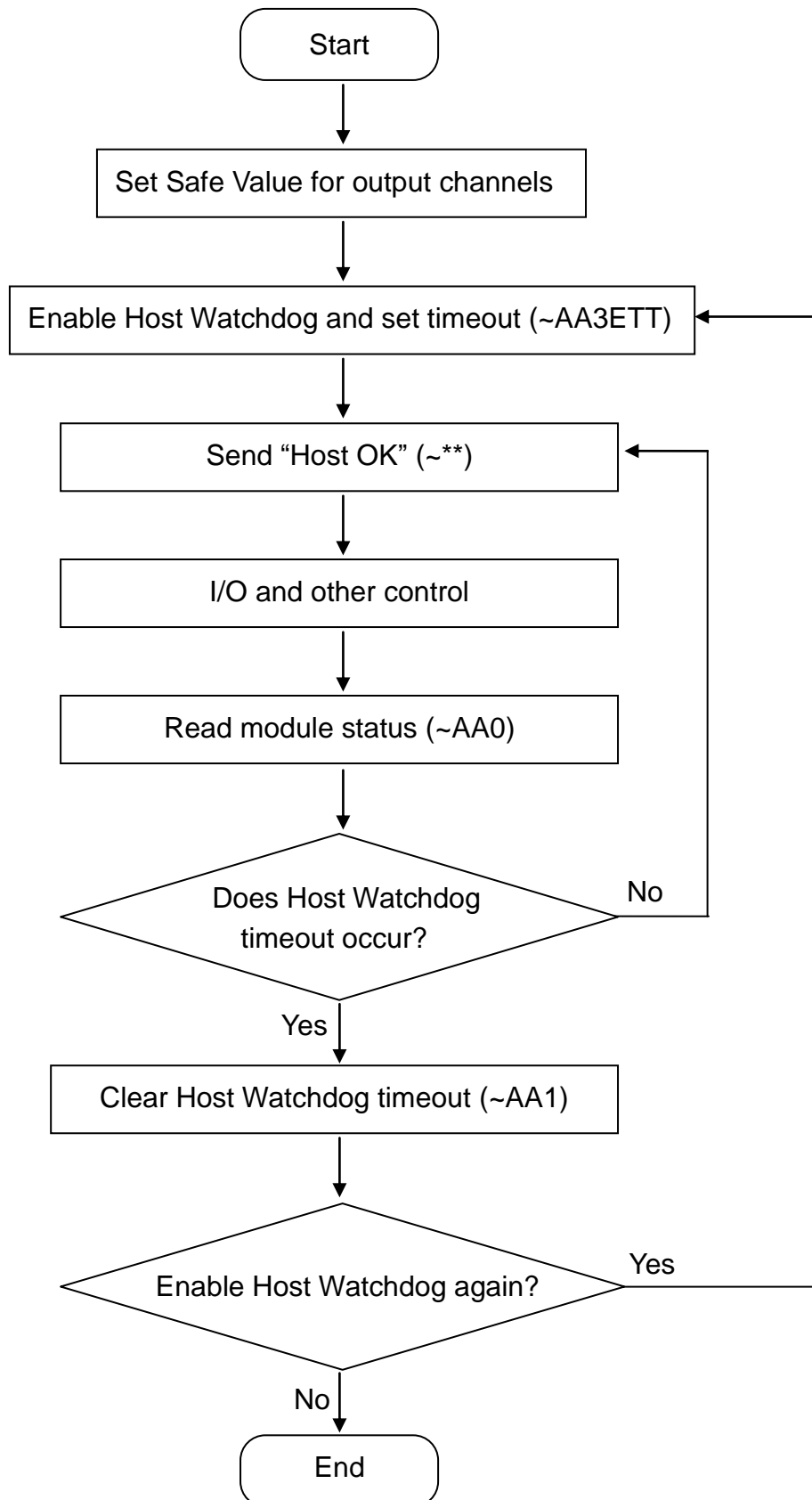
4.7. Dual Watchdog

All I-752N series modules are equipped with hardware Module Watchdog and software Host Watchdog. The I-752N series modules are designed for use in industrial areas, usually they have to operate in harsh environments with great amount of noise or energy transients. The dual watchdog feature is helpful to greatly increase the system reliability.

Module Watchdog is used to monitor the operating status of the module, it will reset the module if the module shuts down or malfunctions. In harsh or noisy environments, the module may fail to work properly due to noise interference or unexplained errors. At this point, Module Watchdog will automatically restart the module, allowing the module to continue to function. When the module restarts due to the event of Module Watchdog timeout, the digital output channels will remain in the power-on value for safety reasons. The output channels can be controlled by the host PC.

Host Watchdog is a software function; users need to enable the function to monitor the communication between the module and host PC. During the enablement of Host Watchdog function, the host PC needs to send a [Host OK] message to the module within the setting time. It is like a heartbeat message to show that the PC is work properly. If the module does not receive the message within the setting time, it determines that the host or communication is abnormal and raises a timeout error. At this time, the analog or digital output channels will stay with the status of safe value. The status of output channels **cannot** be changed until “~AA1” command is sent to clear the timeout error.

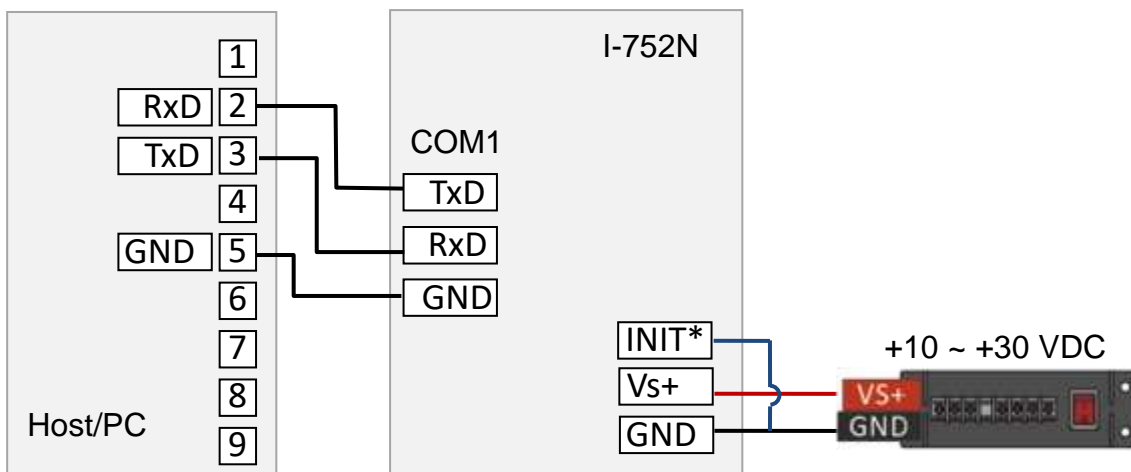
Host Watchdog Program Flowchart



5. INIT Mode and Firmware Update

5.1. INIT Mode

Generally, the I-752N series module functions in normal mode. If the firmware or operating system needs to be updated, the module should start in INIT mode by shorting the INIT* pin and GND pin and then powering on the module. When the module is in INIT mode, the firmware will not be executed automatically. After the update procedure is completed, remove the lead wire between INIT* pin and GND pin, and restart the module back to normal mode.



When the I-752N series module is in INIT mode, the host PC can only connect to COM1 on the module for communication. The wiring diagram is shown as the picture above.

On D-version module, the 7-segment LED display shows sequence numbers in 1 increment from 1 when it is in INIT mode.



5.2. 7188xw




7188xw.exe is used to update firmware or operating system for I-752N series module, transfer data to the module and the RS-232 devices connected to the module, and display the response from these modules and devices.

7188xw.exe is a software tool in the MiniOS7 Utility, you need to download and install the MiniOS7 Utility.

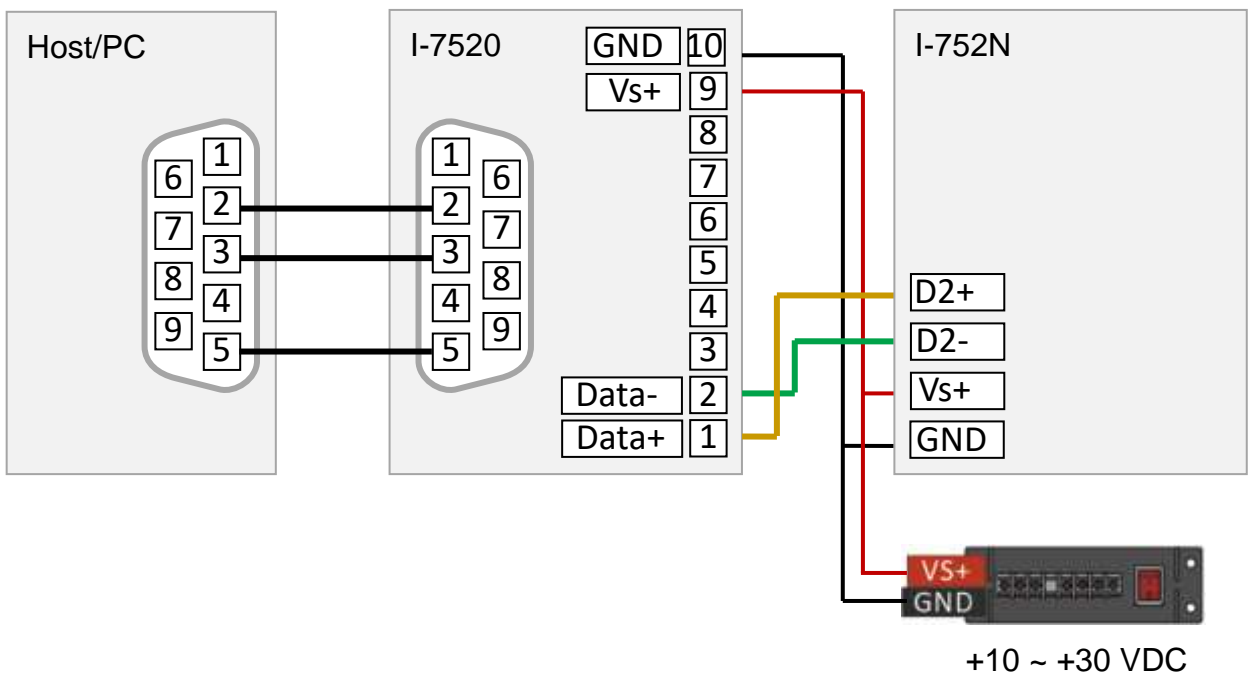
The link for downloading MiniOS7 Utility:



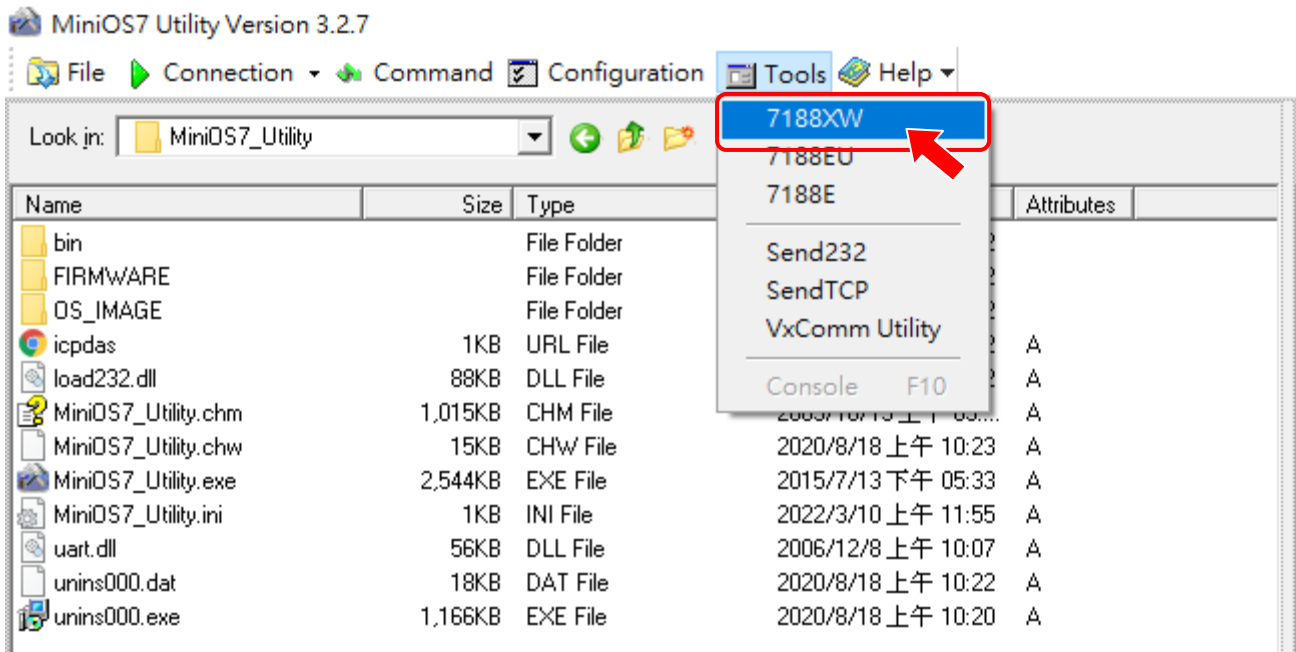
<https://www.icpdas.com/en/download/show.php?num=1053>

FILE NAME	VERSION	FILE DATE	SIZE	NOTE
MiniOS7_UTILITY_V327.exe	v3.2.7	2020-05-14	5.0 MB	
Readme (EN)		2020-05-14		
Readme (TC)		2020-05-14		

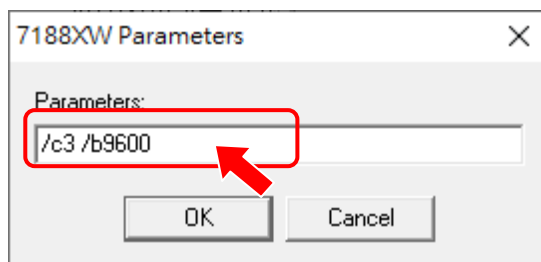
Before starting to use 7188xw.exe, connect the COM2 of the I-752N series module to host/PC and power on the module in normal mode.



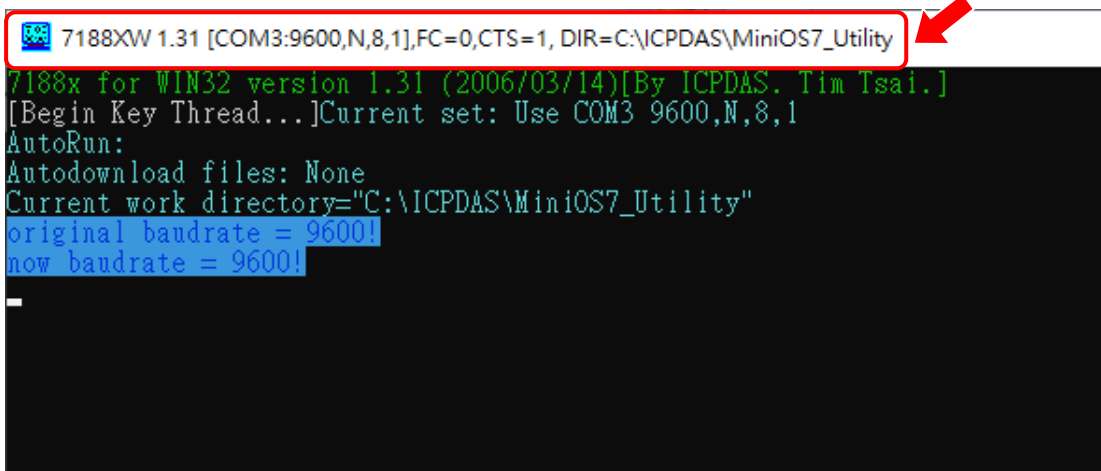
Step1. Launch **MiniOS7 Utility.exe** and select **7188XW** from the Tools menu on the toolbar.



Step2. Enter communication parameters. For example, “/c” + port number specifies the COM port on the PC, “/b” + baud rate value sets baud rate. “/c3 /b9600” means using COM3 on the PC and bard rate 9600 bps to communicate with the I-752N series module.



During the execution of 7188xw.exe, the communication parameters are real-time displayed on the title bar.



The communication parameters can be modified to meet real requirement during the program execution.

■ Changing COM Port

Press ALT + number key for COM Port at the same time. For example, press ALT + 1 to change the communication port to COM1.



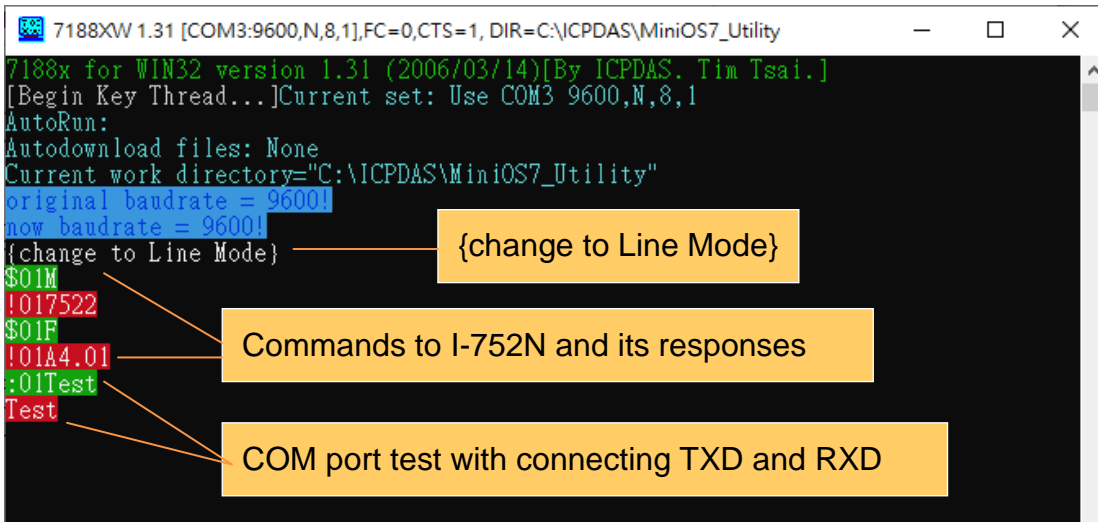
If COM port number is larger than 9, press ALT + C, and then enter “c” and com port number at the 7188xw_CMD> prompt. For example, c15 means to use COM15.

■ Sending command to the I-752N series module

Step1. Press ALT + L to enter Line Mode. (The I-752N series module is in normal mode)

"{change to Line Mode}" is displayed in 7188xw.

Step2. Enter commands for I-752N series module, e.g., \$01M to read module name, \$01F to read firmware version.



```
7188XW 1.31 [COM3:9600,N,8,1],FC=0,CTS=1, DIR=C:\ICPDAS\MiniOS7_Utility
7188x for WIN32 version 1.31 (2006/03/14)[By ICPDAS, Tim Tsai.]
[Begin Key Thread...]Current set: Use COM3 9600,N,8,1
AutoRun:
Autodownload files: None
Current work directory="C:\ICPDAS\MiniOS7_Utility"
original baudrate = 9600!
now baudrate = 9600!
{change to Line Mode}
$01M
!017522
$01F
!01A4.01
:01Test
Test
```

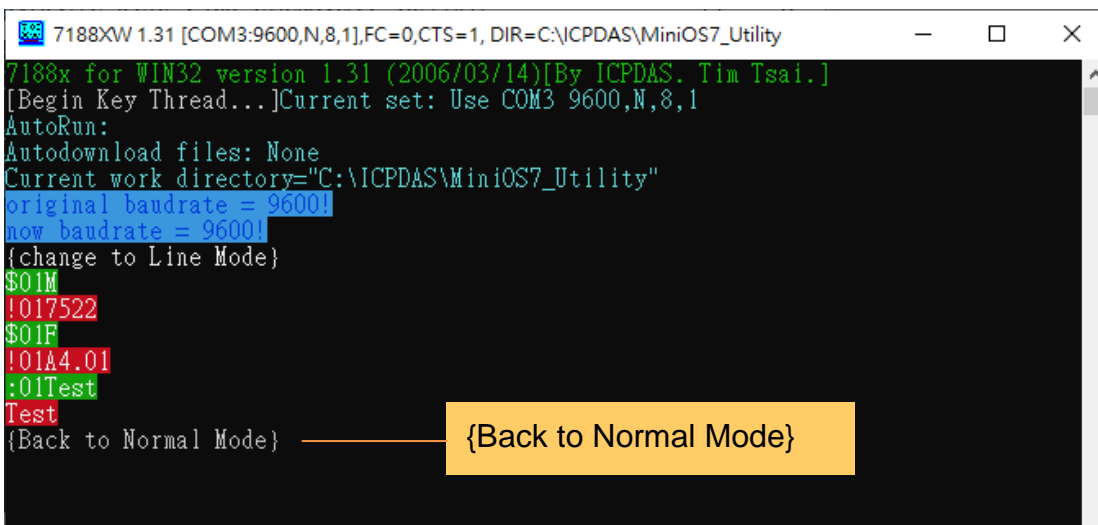
{change to Line Mode}

Commands to I-752N and its responses

COM port test with connecting TXD and RXD

Step3. If TXD and RXD are connected, the string send to COM port (COM1/3/4/5/6/7/8) will be received at the same port. It is an easy means to test communication of a COM port.

Step4. Press ALT + L to exit Line Mode. **{Back to Normal Mode}** is displayed.



```
7188XW 1.31 [COM3:9600,N,8,1],FC=0,CTS=1, DIR=C:\ICPDAS\MiniOS7_Utility
7188x for WIN32 version 1.31 (2006/03/14)[By ICPDAS, Tim Tsai.]
[Begin Key Thread...]Current set: Use COM3 9600,N,8,1
AutoRun:
Autodownload files: None
Current work directory="C:\ICPDAS\MiniOS7_Utility"
original baudrate = 9600!
now baudrate = 9600!
{change to Line Mode}
$01M
!017522
$01F
!01A4.01
:01Test
Test
{Back to Normal Mode}
```

{Back to Normal Mode}

5.3. Firmware Update

Step1. Download the latest firmware file and unzip it.

<https://www.icpdas.com/en/download/show.php?num=2623>

I-752N

I-752N Firmware offers the following files:

- I-7521, I-7522, I-7523 Firmware is support for I-7521, I-7522 and I-7523
- I-7522A, I-7524, I-7527 Firmware is support for I-7522A, I-7524 and I-7527

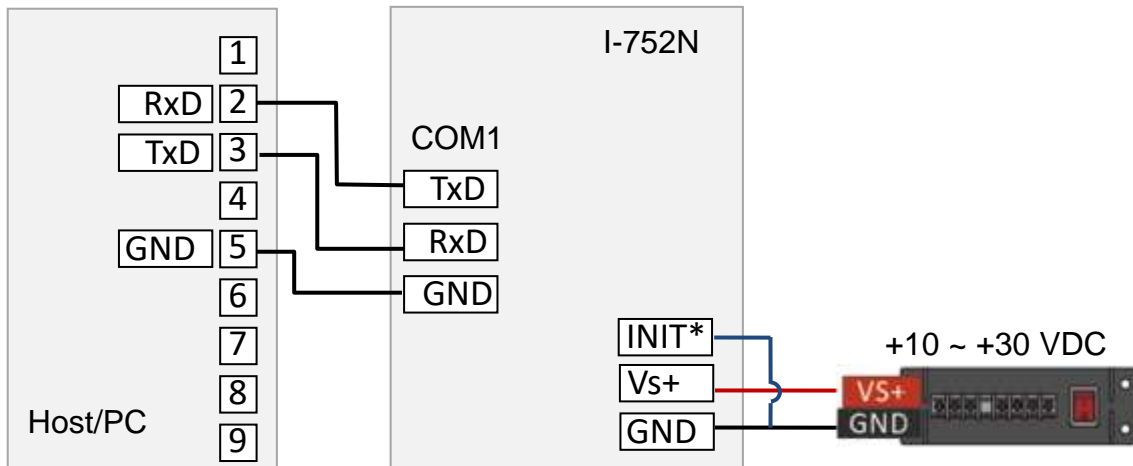
FILE	FILE NAME	VERSION	SIZE	DOWNLOAD
I-7521, I-7522, I-7523 Firmware	752N_C_20191002_V3.11.ZIP	-	18.4 KB	
I-7522A, I-7524, I-7527 Firmware	752N_B_20211015_V4.zip	-	32.0 KB	

Step2. Copy the files in unzipped folder to the installation directory of MiniOS7_Utility
(The default position is C:\ICPDAS\MiniOS7_Utility\)

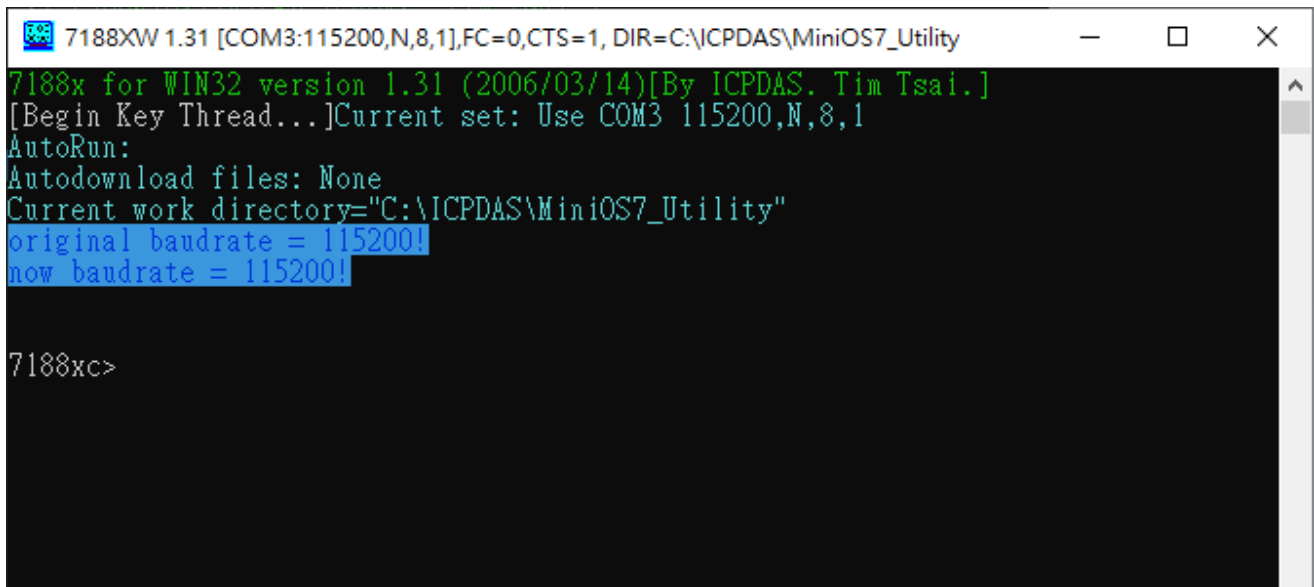
bin	2022/4/20 下午 04:01
FIRMWARE	2022/3/10 下午 01:53
OS_IMAGE	2022/3/10 下午 01:53
752N_C.EXE	2019/10/2 上午 10:06
7188XW.F4	2005/12/9 上午 09:25
autoexec.bat	2005/11/25 下午 12:37
icpdas	2022/3/10 下午 01:53
load232.dll	2007/1/31 下午 12:52
MiniOS7_Utility.chm	2009/10/15 上午 09:38
MiniOS7_Utility.exe	2015/7/13 下午 05:58
MiniOS7_Utility.ini	2022/4/20 下午 03:59

Firmware files

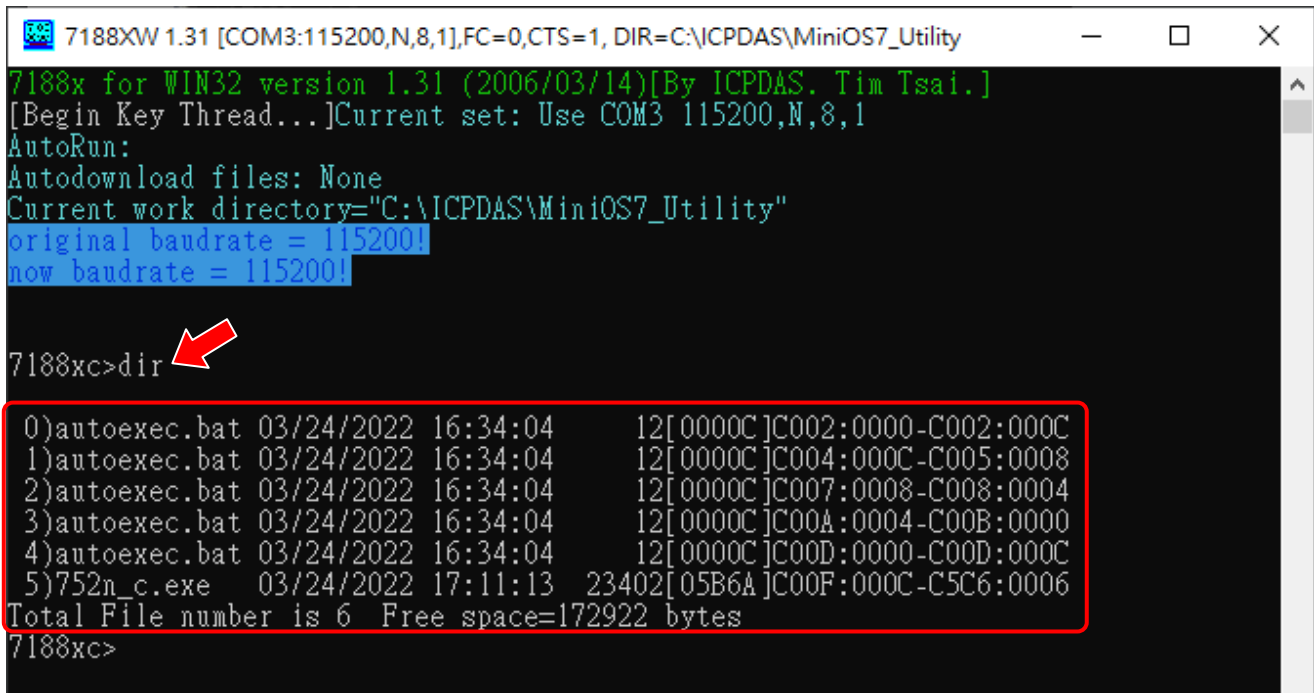
Step3. Connect the COM1 on the I-752N series module to the host/PC as the picture shown below, short the INIT* pin and GND pin, and then power on the module.



Step4. Set the configuration parameters to **baud rate 115200 bps, data format N, 8,1** in 7188XW.exe, and press **Enter** to connect to the I-752N series module. After the connection is established, the prompt “**7188xc>**” is displayed for I-7521/I-7522 /I-7523 or “**7188xc>**” is displayed for I-7522A/I-7524/I-7527.



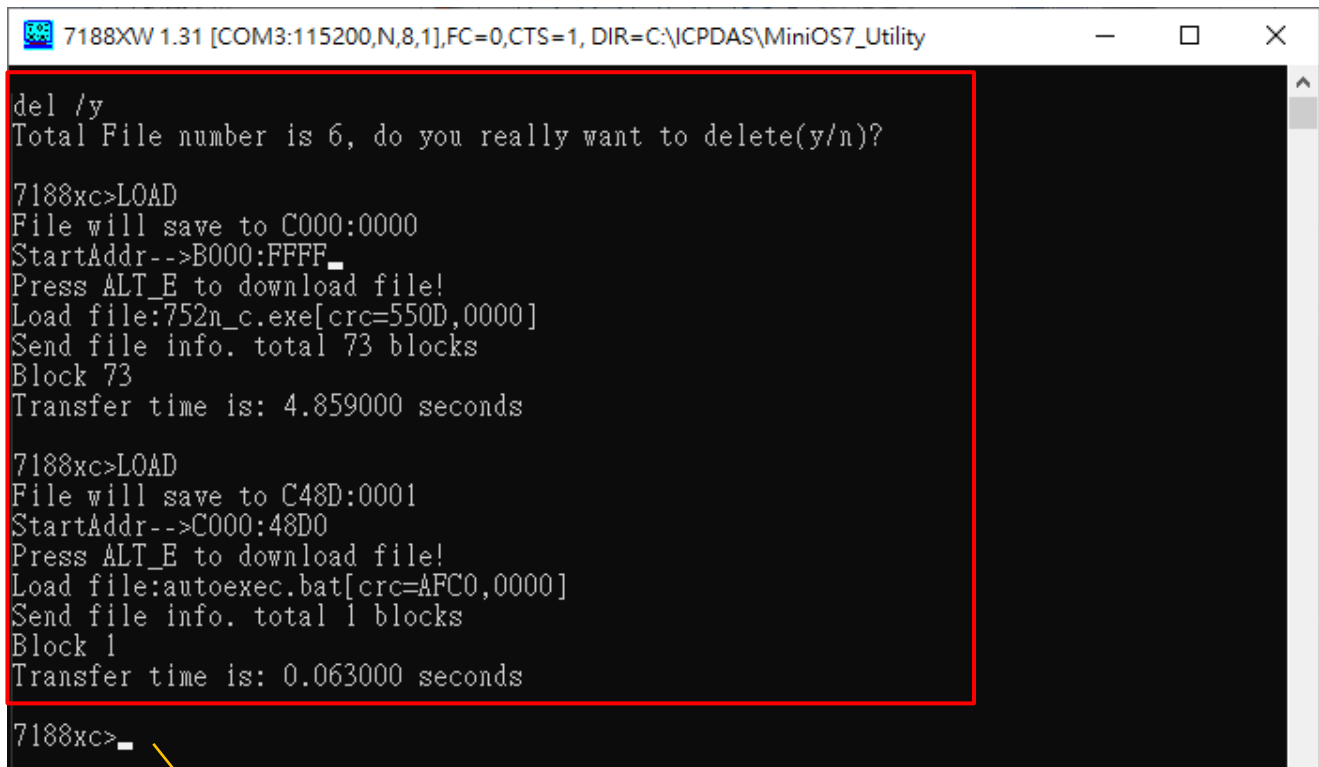
Type "dir" at the prompt and press **Enter** to list files in the module.



```
7188XW 1.31 [COM3:115200,N,8,1],FC=0,CTS=1, DIR=C:\ICPDAS\MiniOS7_Utility
7188x for WIN32 version 1.31 (2006/03/14)[By ICPDAS. Tim Tsai.]
[Begin Key Thread...]Current set: Use COM3 115200,N,8,1
AutoRun:
Autodownload files: None
Current work directory="C:\ICPDAS\MiniOS7_Utility"
original baudrate = 115200!
now baudrate = 115200!

7188xc>dir
0)autoexec.bat 03/24/2022 16:34:04 12[0000C]C002:0000-C002:000C
1)autoexec.bat 03/24/2022 16:34:04 12[0000C]C004:000C-C005:0008
2)autoexec.bat 03/24/2022 16:34:04 12[0000C]C007:0008-C008:0004
3)autoexec.bat 03/24/2022 16:34:04 12[0000C]C00A:0004-C00B:0000
4)autoexec.bat 03/24/2022 16:34:04 12[0000C]C00D:0000-C00D:000C
5)752n_c.exe 03/24/2022 17:11:13 23402[05B6A]C00F:000C-C5C6:0006
Total File number is 6 Free space=172922 bytes
7188xc>
```

Step5. Press the **F4** key once to start the update process, the execution progress will be displayed in the window. The process is complete while the prompt is shown again.



```
del /y
Total File number is 6, do you really want to delete(y/n)?

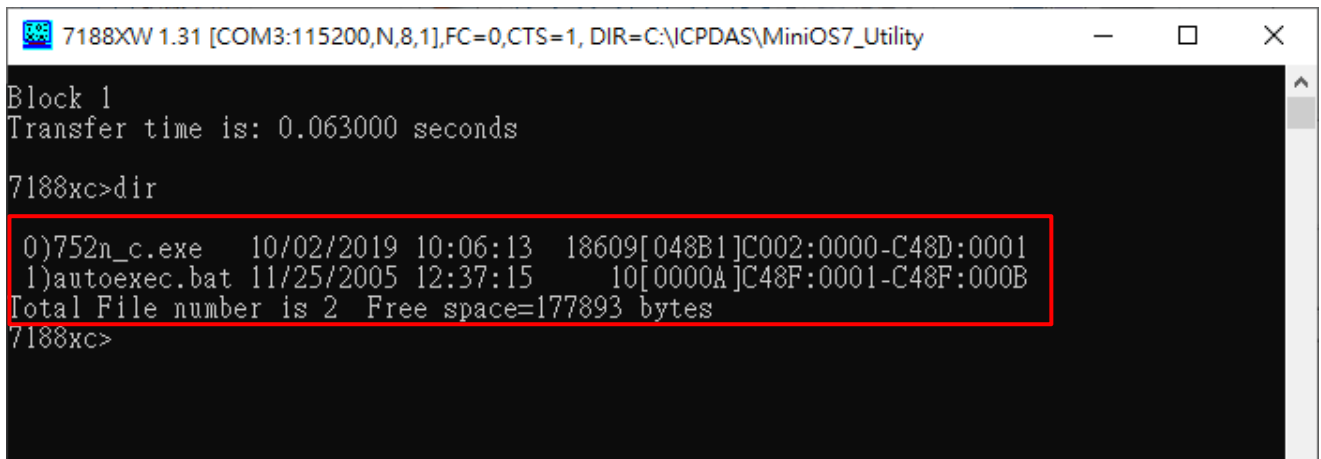
7188xc>LOAD
File will save to C000:0000
StartAddr-->B000:FFFF_
Press ALT_E to download file!
Load file:752n_c.exe[crc=550D,0000]
Send file info. total 73 blocks
Block 73
Transfer time is: 4.859000 seconds

7188xc>LOAD
File will save to C48D:0001
StartAddr-->C000:48D0
Press ALT_E to download file!
Load file:autoexec.bat[crc=4FC0,0000]
Send file info. total 1 blocks
Block 1
Transfer time is: 0.063000 seconds

7188xc>_
```

Update completed

Step6. Type “dir” at the prompt and press **Enter** again, you can see the date for updated firmware files.



```
7188XW 1.31 [COM3:115200,N,8,1],FC=0,CTS=1, DIR=C:\ICPDAS\MiniOS7_Utility
Block 1
Transfer time is: 0.063000 seconds
7188xc>dir
 0)752n_c.exe  10/02/2019 10:06:13  18609[048B1]C002:0000-C48D:0001
 1)autoexec.bat 11/25/2005 12:37:15   10[0000A]C48F:0001-C48F:000B
Total File number is 2  Free space=177893 bytes
7188xc>
```

Step7. Remove the lead wire between INIT* pin and GND pin, and then power cycle the module to go back to normal mode.

5.4. Getting Communication Parameters of a Module with Unknown Settings

For I-752N series module with unknown communication parameters, you can refer to the steps in **section 3.2. Searching Module** to search you modules and set new parameters one by one. On the D-version module, the module address, baud rate and end character settings for every COM port are alternately shown on the 7-segment LED display.

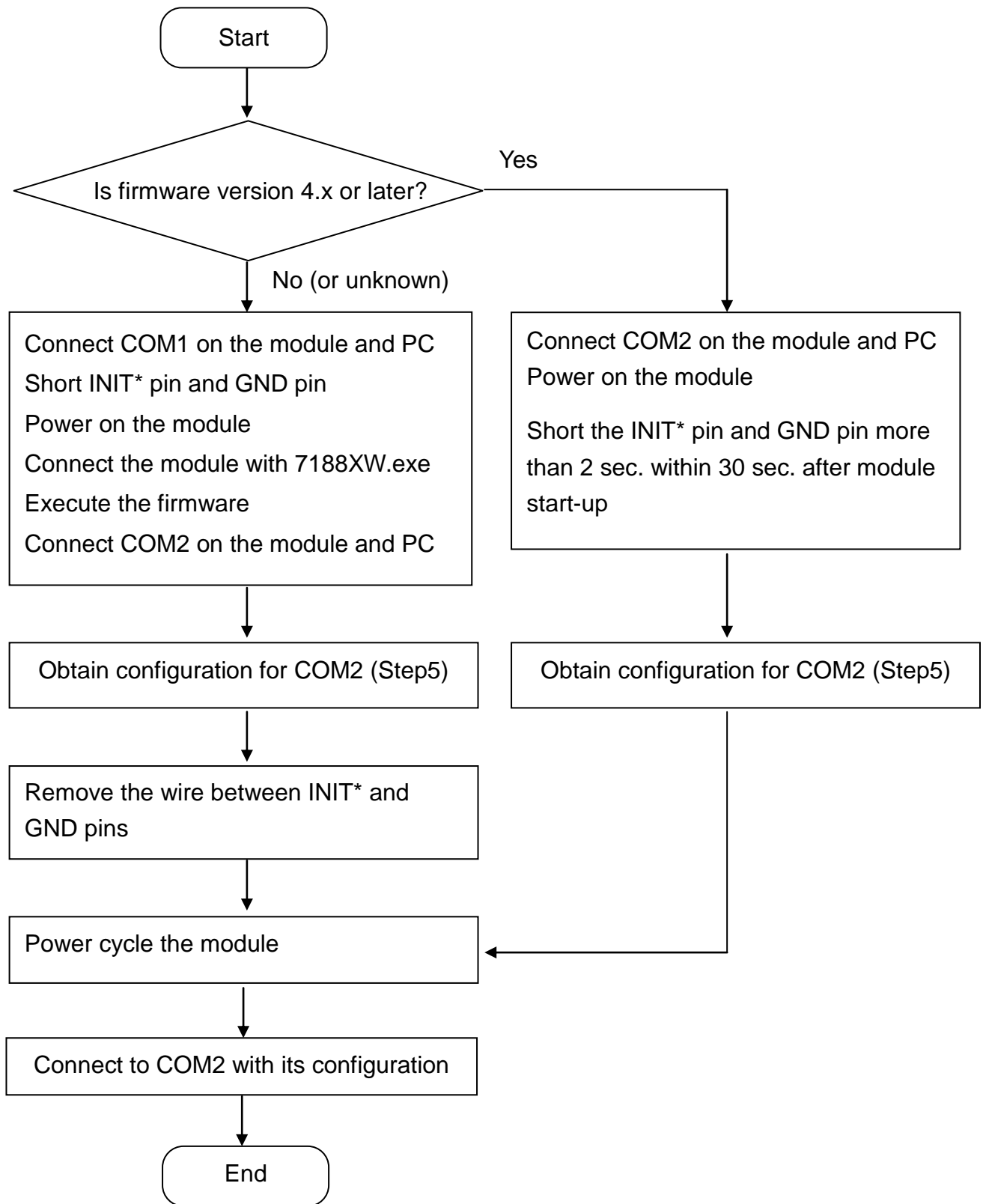
If you cannot find the module in DCON Utility Pro, running the firmware of I-752N series module in INIT mode enables you to communicate the module with specific parameters

Communication parameters for I-752N series module running firmware in INIT mode:

Module address	00
Baud Rate	9600
Data Format	N,8,1
Checksum	Disabled
CrLfmode	0 (0x0D)

With firmware version V4.x and later, shorting the INIT* pin and GND pin more than 2 seconds within 30 seconds after powering on the module enables the module to execute its firmware in INIT mode. At this point, the system LED indicator flashes about once per second. When the firmware is executed in INIT mode, the system indicator will flash at about twice per second, and the 7-segment display of D-version module will show COM port address from 00. It can be used to determine which mode the module is running in.

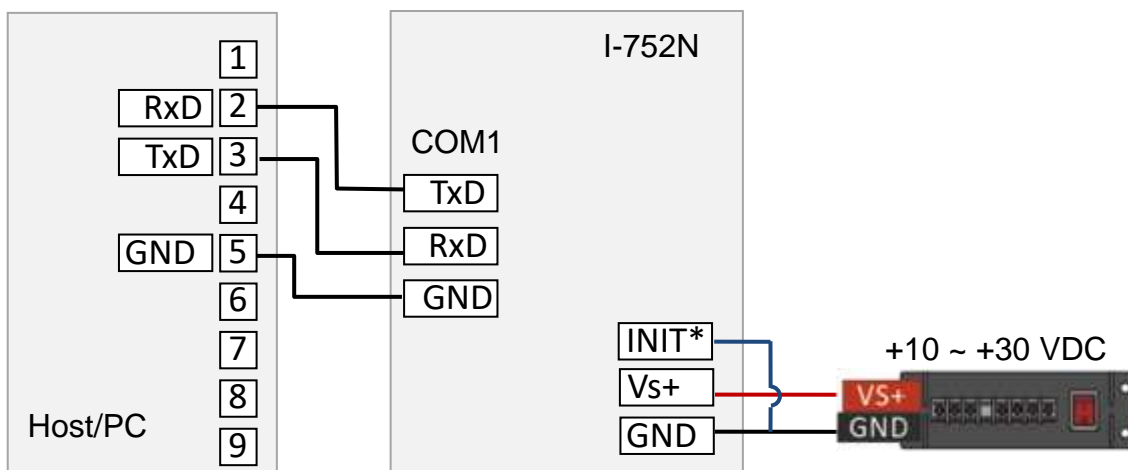
Flowchart for getting communication parameters of a module with unknown settings



When you short the INIT* pin and GND pin more than 2 seconds within 30 seconds after powering on the module to execute its firmware in INIT mode (with firmware version V4.x and later), **start from step4** to connect COM2 on the module.

With **firmware version V3.x and prior**, follow the steps below to connect to COM1 on the module, execute firmware in INIT mode, and read parameters for the module from COM2.

Step1. Connect COM1 on the module and PC as below, short INIT* pin and GND pin and power on the module.

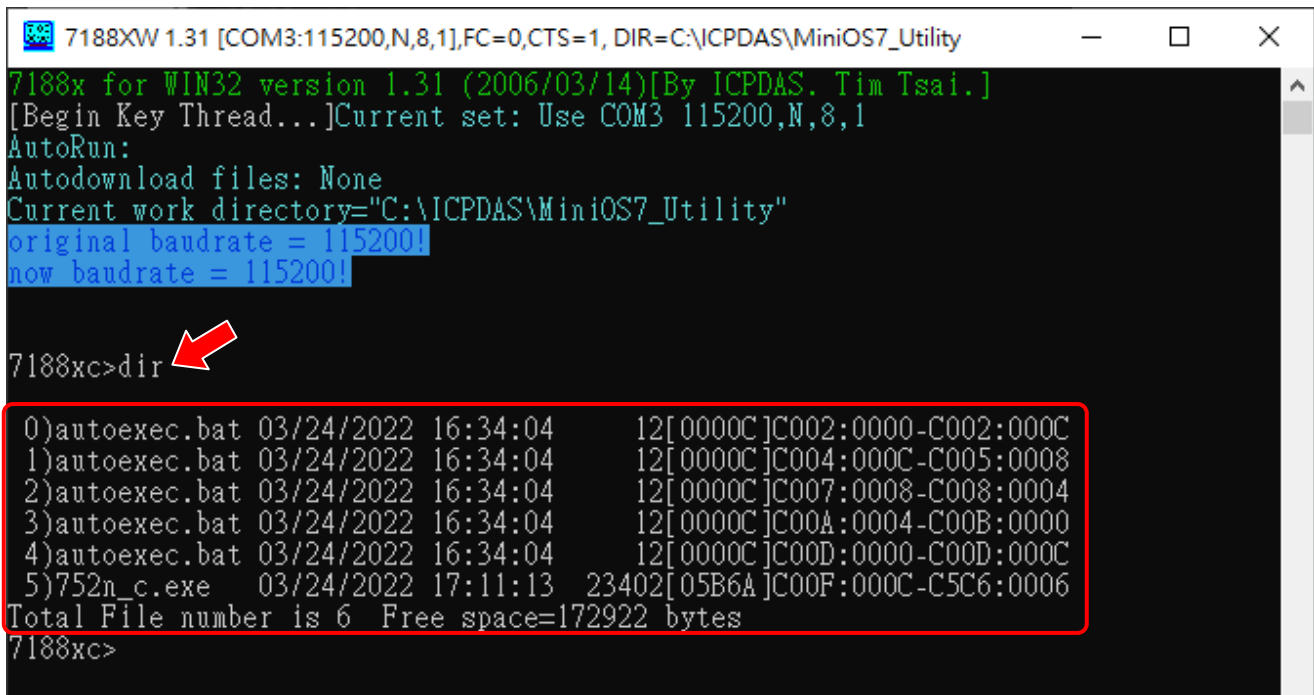


Step2. Set 7188XW.exe to communicate to COM1 on the module with baud rate **115200** bps and data format **8/N/1**. Press the Enter key to display the prompt “7188xc>” for I-7521/I-7522 /I-7523 or “7188xb>” for I-7522A/I-7524/I-7527.

```
7188XW 1.31 [COM3:115200,N,8,1],FC=0,CTS=1, DIR=C:\ICPDAS\MiniOS7_Utility
7188x for WIN32 version 1.31 (2006/03/14)[By ICPDAS. Tim Tsai.]
[Begin Key Thread...]Current set: Use COM3 115200,N,8,1
AutoRun:
Autodownload files: None
Current work directory="C:\ICPDAS\MiniOS7_Utility"
original baudrate = 115200!
now baudrate = 115200!

7188xc>
```

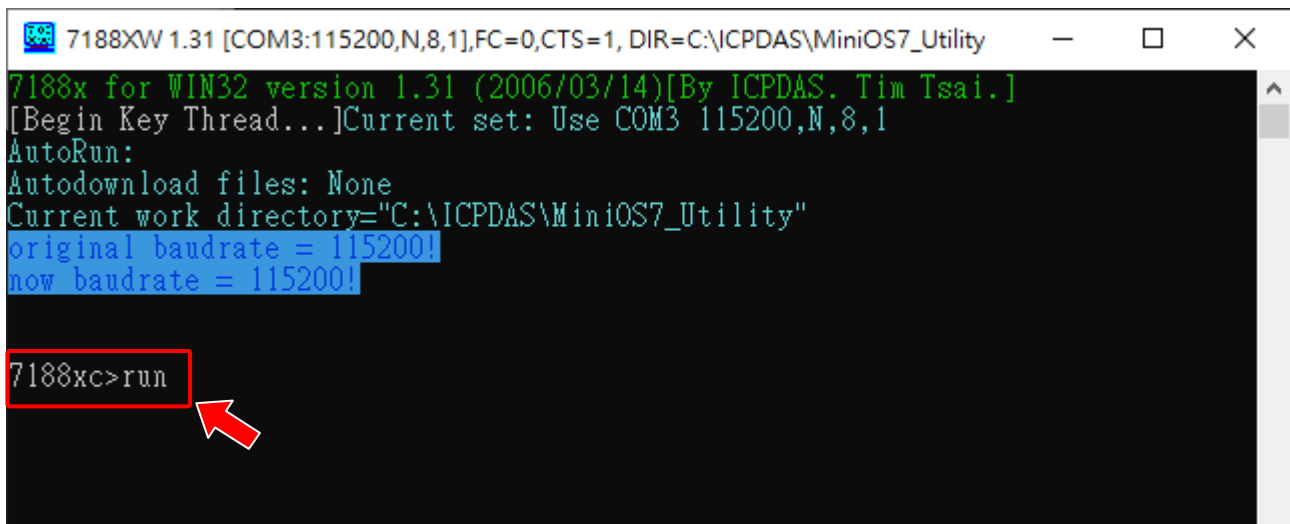
Type "dir" at the prompt and press **Enter** to list files in the module



```
7188XW 1.31 [COM3:115200,N,8,1],FC=0,CTS=1, DIR=C:\ICPDAS\MiniOS7_Utility
7188x for WIN32 version 1.31 (2006/03/14)[By ICPDAS. Tim Tsai.]
[Begin Key Thread...]Current set: Use COM3 115200,N,8,1
AutoRun:
Autodownload files: None
Current work directory="C:\ICPDAS\MiniOS7_Utility"
original baudrate = 115200!
now baudrate = 115200!

7188xc>dir
0)autoexec.bat 03/24/2022 16:34:04 12[0000C]C002:0000-C002:000C
1)autoexec.bat 03/24/2022 16:34:04 12[0000C]C004:000C-C005:0008
2)autoexec.bat 03/24/2022 16:34:04 12[0000C]C007:0008-C008:0004
3)autoexec.bat 03/24/2022 16:34:04 12[0000C]C00A:0004-C00B:0000
4)autoexec.bat 03/24/2022 16:34:04 12[0000C]C00D:0000-C00D:000C
5)752n_c.exe 03/24/2022 17:11:13 23402[05B6A]C00F:000C-C5C6:0006
Total File number is 6 Free space=172922 bytes
7188xc>
```

Step3. Type "run" and press **Enter** to execute the firmware.

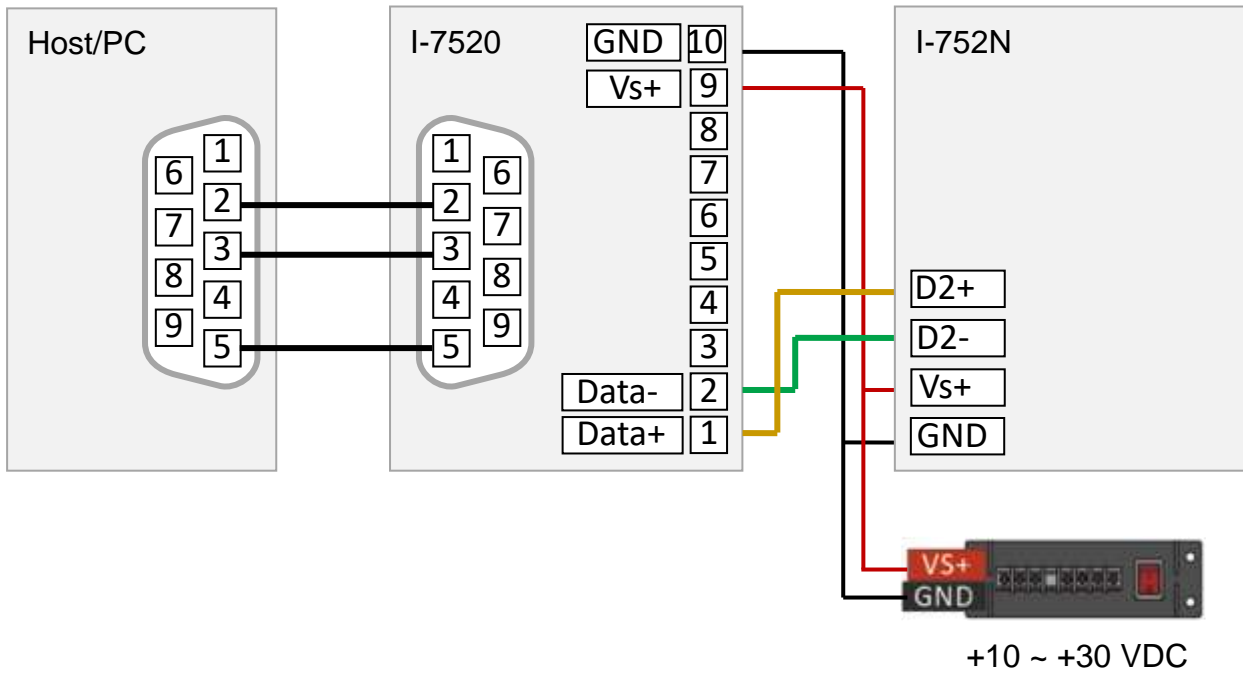


```
7188XW 1.31 [COM3:115200,N,8,1],FC=0,CTS=1, DIR=C:\ICPDAS\MiniOS7_Utility
7188x for WIN32 version 1.31 (2006/03/14)[By ICPDAS. Tim Tsai.]
[Begin Key Thread...]Current set: Use COM3 115200,N,8,1
AutoRun:
Autodownload files: None
Current work directory="C:\ICPDAS\MiniOS7_Utility"
original baudrate = 115200!
now baudrate = 115200!

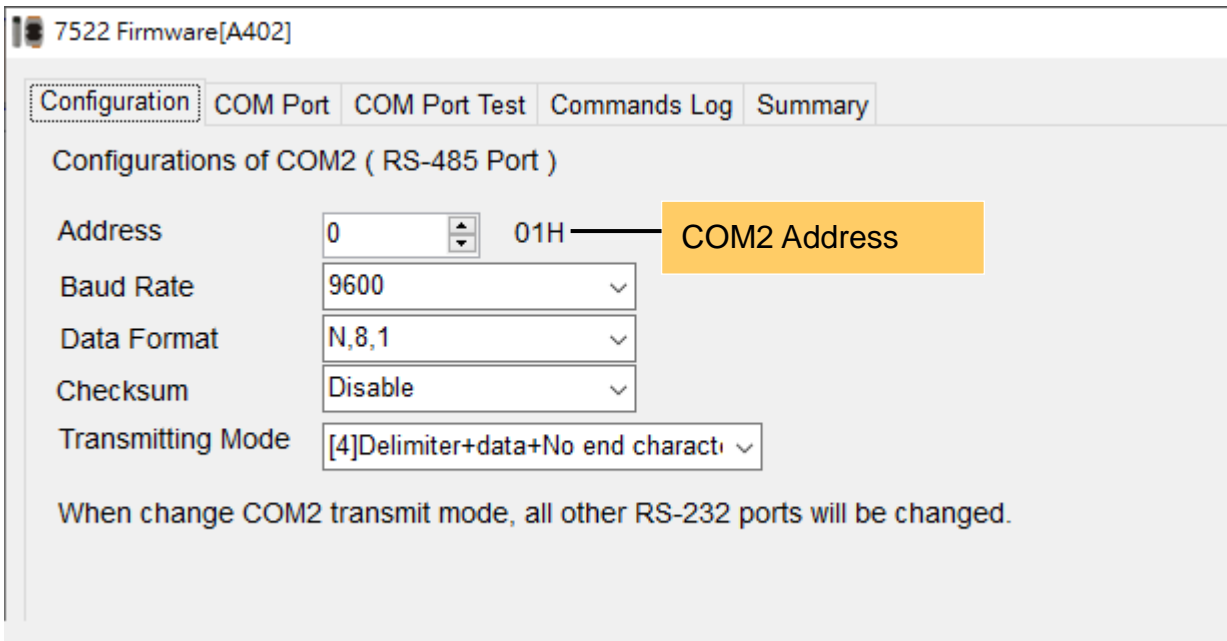
7188xc>run
```

Step4. Change to connect the COM2 on the module and PC.

(If the INIT* pin is shorted to the GND pin, don't remove the wire.)



Step5. Search the module in DCON Utility Pro with baud rate 9600 bps, the module will be search at address 00. Note down the configuration for communicating to COM2 in normal mode.



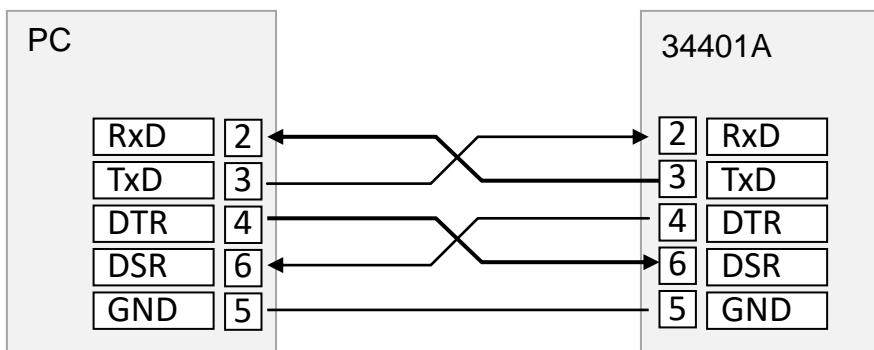
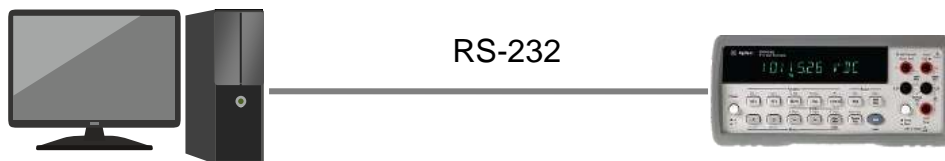
Step6. Power off the module, remove the lead wire between INIT* pin and GND pin if used.

Step7. Power on the module again, and communicate to the module through COM2 with the configuration obtained in step5.

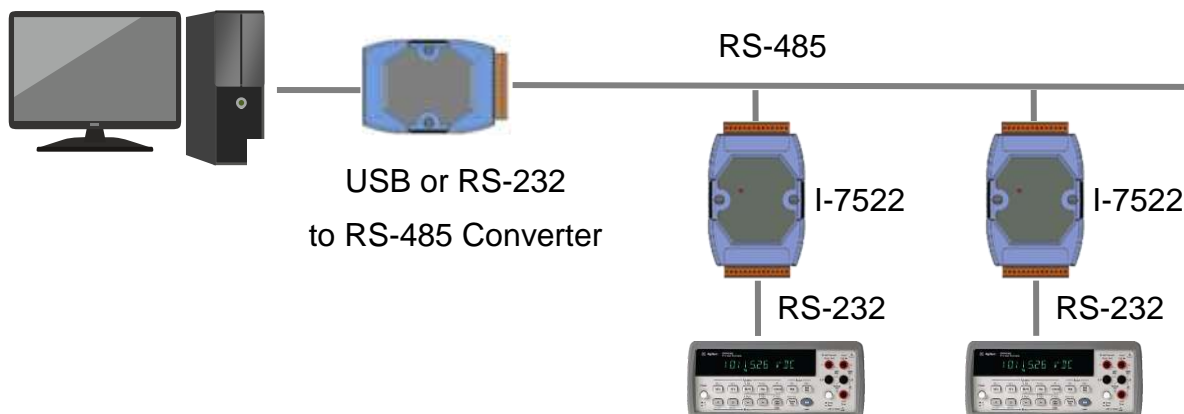
6. Applications

6.1. Connecting I-7522 to One Agilent 34401A

The wiring between Agilent 34401A and PC is usually using a crossover cable or a null-modem adapter.

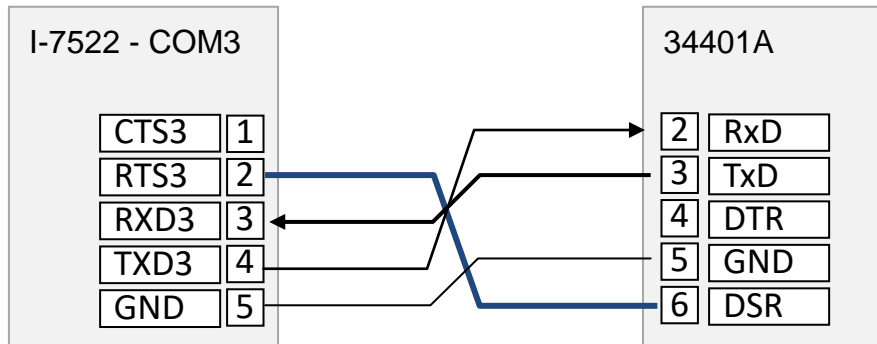


RS-232 only allows one-to-one communication, while the RS-485 interface allows up to 128 transceivers to be connected on the bus. Through the I-752N series module, host PC can use one USB or RS-232 port to communicate to plenty of RS-232 devices by connecting them to a RS-485 network and addressing for each.



Make sure that the interface for 34401A is set to RS-232, as well as baud rate and parity settings. The wiring diagram for connecting the 34401A and the I-7522 is shown as below. Note that the DSR line of 34401A is connected to RTS line of COM3 on I-7522 module.

Here we take baud rate 9600 bps and data format 8/N/1 applied for 34401A as an example.



1. Connect COM2 to PC, and COM3 to 34401A
2. Set baud rate 9600 bps and data format N/8/1 for COM2 and COM3 of I-7522.
3. CrLfmode (end character) for COM2 can be 2- 0x0A (Lf) or 4 – no end character. While only 2 - 0x0A (Lf) can be set for COM3, because the end character in protocol for 34401A is 0x0A (Lf),
4. Launch 7188xw.exe, press ALT + L to enter Line Mode.
5. Type “:02*idn?” and press Enter to bypass “*idn?” command to the 34401A connected to COM3. The response from 34401A “HEWLETT-PACKARD,34401A,0,11-5-3” will be displayed on the screen.
6. Type “:01syst:rem” and press Enter to enable remote control of 34401A.
7. Type “:01read?” and press Enter to read measurement value.

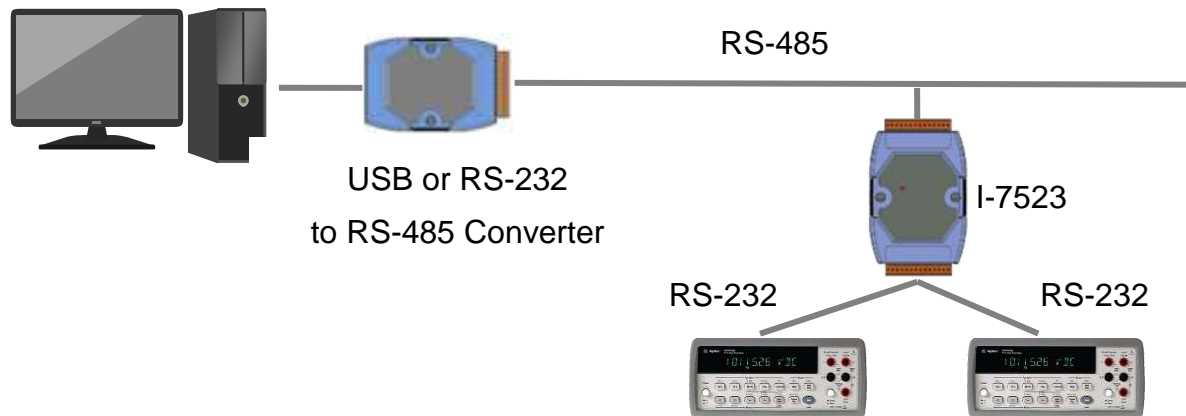
```

7188XW 1.31 [COM3:9600,N,8,1],FC=0,CTS=1, DIR=C:\ICPDAS\MiniOS7_Utility
original baudrate = 9600!
now baudrate = 9600!
{change to Line Mode}
$01T0
!012
$02T1
!022
:02*idn?
HEWLETT-PACKARD,34401A,0,11-5-3
:01syst:rem
:01read?
+6.74000000E-05

```


6.2. Connecting I-7523 to Two Agilent 34401A

The wiring to connect the two Agilent 34401A to COM3 and COM4 of I-7523 is similar to description in section 6.1.. Because COM 4 is a 3-wire RS-232 without RTS line, the DSR lines of the two 34401A are connected to RST line of COM3 on I-7523.



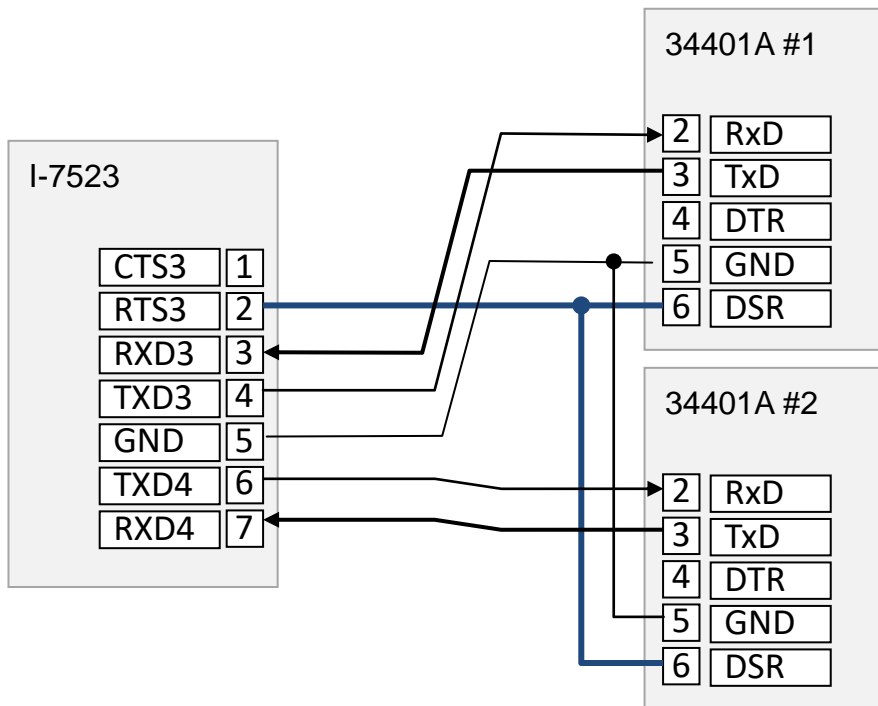
✧ Settings for 34401A

	34401A #1	34401A #2
Connected port on I-7523	COM3	COM4
Interface	RS-232	RS-232
Baud rate	9600	9600
Data format	N81	N81
End character	0x0A (LF)	0x0A (LF)

✧ Settings for I-7523

	COM2	COM3	COM4
Address	01	02	03
Baud rate	9600	9600	9600
Data format	N81	N81	N81
End character	4 (No end character)	2 (Lf)	2 (Lf)

✧ Wiring Diagram

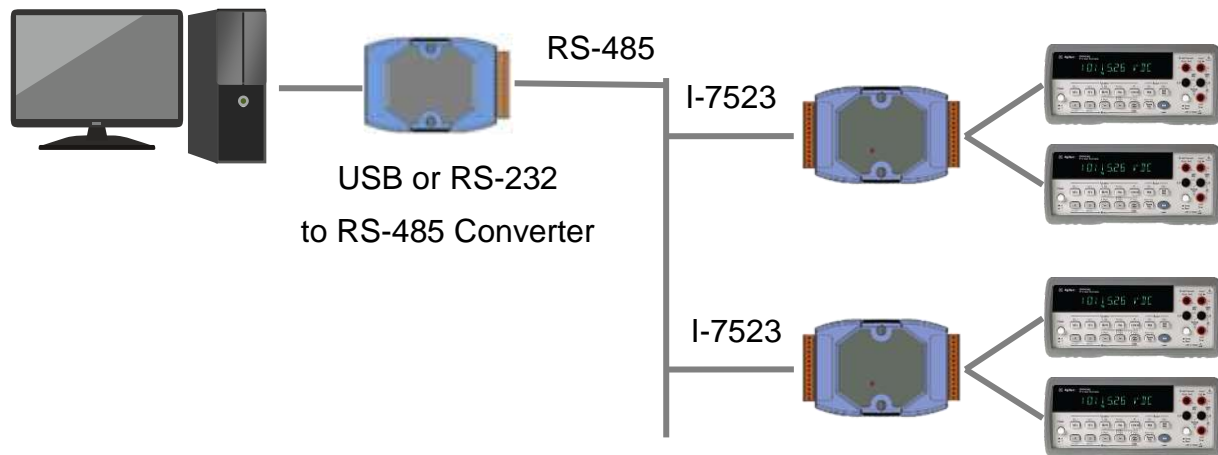


✧ Testing

1. Launch 7188xw.exe, press ALT + L to enter Line Mode.
2. Send **“:02system:rem”**, **“:03system:rem”** to enable remote control of 34401A.
3. Send **“:02read?”**, **“:03read?”** to read measurement value

6.3. Connecting Four Agilent 34401A with Two I-7523

The wiring to connect four Agilent 34401A to 2 I-7523 modules is similar to description in section 6.2.. When two or more I-7523 modules are connected to a RS-485 network, address for every COM port needs be set to a unique number in a range of 1 to 255.



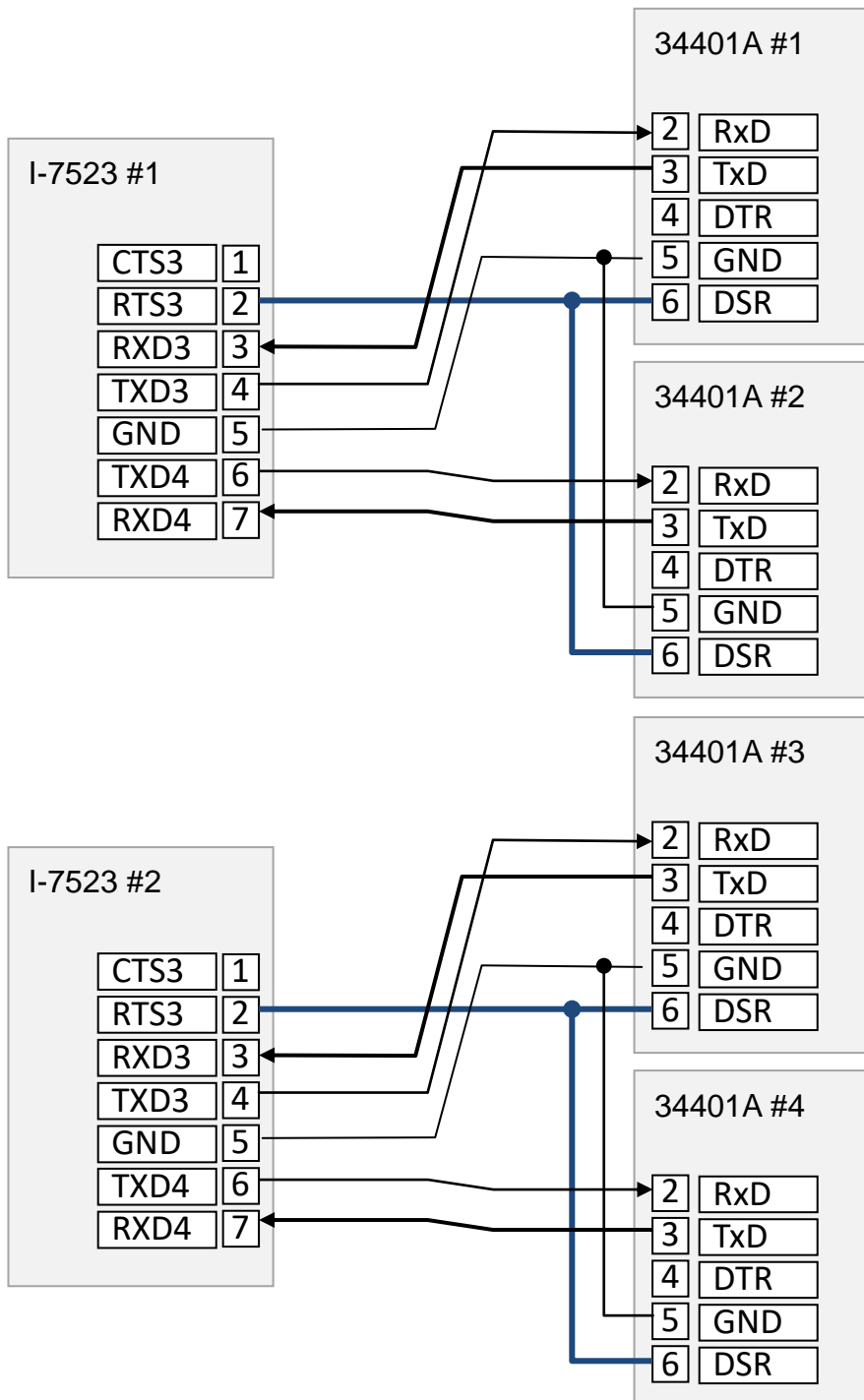
✧ Settings for 34401A

	34401A #1	34401A #2	34401A #3	34401A #4
Connected port on I-7523	COM3	COM4	COM3	COM4
Interface	RS-232	RS-232	RS-232	RS-232
Baud rate	9600	9600	9600	9600
Data format	N81	N81	N81	N81
End character	0x0A (LF)	0x0A (LF)	0x0A (LF)	0x0A (LF)

✧ Settings for I-7523

	I-7523 #1			I-7523 #2		
Module address	01			04		
COM address	COM2(01)	COM3(02)	COM4(03)	COM2(04)	COM3(05)	COM4(06)
Baud rate	9600	9600	9600	9600	9600	9600
Data format	N81	N81	N81	N81	N81	N81
End character	4 (Not use)	2 (Lf)	2 (Lf)	4 (Not use)	2 (Lf)	2 (Lf)

✧ Wiring Diagram

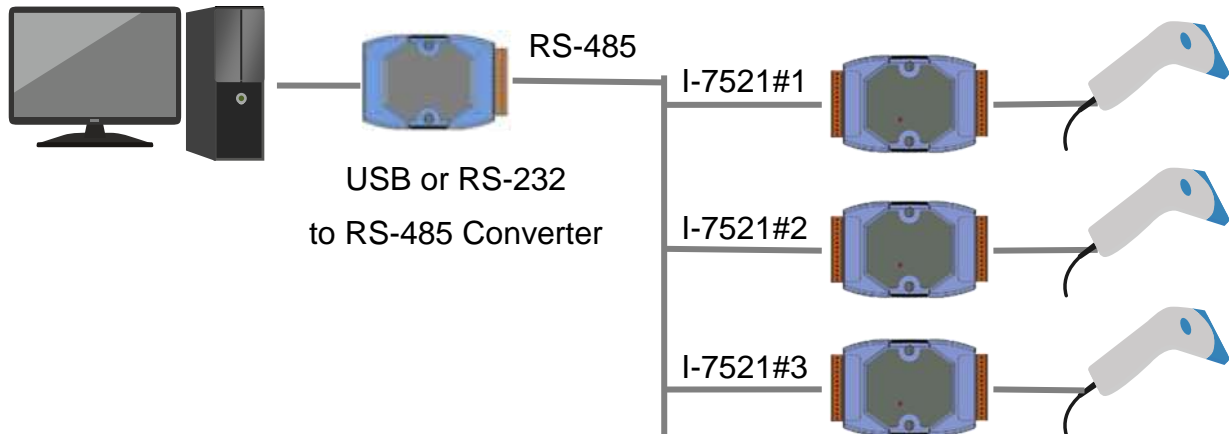


✧ Testing

1. Launch 7188xw.exe, press ALT + L to enter Line Mode.
2. Send `":02syst:rem"`, `":03syst:rem"`, `":05syst:rem"`, `":06syst:rem"` to enable remote control of 34401A.
3. Send `":02read?"`, `":03read?"`, `":05read?"`, `":06read?"` to read measurement value

6.4. Reading Data from Multiple Barcode Scanner

Sometimes, barcode scanners function in continuous mode in which the scanners transfer decoded data successively without request commands from host PC. The received records will be stored in the buffer temporarily, \$AAU/\$AAUR/\$AAUA command (section 7.4.24 ~ 7.4.26) is used to read records in COM port buffer. With using “\$AAEV” command (section 7.4.22) to enable the address prefix to each record, it is very easy to identify device from which a record is sent.



✧ Settings for barcode scanner

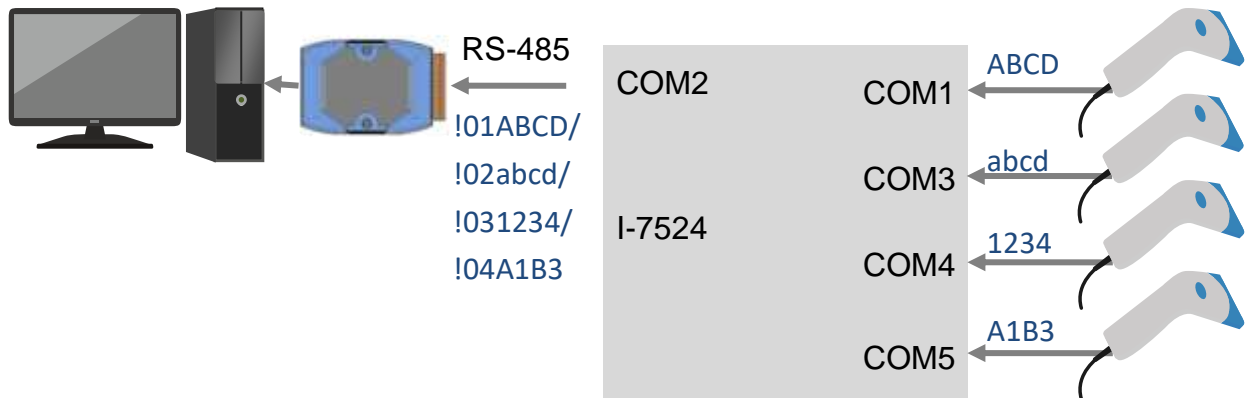
	#1	#2	#3
Connected port on I-7521	COM1	COM1	COM1
Baud rate	9600	9600	9600
Data format	N81	N81	N81
End character	0x0D (CR)	0x0D (CR)	0x0D (CR)

✧ Settings for I-7521

	I-7521 #1	I-7521 #2	I-7521 #3
Module address	01	02	03
COM1 address	01	02	03
Baud rate	9600	9600	9600
Data format	N81	N81	N81
End character	0 (CR)	0 (CR)	0 (CR)

6.5. Bypassing data from devices to COM2 (host PC)

The I-752N series can be set to bypass data from a series device to COM2 (host PC) when the device transfers data successively. The “\$AAHV” command (section 7.4.23) is used to enable the pypass function for a COM port. With using the “\$AAEV” command (section 7.4.22) to enable the address prefix for each message, it is easy to identify device from which a message is sent.



✧ Settings for barcode scanner

	#1	#2	#3	#4
Connected port	COM1	COM3	COM4	COM5
Baud rate	9600	9600	9600	9600
Data format	N81	N81	N81	N81
End character	0x0D (CR)	0x0D (CR)	0x0D (CR)	0x0D (CR)

✧ Settings for I-7524

	COM1	COM3	COM4	COM5
address	01	02	03	04
Baud rate	9600	9600	9600	9600
Data format	N81	N81	N81	N81
End character	0 (CR)	0 (CR)	0 (CR)	0 (CR)

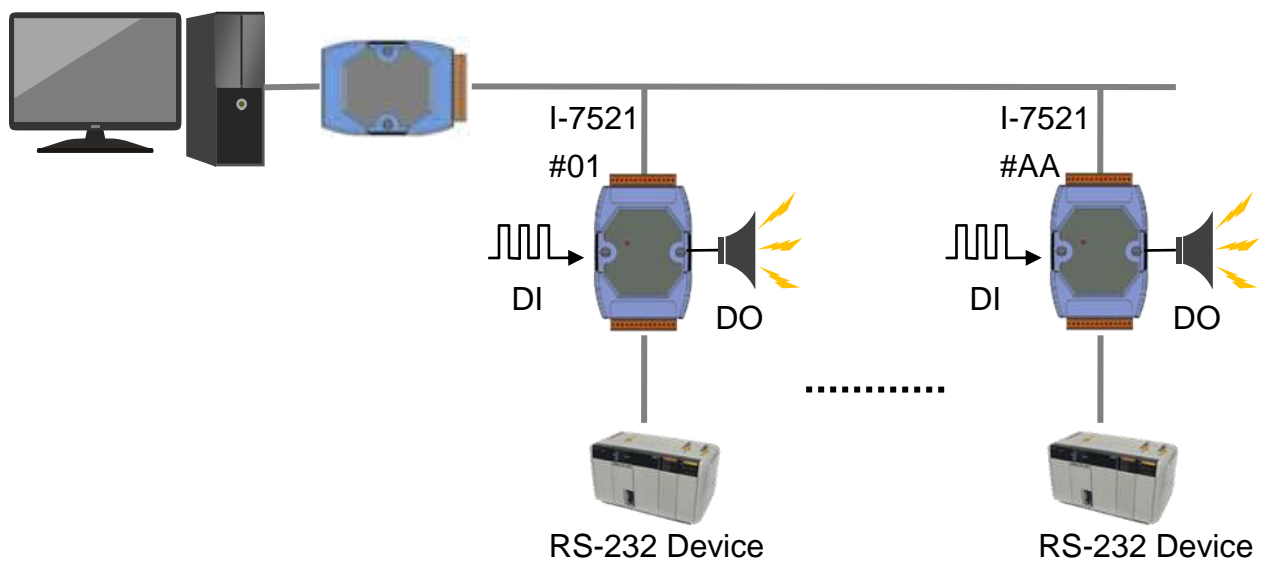


If the function is enabled, all the COM ports on a module needs be set to enable the function. Besides, only one I-752N series module can be used on a RS-485 network and PC can receive data from the RS-485 network only.

6.6. Real-time DI Monitoring and DO Control

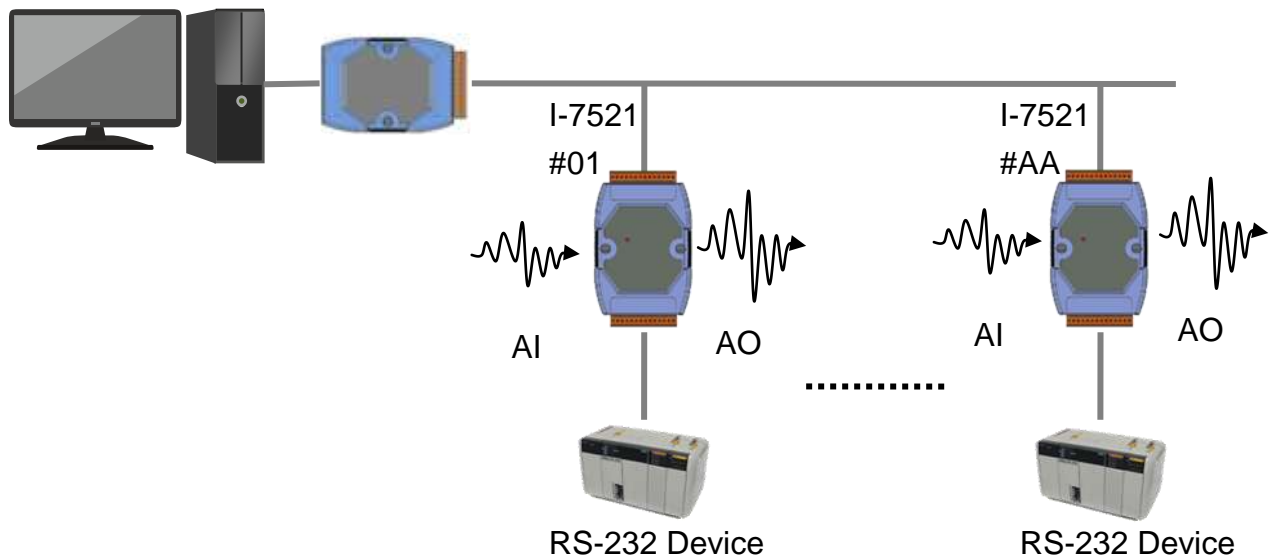
With a slight modification of firmware for the I-752N series module, the module can scan DI status and control DO lines in real time. For example, you can modify the firmware for your module to check the DI status for monitoring the status of emergency button or signal connected to DI channel, and control DO to turn off important device or turn on alarm light or sound if an emergency signal is detected. It can respond to emergency events more quickly without going through a computer.

(Refer to demo “7521odm1”)



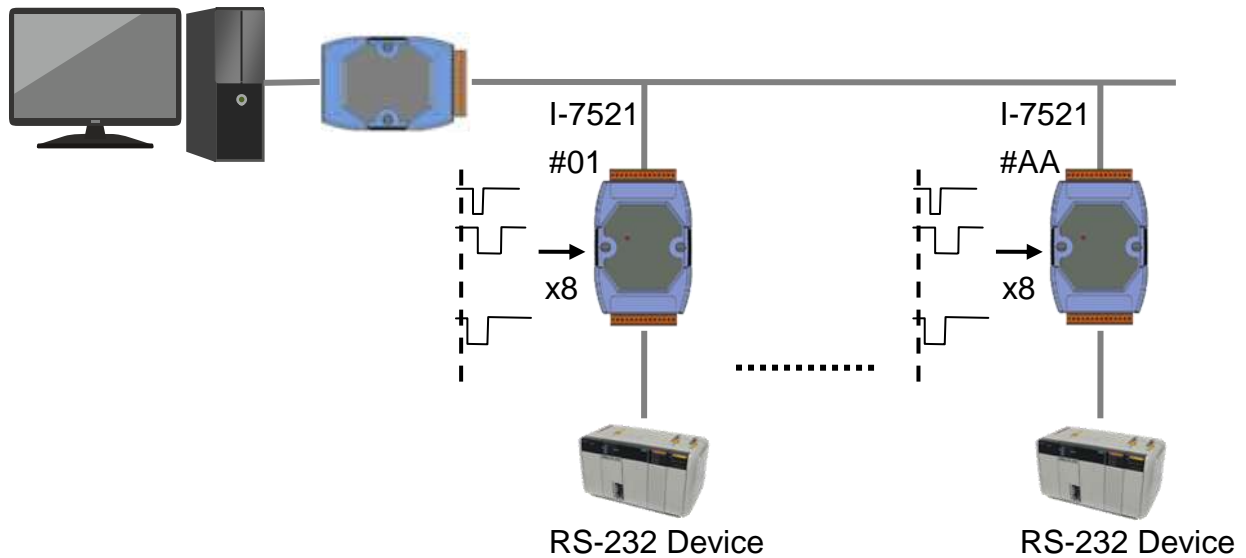
6.7. Real time A/D Monitoring and D/A Control

By installing X301 I/O expansion board into the I-7521 module, the firmware can be modified to monitor one A/D signal and control one D/A channel. Refer to demo program 7521odm2 for details of how to control AO according to AI signal. It can simplify the I/O configuration and reduce the workload of the host computer.



6.8. 8-channel Long Term Event Counters

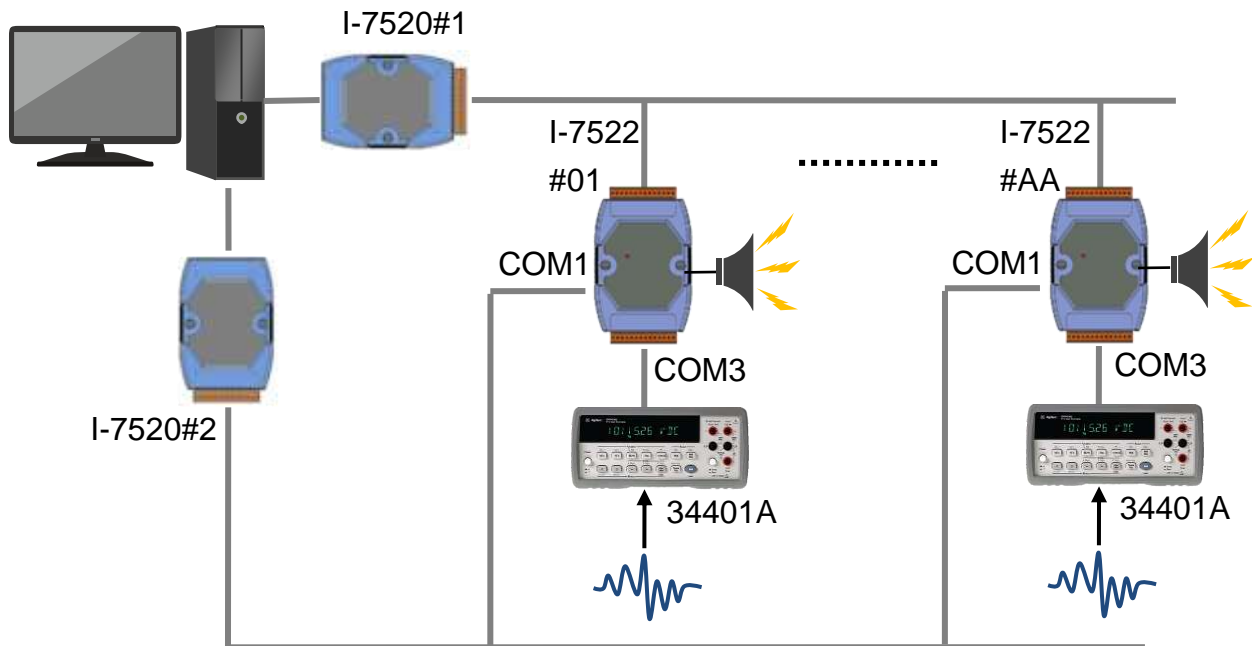
Installing X100 I/O expansion board into the I-7521 module enables the module to read 8 event counters in real time. The timing diagram of the event counters will be latched until a clear command is sent by host PC or the module restarts. Refer to the 7521ODM3.c file for the source code of the firmware.



6.9. Signal Monitor and Alarm Control

In this application, one COM port on host computer is connected to COM2 of I-7522 modules for sending commands and receiving data, another COM port is connected to COM1 of I-7522 modules and used as an emergency message network. All I-7522 modules will automatically monitor the analog signals on the connected HP34401A. When an emergency event occurs, the I-7522 module will send an emergency message to the second network. If there are multiple I-7522 modules send emergency messages at the same time, the I-7522 module will continue to send the message until it is received by the computer.

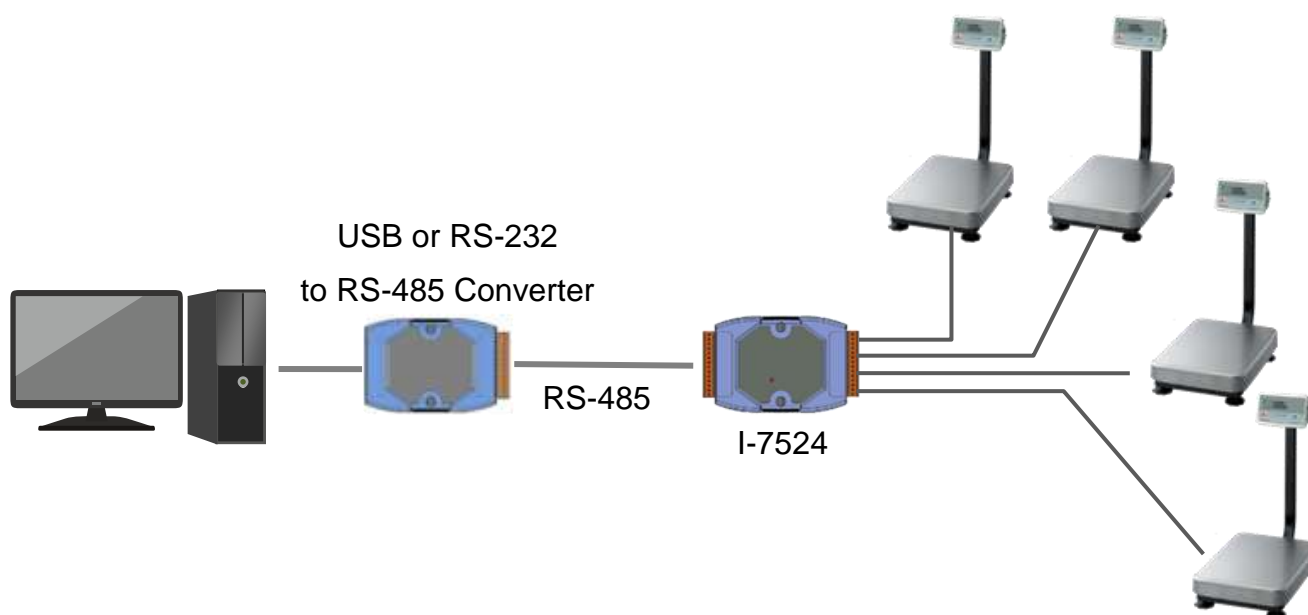
All procedures take place in the I-7522 module. The host computer only reads the analog values as part of its system monitoring function. Refer to the 7522ODM5.c file for the source code of the firmware.



6.10. Electronic Scale Application

The I-752N series module allows the computer host or PLC to connect multiple electronic scales (with RS-232 interface) through one RS-232 or USB port. Users can choose to connect multiple electronic scales to one I-752N series module, or use multiple I-7521 to connect these scales to an RS-485 network one to one. In addition, the DI channels on the I-752N series module can be used to confirm the status of emergency button or signal, as well as DO channels can be used to turn off important devices or turn on alarm devices.

In the following section, we will take FG-30KAM as an example to introduce how to apply the I-752N module to obtain measured data from electronic scales.



✧ Settings for FG-30KAM

	FG-30KAM #1	FG-30KAM #2	FG-30KAM #3	FG-30KAM#4
Baud rate	9600	9600	9600	9600
Data format	E71	E71	E71	E71
End character	0x0D0A (CrLf)	0x0D0A (CrLf)	0x0D0A (CrLf)	0x0D0A (CrLf)

✧ Settings for I-7524

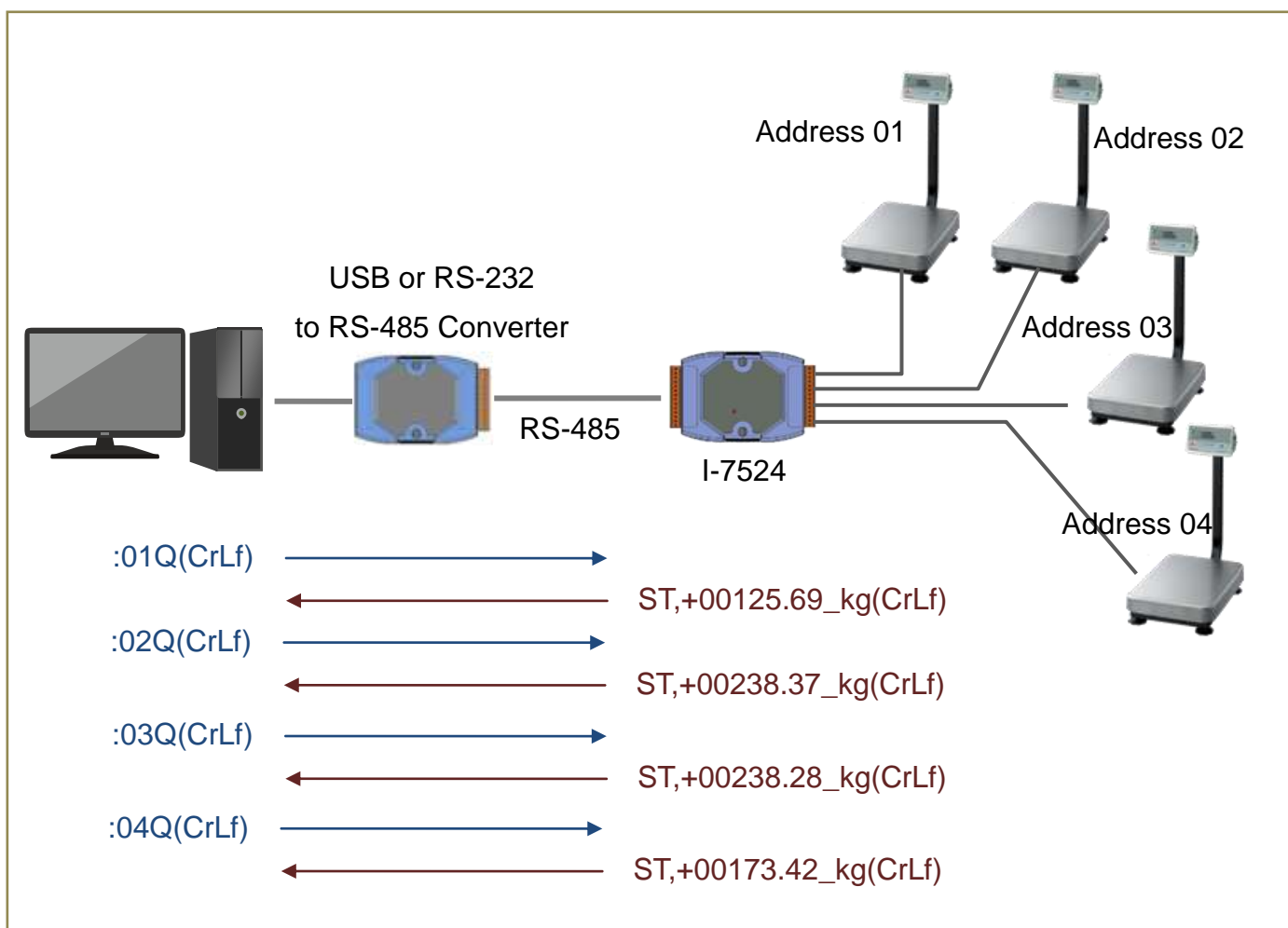
	I-7524				
Module Address	01				
COM (Address)	COM2(01)	COM1(01)	COM3(02)	COM4(03)	COM5(04)
Baud rate	9600	9600	9600	9600	9600
Data format	E71	E71	E71	E71	E71
End character	1 (CrLf)	1 (CrLf)	1 (CrLf)	1 (CrLf)	1 (CrLf)

According to the requirements of the system, FG-30KAM can be set to actively output the measured data, or response the data to the command from host computer. Make sure that the FG-30KAM is correctly configured to meet your application.

Mode1. Reading data from scales one by one

The data output mode of FG-30KAM is set to command mode each. It sets the scale to be controlled by commands that come from the host computer and so on.

Sending “:AAQ(CrLf)” command (section 7.4.19) to the I-752N series module can bypass “Q” command to a FG-30KAM, where AA specifies the address for a COM port to which the scale is connected, as well as (CrLf) is the end character for command and response strings of FG-30KAM. The host computer sends “:AAQ(CrLf)” commands to read the measured values in sequence.

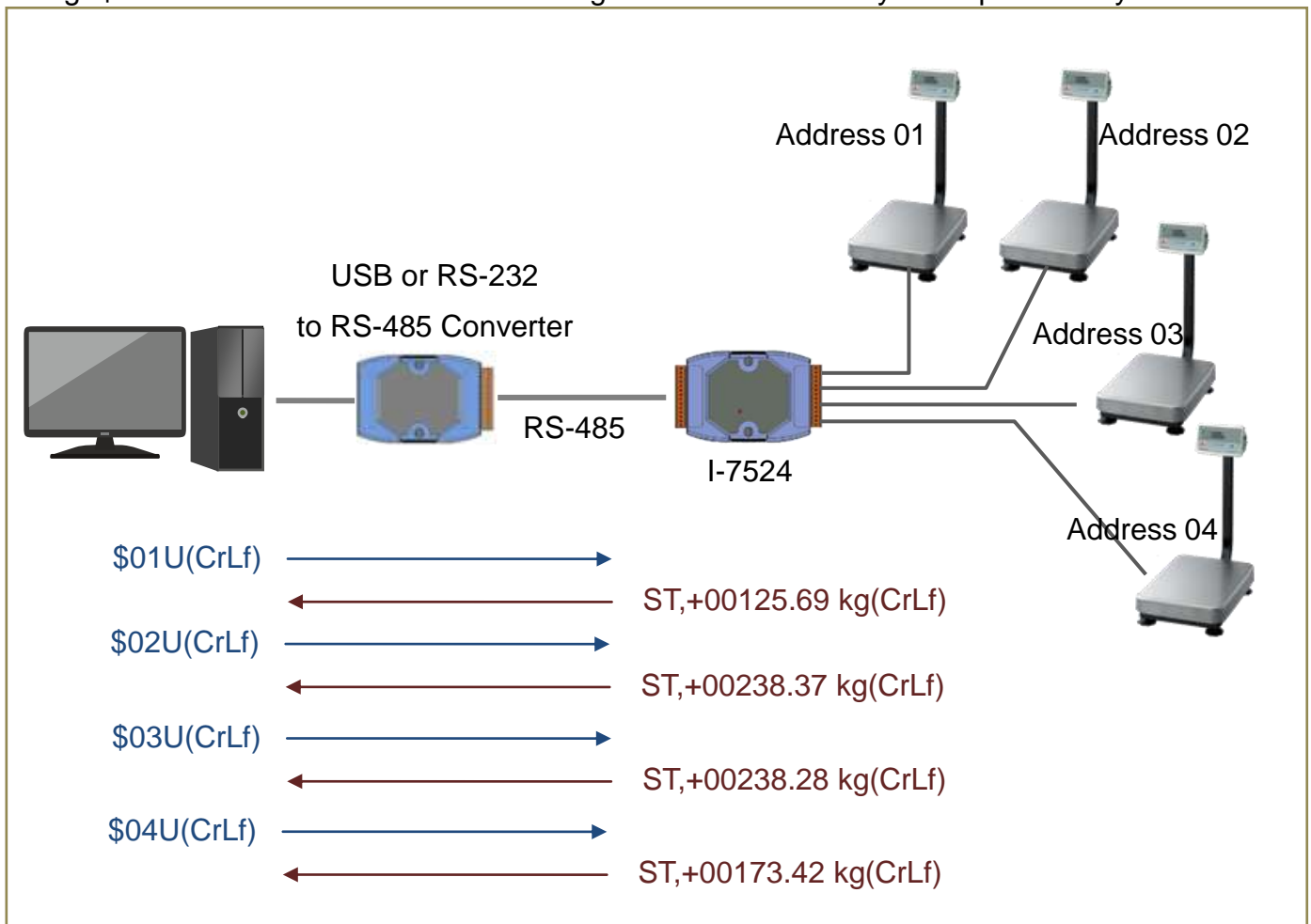


Mode2. Reading data from COM port buffer

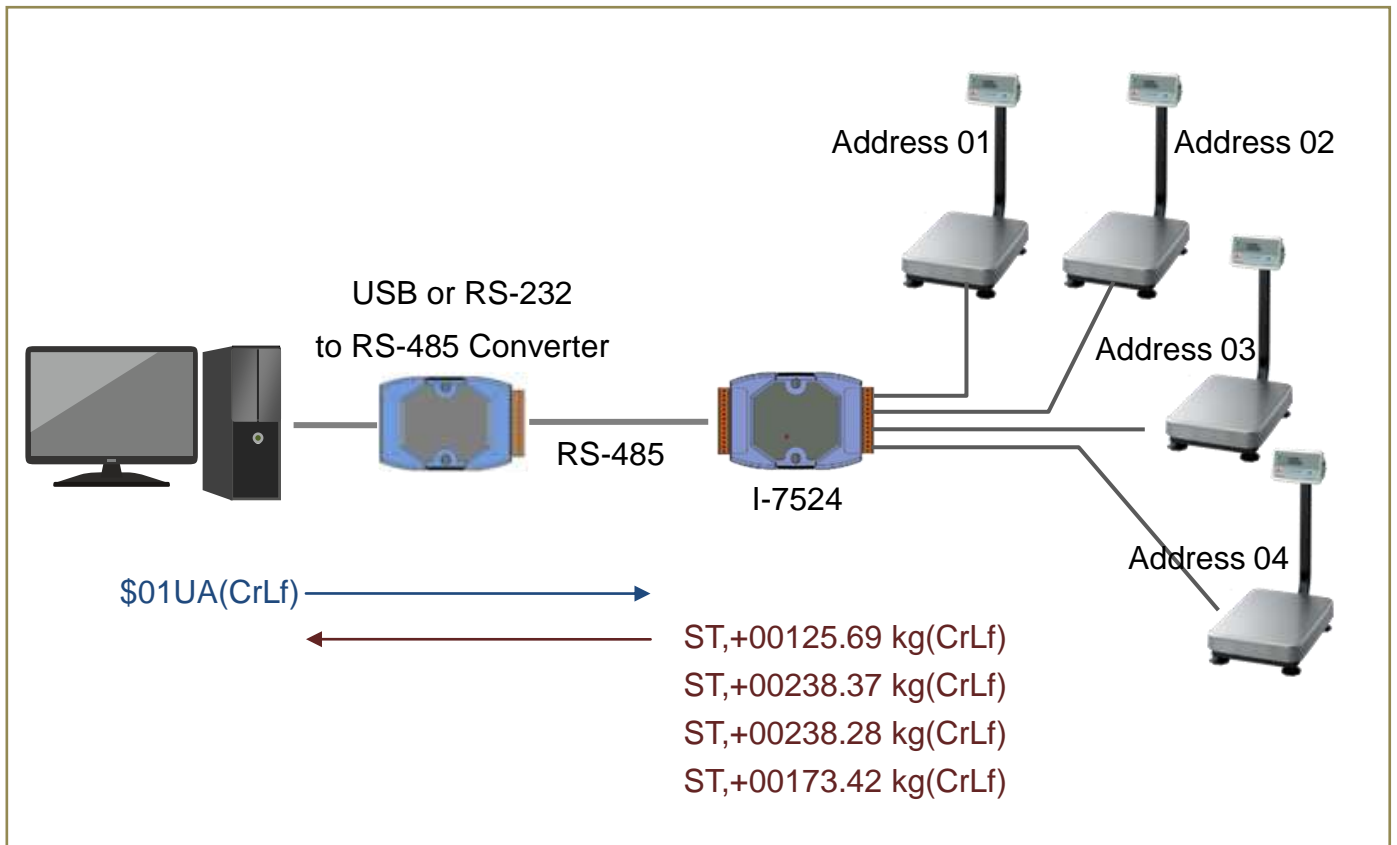
The data output mode of FG-30KAM is set to stream mode each. It sets the scales to output measured values at about 10 times per second. The data will be stored temporarily in the COM port buffer.

“\$AAU” command (section 7.4.24) and “\$AAUR” (section 7.4.25) can be used to read data from a COM port buffer, while “\$AAUA” (section 7.4.26) can read one data from buffer of every COM port on the module.

Using “\$AAU” command to read one message from buffer of every COM port one by one



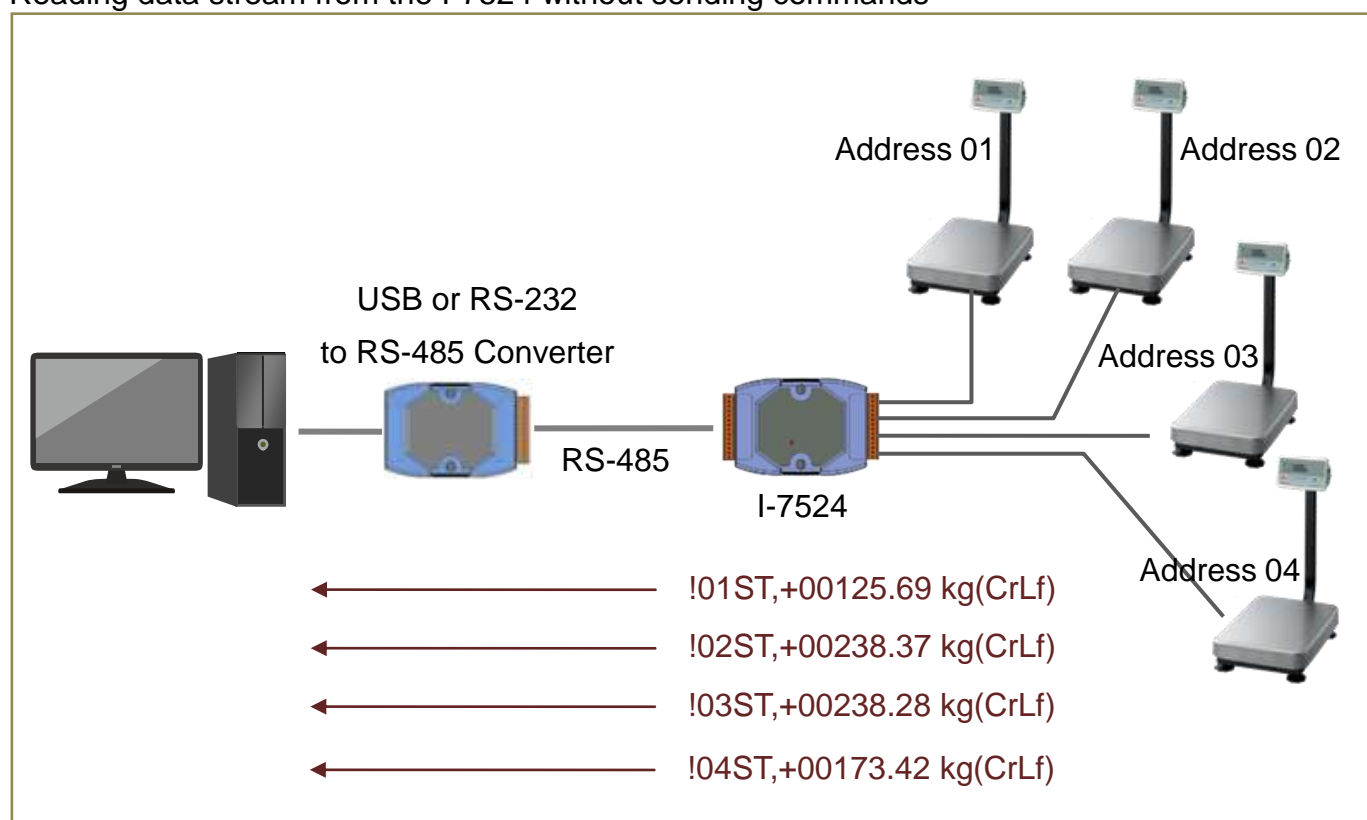
Using "\$AAUA" command to read one message from buffer if every COM port



Mode3. Bypassing Data from FG-30KAM to Host Computer

The FG-30KAM scale outputs the current display data with an update rate of approximately 10 times per second when it functions in stream mode. By using “\$AAHV” command (section 7.4.23) to set the I-752N series module to bypass data from FG-30KAM to host computer, and “\$AAEV” command (section 7.4.22) to enable COM port address prefix for every messages, the host computer can easy to receive data without sending commands, and identify which device a message is transmitted from according to its COM port address prefix.

Reading data stream from the I-7524 without sending commands

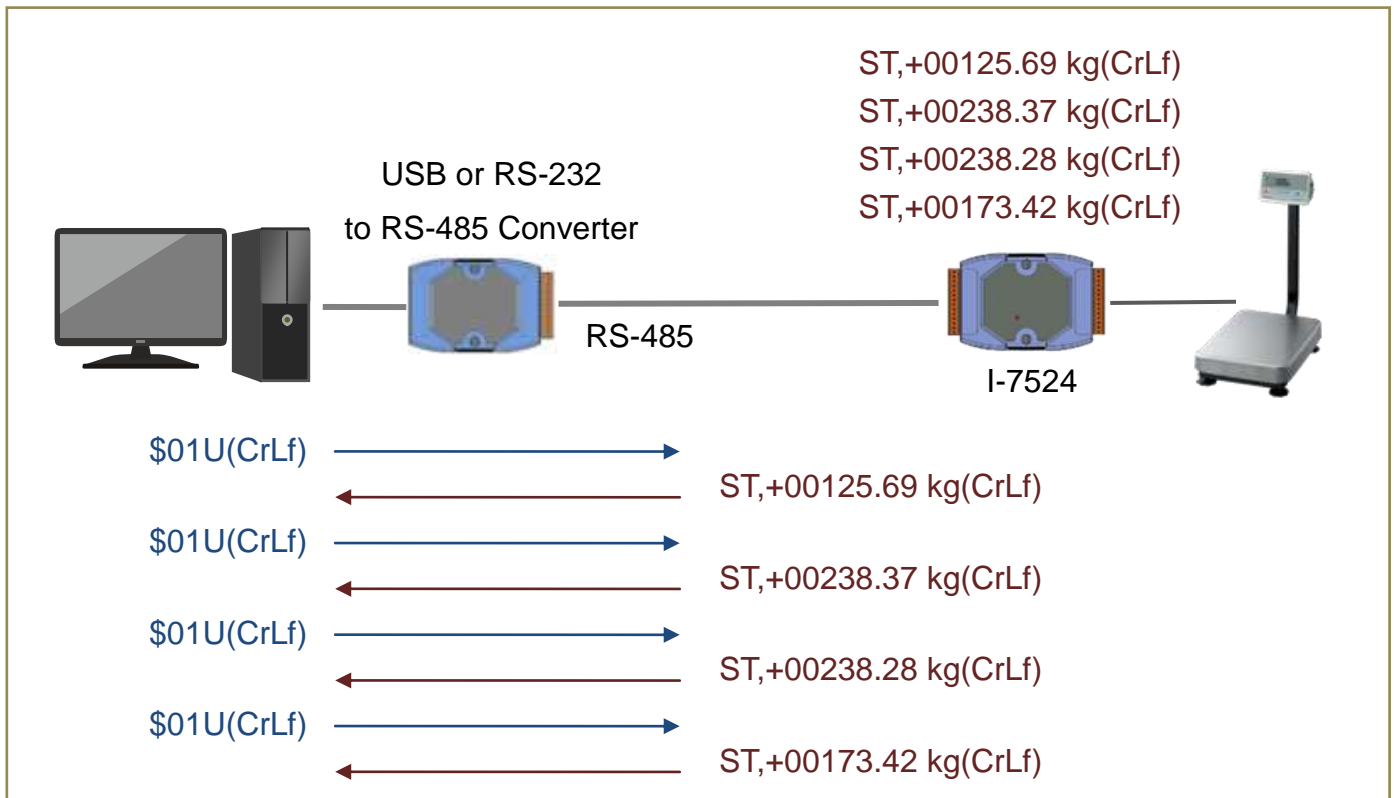


If the function is enabled, all the COM ports on a module needs be set to enable the function. In addition, only one I-752N series module can be used on a RS-485 network and PC can receive data from the RS-485 network only.

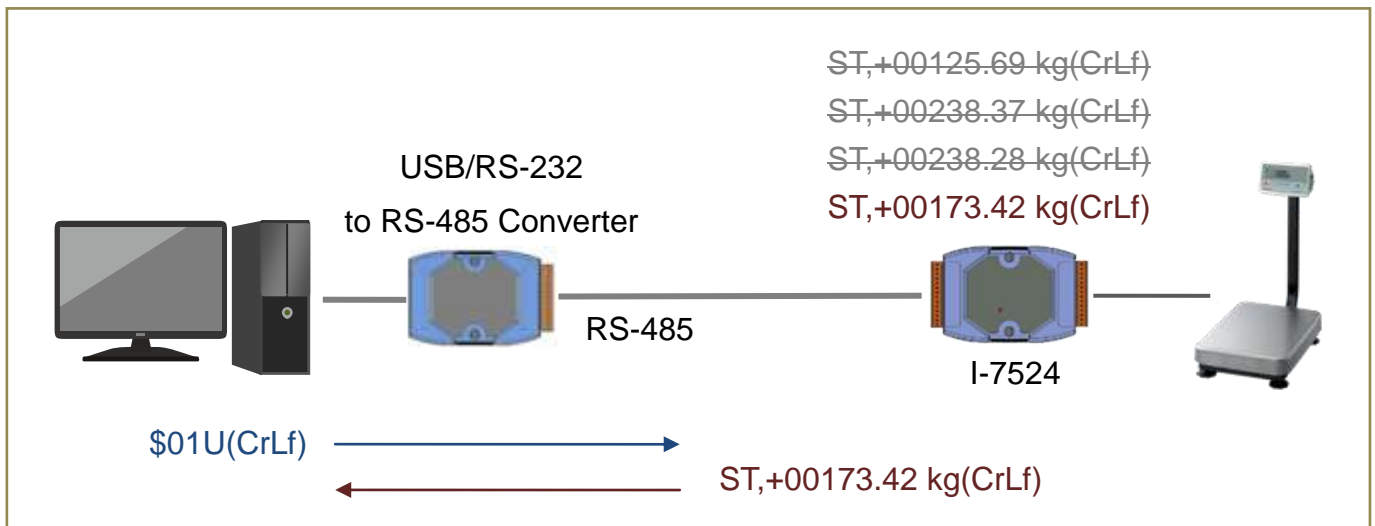
Mode4: Reading the latest record from buffer

The major mode of the COM port buffer is to fill up data received from the connected device until the buffer is full. If the reading speed is less than the speed of filling up data, host PC will always read previous data stored in buffer. \$AAN[buffermode] command (section 7.4.32) can be used to set a buffer to keep the latest data only, then the computer can get the latest record every time.

Keeping all received records except the read-out ones (default)



Keeping the latest record only



7. Command Sets

7.1. Command Format

The commands for the I-752N series module usually consist of a delimiter character, a module (or COM Port) address, one or two characters as command code, setting values and ending character(s). Take the command \$AAA[addr][chk](CrLf) as an example:

- \$ is the delimiter character.
- The first two AA characters are module or COM Port address, ranging from 00 to FF in hexadecimal format, equivalent to 0 to 255 in decimal.
- The third A is the command code.
- [addr] is the new address (setting value) in the command.
- [chk] is 2 characters of checksum, it needs to add the checksum result if the checksum function is enabled.
- (CrLf): end character is the same as the CrLfmode set for COM2.

7.2. COM Port Address

RS-485 allows multiple devices to communicate at half-duplex on a single pair of wires. A combination of one RS-232 port and COM2 is regarded as an I-7521, using a unique address. Therefore, the combination of COM1 and COM2 forms the first I-7521, uses the module address. The second RS-232 port (COM3) and COM 2 form the second I-7521, use the address incremented by adding 1 to the previous COM port, so on and so forth. Assuming that the module address of your I-752N series module is AA, the COM Port address is defined as follows:

	I-7521	I-7522	I-7523	I-7522A	I-7524	I-7527
COM1(RS-232)	AA	AA	AA	AA	AA	AA
COM2(RS-485)	AA	AA	AA	AA	AA	AA
COM3 (RS-232)/(RS-422)	-	AA+1	AA+1	AA+1	AA+1	AA+1
COM4(RS-232)	-	-	AA+2	-	AA+2	AA+2
COM5(RS-232)	-	-	-	-	AA+3	AA+3
COM6(RS-232)	-	-	-	-	-	AA+4
COM7(RS-232)	-	-	-	-	-	AA+5
COM8(RS-232)	-	-	-	-	-	AA+6

7.3. Command Set Table

Communication Parameters

Command	Response	Description	Section
\$AAA[addr]	!AA	Reads/sets the module address	7.4.1
\$AABN[baud-rete]	!AA[baud-rete]	Reads/sets the baud rate for COM-1/2/3/4/5/6/7/8	7.4.2
\$AADN[data-bit]	!AA[data-bit]	Reads/sets the data bit for COM-1/2/3/4/5/6/7/8	7.4.3
\$AAPN[parity-bit]	!AA[parity-bit]	Reads/sets the parity for COM-1/2/3/4/5/6/7/8	7.4.4
\$AAON[stop-bit]	!AA[stop-bit]	Reads/sets the stop bit for COM-1/2/3/4/5/6/7/8	7.4.5
\$AA6[ID]	!AA	Sets the ID-string for COM-1/3/4/5/6/7/8	7.4.6
\$AA7	!AA[ID]	Reads the ID-string for COM-1/3/4/5/6/7/8	7.4.7
\$AAKV	!AA[checksum]	Reads/sets the checksum status of COM2 (RS485)	7.4.8
\$AATN[CrLfmode]	!AA[CrLfmode]	Reads/sets the end character used to judge the end of a command/ response for COM1/2/3/4/5/6/7/8	7.4.9
\$AAW	!AA(status)	Reads the CTS_status of COM-1/3/4/5	7.4.10
\$AAXV	!AA	Sets the RTS_state of COM-1/3/4/5	7.4.11
\$AA2	!AA40BDPK	Reads the configuration of COM2 (RS485)	7.4.12
\$AAIV	!AA	Restores COM ports configuration to factory settings. (Supported on firmware version 3.08 and later)	7.4.13

Module Information

Command	Response	Description	Section
\$AA5	!AAS	Reads the reset status	7.4.14
\$AAF	!AA[number]	Reads the firmware version number	7.4.15
\$AAM	!AA[name]	Reads the model number	7.4.16

Data Transfer

Command	Response	Description	Section
\$AAC[delimiter]	!AA[delimiter]	Reads/sets the delimiter for command to bypass data to COM-1/3/4/5/6/7/8	7.4.17
\$AAD	!AA[delimiter]	Reads the delimiter for command to bypass data to COM-1/3/4/5/6/7/8	7.4.18
(delimiter)AA(bypass)	Depend on device	Bypasses the data string to COM-1/3/4/5/6/7/8	7.4.19
\$AAJN[timeout]	!AA[timeout]	Reads/sets the delay time to determine whether the end of a command/response.	7.4.20
\$AAGN[trigger-level]	!AA[trigger-level]	Reads/sets the trigger level for the RS-232 COM port buffer to transmit data,	7.4.21
\$AAEV	!AA(status)	Reads/sets the address prefix status for the response. (Supported on firmware version 3.05 and later)	7.4.22
\$AAHV	!AA(status)	Reads/sets the status for bypassing data string to COM2. (Supported on firmware version 3.08 and later)	7.4.23
\$AAU	[data]	Reads data from the RS-232 COM port buffer. No response if no data in buffer.	7.4.24
\$AAUR	[data]	Reads data from the RS-232 COM port buffer. No response if no data in	7.4.25

		buffer. "N/A" returned if no data in buffer. (Supported on firmware version 4 and later)	
\$AAUA	[data]	Reads one message from every RS-232 COM port buffer. (Supported on firmware version 4 and later)	7.4.26
\$AAUC	!AA	Clears data in every COM port buffer. (Supported on firmware version 4 and later)	7.4.27
\$AAUN	!AAn	Reads the message quantity in a COM port buffer. (Supported on firmware version 4 and later)	7.4.28
\$AAUL[minlength]	!AA[minlength]	Reads/sets the minimum length of response. (Supported on firmware version 4 and later)	7.4.29
\$AAUD[padchar]	!AA[padchar]	Reads/sets the character used to fill up a message, while its length is less than the minimum length set by command \$AAUL. (Supported on firmware version 4 and later)	7.4.30
\$AAUS[seperator]	!AA[seperator]	Reads/sets the seperator character for reading one message from the buffer of every COM port. (Supported on firmware version 4 and later)	7.4.31
\$AAN[buffermode]	!AA[buffermode]	Reads/sets the usage mode for buffer of RS-232 port. (Supported on firmware version 4 and later)	7.4.32
\$AAS[lastdatahdl]	\$AA[lastdatahdl]	Reads/sets the retention mode of the last message. (Supported on firmware version 4 and later)	7.4.33

DI/DO Command

Command	Response	Description	Section
\$AAYN	!AA(status)	Reads the onboard DI-1/2/3/4/5	7.4.35
\$AAZNV	!AA(status)	Reads/sets the onboard DO-1/2/3/4/5	7.4.36
#**	No Response	Simultaneous sampling of DI channels on multiple modules	7.4.37
\$AA4	!AASV	Reads the synchronized DI data	7.4.38
\$AAL[data]	!AA	Sets DO-1/2/3/4 on the I-7522A module	7.4.39
\$AAR	!AAS	Reads DI-1/2/3/4 on the I-7522A module.	7.4.40
@AA[data]	>AA[data]	Reads/sets the onboard DI/DO-1/2/3/4/5	7.4.41
#AABBHH	>	Sets the multiple onboard DO-1/2/3/4/5	7.4.42
#AABCDD	>	Sets the single onboard DO-1/2/3/4/5	7.4.43

Watchdog Command

Command	Response	Description	Section
~**	No Response	Broadcasts the “Host is OK” message to the modules on the RS-485 network.	7.4.44
~AA0	!AASS	Reads the module status.	7.4.45
~AA1	!AA	Resets the module status.	7.4.46
~AA2	!AASTT	Reads the host watchdog status and value	7.4.47
~AA3ETT	!AASTT	Enables/disables the host watchdog and sets the timeout value.	7.4.48
~AA4P	!AAV	Reads the power-on value for DO channels.	7.4.49
~AA4S	!AAV	Reads the safe value for DO channels.	7.4.49
~AA5P	!AAV	Sets the power-on value for DO	7.4.50
~AA5S	!AAV	Sets the safe value for DO channels.	7.4.50

New Commands in Firmware Version V4

Command	Description	Section
\$AATN[CrLfmode]	Adds support for Modbus ASCII and Modbus RTU.	7.4.9
\$AAUR	Returns "N/A" if there is no data in COM port buffer.	7.4.25
\$AAUA	Reads one message from buffer of every RS-232 port.	7.4.26
\$AAUC	Clears data in buffer of a specified RS-232 port.	7.4.27
\$AAUN	Reads message quantity in buffer of a specified COM port.	7.4.28
\$AAUL	Reads/sets the minimum length of response	7.4.29
\$AAUD	Reads/sets the character used to fill up a message, while its length is less than the minimum length set by command \$AAUL.	7.4.30
\$AAUS	Reads/sets the delimiter character for reading one message from buffer of every COM port.	7.4.31
\$AAN	Reads/sets the usage mode for buffer of RS-232 port.	7.4.32
\$AAS	Reads/sets the retention mode of the last message.	7.4.33

7.4. Commands

7.4.1. \$AAA[addr]

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527

- **Description:** Reads/sets the module address
\$AAA[chk](CrLf) - reads the module address saved in EEPROM
\$AAA[addr][chk](CrLf) - sets the module address
- **Syntax:** \$AAA[addr][chk](CrLf)
\$ Delimiter character
AA 2-character module address, ranged from 00 to FF in HEX format
A Command code
[chk] 2-character checksum. If the checksum is disabled -> no [chk]
(CrLf) End character
- **Response:** valid command: !AA[chk](CrLf)
 invalid command: ?AA[chk](CrLf)
 no response: syntax error, communication error, or address error
! Delimiter character indicating a valid command
? Delimiter character indicating an invalid command
AA 2-character module address in HEX format
[chk] 2-character checksum. If the checksum is disabled -> no [chk]
(CrLf) End character
- **Example:**

Command: \$01A02(CrLf)	Sets the module address to 02
Response: !01(CrLf)	Complete message from module address 01
Command: \$02AA0(CrLf)	Sets the module address to A0
Response: !02(CrLf)	Complete message from module address 02
Command: \$00A(CrLf)	Reads the module address stored in EEPROM
Response: !02(CrLf)	Returns the module address in EEPROM is 02
- **Notes:** If the module starts up in INIT mode, (powered on with INIT* pin and GND pin shorted), the module address is 00 in commands for reading settings in EEPROM.

7.4.2. \$ AABN[baud-rate]

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527

- **Description:** Reads/sets the baud rate for COM-1/2/3/4/5/6/7/8
\$AABN[chk](CrLf) - reads the baud rate for COM-1/2/3/4/5/6/7/8 saved in EEPROM
\$AABN[baud-rate][chk](CrLf) - sets the baud rate for COM1 ~ 8
- **Syntax:** \$AABN[baud-rate][chk](CrLf)
 - \$ Delimiter character
 - AA 2-character COM port address, ranged from 00 to FF in HEX format
 - B Command code
 - N=0 Reads/sets the baud rate for COM2
 - N=1 Reads/sets the baud rate for COM1/3/4/5/6/7/8
 - [baud rate] Available values are 300/600/1200/2400/4800/9600/19200/38400/57600/115200
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Response:** valid command: !AA[baud-rate][chk](CrLf)
invalid command: ?AA[chk](CrLf)
no response: syntax error, communication error, or address error
 - ! Delimiter character indicating a valid command
 - ? Delimiter character indicating an invalid command
 - AA 2-character COM port address in HEX format
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Example:**

Command:	\$01B0115200(CrLf)	Sets the baud rate for COM2 on module address 01 to 115200
Response:	!01(CrLf)	Complete message from COM2 (address 01)
Command:	\$01B19600(CrLf)	Sets the baud rate for COM1 on module address 01 to 9600
Response:	!01(CrLf)	Complete message from COM1 (address 01)
Command:	\$02B138400(CrLf)	Sets the baud rate for COM3 on module address 01 to 38400
Response:	!02(CrLf)	Complete message from COM3 (address 02)
Command:	\$03B1 (CrLf)	Reads the baud rate for COM4 on module address 01
Response:	!0357600(CrLf)	Returns baud rate is 57600 for COM4 (address 03)

7.4.3. \$AADN[data-bit]

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527

- **Description:** Reads/sets the data bit for COM-1/2/3/4/5/6/7/8
\$AADN[chk](CrLf) – reads the data bit saved in EEPROM
\$AADN[data-bit][chk](CrLf) - sets the data bit for COM-1/2/3/4/5/6/7/8
- **Syntax:** \$AADN[data-bit][chk](CrLf)
\$ Delimiter character
AA 2-character COM port address, ranged from 00 to FF in HEX format
D Command code
N=0 Reads/sets the data bit for COM2
N=1 Reads/sets the data bit for COM1/3/4/5/6/7/8
[data-bit] Available value is 7 or 8
[chk] 2-character checksum. If the checksum is disabled -> no [chk]
(CrLf) End character
- **Response:** valid command: !AA[data-bit][chk](CrLf)
 invalid command: ?AA[chk](CrLf)
 no response: syntax error, communication error, or address error
! Delimiter character indicating a valid command
? Delimiter character indicating an invalid command
AA 2-character COM port address in HEX format
[chk] 2-character checksum. If the checksum is disabled -> no [chk]
(CrLf) End character
- **Example:** (takes I-7523 with module address 01 as examples)
Command: \$01D08(CrLf) Sets the data bit for COM2 to 8 bits
Response: !01(CrLf) Complete message from COM2 (address 01)
Command: \$01D17(CrLf) Sets the data bit for COM1 to 7 bits
Response: !02(CrLf) Complete message from COM1 (address 01)
Command: \$02D17(CrLf) Sets the data bit for COM3 to 7 bits
Response: !02(CrLf) Complete message from COM3 (address 02)
Command: \$03D17(CrLf) Sets the data bit for COM4 to 7 bits
Response: !03(CrLf) Complete message from COM4 (address 03)

- **Notes:**

The data bit support table

	I-7521	I-7522	I-7523	I-7522A	I-7524	I-7527
COM1	7/8	7/8	7/8	7/8	7/8	7/8
COM2	7/8	7/8	7/8	7/8	7/8	7/8
COM3	-	7/8	7/8	7/8	7/8	7/8
COM4	-	-	7/8	-	7/8	7/8
COM5	-	-	-	-	7/8	7/8
COM6	-	-	-	-	-	7/8
COM7	-	-	-	-	-	7/8
COM8	-	-	-	-	-	7/8

7.4.4. \$AAPN[parity-bit]

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527

- **Description:** Reads/sets the parity bit for COM-1/2/3/4/5/6/7/8
\$AAPN[chk](CrLf) – reads the parity bit saved in EEPROM
\$AAPN[parity-bit][chk](CrLf) - sets the parity bit for COM-1/2/3/4/5/6/7/8
- **Syntax:** \$AAPN[parity-bit][chk](CrLf)
 - \$ Delimiter character
 - AA 2-character COM port address, ranged from 00 to FF in HEX format
 - P Command code
 - N=0 Reads/sets parity bit for COM2
 - N=1 Reads/sets parity bit for COM1/3/4/5/6/7/8
 - [data-bit] 0=NONE, 1=EVEN, 2=ODD
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Response:** valid command: !AA[parity-bit][chk](CrLf)
invalid command: ?AA[chk](CrLf)
no response: syntax error, communication error, or address error
 - ! Delimiter character indicating a valid command
 - ? Delimiter character indicating an invalid command
 - AA 2-character COM port address in HEX format
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Example:** (takes I-7523 with module address 01 as examples)

Command:	\$01P00(CrLf)	Sets the parity bit for COM2 to NONE
Response:	!01(CrLf)	Complete message from COM2 (address 01)
Command:	\$01P10(CrLf)	Sets the parity bit for COM1 to NONE
Response:	!02(CrLf)	Complete message from COM1 (address 01)
Command:	\$02P11(CrLf)	Sets the parity bit for COM3 to EVEN
Response:	!02(CrLf)	Complete message from COM3 (address 02)
Command:	\$03P12(CrLf)	Sets the parity bit for COM4 to ODD
Response:	!03(CrLf)	Complete message from COM4 (address 03)

- **Notes:**

The parity bit support table

	I-7521	I-7522	I-7523	I-7522A	I-7524	I-7527
COM1	N/E/O	N/E/O	N/E/O	N/E/O	N/E/O	N/E/O
COM2	N/E/O	N/E/O	N/E/O	N/E/O	N/E/O	N/E/O
COM3	-	N/E/O	N/E/O	N/E/O	N/E/O	N/E/O
COM4	-	-	N/E/O	-	N/E/O	N/E/O
COM5	-	-	-	-	N/E/O	N/E/O
COM6	-	-	-	-	-	N/E/O
COM7	-	-	-	-	-	N/E/O
COM8	-	-	-	-	-	N/E/O

7.4.5. \$AAON[stop-bit]

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527

- **Description:** Reads/sets the stop bit for COM1/2/3/4/5/6/7/8
\$AAON[chk](CrLf) – reads the stop bit saved in EEPROM
\$AAON[stop-bit][chk](CrLf) – sets the stop bit for COM 3/4/5/6/7/8
- **Syntax:** \$AAPN[stop-bit][chk](CrLf)
 - \$ Delimiter character
 - AA 2-character COM port address, ranged from 00 to FF in HEX format
 - O Command code
 - N=0 Reads/sets the stop bit for COM2
 - N=1 Reads/sets the stop bit for COM1/3/4/5/6/7/8
 - [stop-bit] Available value is 1 or 2
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Response:** valid command: !AA[stop-bit][chk](CrLf)
invalid command: ?AA[chk](CrLf)
no response: syntax error, communication error, or address error
 - ! Delimiter character indicating a valid command
 - ? Delimiter character indicating an invalid command
 - AA 2-character COM port address in HEX format
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Example:** (takes I-7523 with module address 01 as examples)
 - Command: \$02O12(CrLf) Sets the stop bit for COM3 to 2
 - Response: !02(CrLf) Complete message from COM3 (address 02)
 - Command: \$03O12(CrLf) Sets the stop bit for COM4 to 2
 - Response: !03(CrLf) Complete message from COM4 (address 03)

● **Notes:**

The stop bit support table

	I-7521	I-7522	I-7523	I-7522A	I-7524	I-7527
COM1	N81, E81, O81, N82, N71, E71, O71, N72, E72, O72					
COM2	N81, E81, O81, N71, E71, O71					
COM3	-	1/2	1/2	1/2	1/2	1/2
COM4	-	-	1/2	-	1/2	1/2
COM5	-	-	-	-	1/2	1/2
COM6	-	-	-	-	-	1/2
COM7	-	-	-	-	-	1/2
COM8	-	-	-	-	-	1/2

1. The maximum number of bits for COM1 is 10, including data bits, parity bit(s) and stop bit(s). Therefore, if data bits is set to 8 and stop bit is set to 2, parity can be set to NONE only for COM1.
2. The stop bit for COM2 is fixed to 1.
3. COM3 ~ COM8 can support stop bit 1 or 2
4. COM 1/3/4/5/6/7/8 can be used to connect with HP 34401A.

7.4.6. \$AA6[ID]

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527

- **Description:** Sets ID for COM-1/3/4/5/6/7/8, with a maximum length of 50 characters

- **Syntax:** \$AA6[ID][chk](CrLf)

\$ Delimiter character

AA 2-character COM port address, ranged from 00 to FF in HEX format

6 Command code

[ID] the ID for COM port, up to 50 characters

[chk] 2-character checksum. If the checksum is disabled -> no [chk]

(CrLf) End character

- **Response:** valid command: !AA[chk](CrLf)
invalid command: ?AA[chk](CrLf)
no response: syntax error, communication error, or address error

! Delimiter character indicating a valid command

? Delimiter character indicating an invalid command

AA 2-character COM port address in HEX format

[chk] 2-character checksum. If the checksum is disabled -> no [chk]

(CrLf) End character

- **Example:** (takes I-7523 with module address 01 as an example)

Command: \$016Temperature1(CrLf) Sets the ID for COM1 to Temperature1

Response: !01(CrLf) Complete message from COM1 (address 01)

Command: \$026HP34401A-1(CrLf) Sets the ID for COM3 to HP34401A-1

Response: !02(CrLf) Complete message from COM3 (address 02)

Command: \$036HP34401A-2(CrLf) Sets the ID for COM4 to HP34401A-2

Response: !03(CrLf) Complete message from COM4 (address 03)

7.4.7. \$AA7

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527

- **Description:** Reads the ID for COM-1/3/4/5/6/7/8
- **Syntax:** \$AA7[chk](CrLf)
 - \$ Delimiter character
 - AA 2-character COM port address, ranged from 00 to FF in HEX format
 - 7 Command code
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Response:** valid command: !AA[ID][chk](CrLf)
invalid command: ?AA[chk](CrLf)
no response: syntax error, communication error, or address error
 - ! Delimiter character indicating a valid command
 - ? Delimiter character indicating an invalid command
 - AA 2-character COM port address in HEX format
 - [ID] ID for COM Port, up to 50 characters
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Example:** (takes I-7523 with module address 01 as an example)
 - Command: \$017(CrLf) Reads the ID for COM1
 - Response: !01Temperature1(CrLf) Returns ID of COM1 is Temperature1
 - Command: \$027(CrLf) Reads the ID for COM3
 - Response: !02HP34401A-1(CrLf) Returns ID of COM3 is HP34401A-1
 - Command: \$037(CrLf) Reads the ID for COM4
 - Response: !03HP34401A-2 (CrLf) Returns ID of COM4 is HP34401A-2

7.4.8. \$AAKV

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527

- **Description:** Reads/sets the checksum status for COM2 (RS485)
\$AAK[chk](CrLf) – reads the checksum status for COM2 saved in EEPROM
\$AAKV[chk](CrLf) – enables/disables the checksum for COM2
- **Syntax:** \$AAKV[chk](CrLf)
 - \$ Delimiter character
 - AA 2-character COM port address, ranged from 00 to FF in HEX format
 - K Command code
 - V=0 Disables the checksum for COM2
 - V=1 Enables the checksum for COM2
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Response:** valid command: !AA(V)[chk](CrLf)
invalid command: ?AA[chk](CrLf)
no response: syntax error, communication error, or address error
 - ! Delimiter character indicating a valid command
 - ? Delimiter character indicating an invalid command
 - AA 2-character COM port address in HEX format
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Example:** (takes two I-7523 modules with address 01 and 04 as examples)
 - Command: \$01K000(CrLf) Disables the checksum for COM2 with checksum result -00 for the checksum enabled module.
 - Response: !0182(CrLf) Complete message with checksum 82
 - Command: \$04K1(CrLf) Enables the checksum for COM2 on the module with address 4
 - Response: !04(CrLf) Complete message from module address 04

7.4.9. \$AATN[CrLfmode]

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527

- **Description:** Reads/sets the end character used to judge the end of a command/response for COM1/2/3/4/5/6/7/8

\$AATN[chk](CrLf) – reads the CrLfmode saved in EEPROM

\$AATN[CrLfmode][chk](CrLf) – sets the CrLfmode for COM1 ~ 8

- **Syntax:** \$AATN[CrLfmode][chk](CrLf)

\$ Delimiter character

AA 2-character COM port address, ranged from 00 to FF in HEX format

T Command code

N=0 Reads/sets the CrLfmode for COM2

N=1 Reads/sets the CrLfmode for COM 1/3/4/5/6/7/8

[CrLfmode] 0 = the end character is 0x0D (CR)

1 = the end character is 0x0D+0x0A (CR+LF)

2 = the end character is 0x0A (LF)

3 = the end character is 0x0A+0x0D (LF+CR)

4 = no end character, timeout is used instead of end character

5 = uses Modbus ASCII protocol

6 = uses Modbus RTU protocol

[chk] 2-character checksum. If the checksum is disabled -> no [chk]

(CrLf) End character

- **Response:** valid command: !AA[CrLfmode][chk](CrLf)
invalid command: ?AA[chk](CrLf)
no response: syntax error, communication error, or address error

! Delimiter character indicating a valid command

? Delimiter character indicating an invalid command

AA 2-character COM port address in HEX format

[chk] 2-character checksum. If the checksum is disabled -> no [chk]

(CrLf) End character

- **Example:** (takes I-7523 with module address 01 as an example)

Command: \$01T0(CrLf) Reads CrLfmode for COM2

Response: !014(CrLf) Returns CrLfmode of COM2 is 4 - no end character

Command: \$01T1(CrLf) Reads CrLfmode for COM1

Response: !011(CrLf) Returns CrLfmode of COM1 is 1 - (CR+LF)

- **Notes:**

(1) **Once CrLfmode 5 or 6 is set for a port, all the COM ports on the same module can be set to 5 or 6 only.**

(2) If mode 5 or 6 is set for COM2 (RS-485), all other com ports on the same module will be set to the same mode as COM2 automatically.

(3) If all the RS-232 ports are set to use the same CrLfmode (5 or 6) as COM2, they can be individually set to another mode different from COM2.

For example, while \$AAT05 is used to set COM2 to use Modbus ASCII protocol, all other COM ports on the same module will be set to mode5 automatically. At this point, \$AAT16 can be used to set other RS-232 ports to use mode6 (Modbus RTU).

Similarly, if all RS-232 ports are set to use mode6 caused by mode6 is set for COM2, you can use \$AAT15 to set these RS-232 ports to mode5. (Modbus ASCII)

(4) When mode 5 or 6 is set for COM2, it still can respond to the original reading/setting commands.

(5) If there are several RS-232 devices using different end character and connecting to the same I-752N module, set the end character for a COM port in line with the connected device, and then set COM2 to one of these modes.

For example, if the following devices are connected to the same I-7524 module:

RS-232 Device1 uses end character: CR (set to mode0)

RS-232 Device2 uses end character: CR+LF (set to mode1)

RS-232 Device3 uses end character: LF (set to mode2)

RS-232 Device4 uses end character: LF+CR (set to mode3)

The CrLfmode for COM2 can be set to 0 (CR, the same as device1). When COM2 receives data from Host PC with CR end character, it will delete the end character and bypass data with corresponding end character to the specified RS-232 port.

(6) If one of the RS-232 ports is set to mode4 (no end character), it is recommended to set mode 4 for COM2, too.

7.4.10. \$AAW

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527

- **Description:** Reads the status of CTS line of COM1/3/4/5.
- **Syntax:** \$AAW[chk](CrLf)
 - \$ Delimiter character
 - AA 2-character COM port address, ranged from 00 to FF in HEX format
 - W Command code
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Response:**
 - valid command: !AAS[chk](CrLf)
 - invalid command: ?AA[chk](CrLf)
 - no response: syntax error, communication error, or address error
 - ! Delimiter character indicating a valid command
 - ? Delimiter character indicating an invalid command
 - AA 2-character COM port address in HEX format
 - S=0 CTS line is low (inactive)
 - S=1 CTS line is high (active)
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Example:** (takes I-7523 with module address 01 as an example)
 - Command: \$01W(CrLf) Reads the status of CTS line of COM1
 - Response: !010(CrLf) Returns the CTS line of COM1 is low
 - Command: \$02W(CrLf) Reads the status of CTS line of COM3
 - Response: !021(CrLf) Returns the CTS line of COM3 is high

7.4.11. \$AAXV

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527

- **Description:** Sets the RTS line of COM 1/3/4/5
- **Syntax:** \$AAXV[chk](CrLf)
 - \$ Delimiter character
 - AA 2-character COM port address, ranged from 00 to FF in HEX format
 - X Command code
 - V=0 Sets the RTS line to low (inactive)
 - V=1 Sets the RTS line to high (active)
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Response:** valid command: !AA[stop-bit][chk](CrLf)
invalid command: ?AA[chk](CrLf)
no response: syntax error, communication error, or address error
 - ! Delimiter character indicating a valid command
 - ? Delimiter character indicating an invalid command
 - AA 2-character COM port address in HEX format
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Example:** (takes I-7523 with module address 01 as an example)
 - Command: \$01X0CrLf Sets the RTS line of COM1 to low (inactive)
 - Response: !01(CrLf) Complete message from COM1 (address 01)
 - Command: \$02X1(CrLf) Sets the RTS line of COM3 to high (active)
 - Response: !02(CrLf) Complete message from COM3 (address 02)

7.4.12. \$AA2

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527

- **Description:** Reads the COM2 configuration saved in EEPROM
- **Syntax:** \$AA2[chk](CrLf)
 - \$ Delimiter character
 - AA 2-character module address, ranged from 00 to FF in HEX format
 - 2 Command code
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Response:** valid command: !AA40BDPK[chk](CrLf)
 - invalid command: ?AA[chk](CrLf)
 - no response: syntax error, communication error, or address error
 - ! Delimiter character indicating a valid command
 - ? Delimiter character indicating an invalid command
 - AA 2-character module address in HEX format
 - 40 Type code of module
 - B Baud rate: 1= 300, 2= 600, 3= 1200, 4= 2400, 5= 4800,
6= 9600, 7= 19200, 8= 38400, 9= 57600, A= 115200
 - D Data bit: 7 or 8
 - P parity bit: 0= NONE, 1= EVEN, 2= ODD
 - K checksum: 0= checksum disabled, 1= checksum enabled
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Example:** (takes the module powered on in INIT mode as an example)
 - Command: \$002(CrLf) Reads the configuration of COM2 with address 0
 - Response: !00406800(CrLf) Returns the COM2 is set to baud rate 9600 bps,
Data bit = 8 bits, parity bit = NONE, checksum= disabled
 - Command: \$002(CrLf) Reads the configuration of COM2 with address 0
 - Response: !0040A801(CrLf) Returns the COM2 is set to Baud Rate 115200 bps,
Data bit = 8 bits, parity bit = NONE, checksum= enabled

7.4.13. \$AAIV

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527

- **Description:** Restores all COM ports configuration to factory defaults. The address in \$AAIV command can be any COM port address of the module.
- **Syntax:** \$AAIV[chk](CrLf)
 - \$ Delimiter character
 - AA 2-character module address, ranged from 00 to FF in HEX format
 - I Command code
 - V=1 Restores all COM ports configuration to factory defaults.
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Response:** valid command: !AA[chk](CrLf)
 - invalid command:?[chk](CrLf)
 - no response: syntax error, communication error, or address error
 - ! Delimiter character indicating a valid command
 - ? Delimiter character indicating an invalid command
 - AA 2-character COM port address in HEX format
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Example:** (takes I-7522 with module address 01 as an example)
 - Command: \$02I1(CrLf) Restores all COM ports settings to factory defaults
 - Response: !02(CrLf) Complete message from COM3 (address 02)
 - Command: \$01I1(CrLf) Restores all COM ports settings to factory defaults
 - Response: !01(CrLf) Complete message from COM1 (address 01)

- **Notes:**

COM port factory default settings

	COM1	COM2	COM3	COM4	COM5	COM6	COM7	COM8	
Baud Rate	9600 bps								
Data bit	8 bits								
Parity bit	None								
Stop bit	1								
Checksum	Disable								
CrLfmode	4								
WatchDog	Disable								
Delimiter for bypass command (Sec 5.5.10)	:								
Address prefix (Sec.5.5.38)	Disabled	None						Disabled	
Bypass data to COM2 (Sec.5.5.39)	Disabled	None						Disabled	

7.4.14. \$AA5

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527

- **Description:** Module watchdog is a hardware watchdog; it is enabled upon the module is powered-on and it will automatically reset the module while the module is abnormal. After the watchdog reset, all analog and digital output channels will revert to their initial start (power-on value), it may differ from the latest status in the previous execution. If a watchdog reset is read by the \$AA5 command, the host must send output commands to the module again in order to make the output state the same after restart.
- **Syntax:** \$AA5[chk](CrLf)
 - \$ Delimiter character
 - AA 2-character module address, ranged from 00 to FF in HEX format
 - 5 Command code
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Response:** valid command: !AAS[chk](CrLf)
 - invalid command: ?AA[chk](CrLf)
 - no response: syntax error, communication error, or address error
 - ! Delimiter character indicating a valid command
 - ? Delimiter character indicating an invalid command
 - AA 2-character module address in HEX format
 - S=0 The module has not been restarted since the last reading
 - S=1 It is the first time to read reset status after powered-on, or the module had been restarted since the last reading.
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Example:** (takes I-7521 with module address 01 as an example)
 - Command: \$015(CrLf) Reads the reset status of module address 01
 - Response: !011(CrLf) Returns it's the first time to read reset status (after the module was powered-on).
 - Command: \$015(CrLf) Reads the reset status of module address 01
 - Response: !010(CrLf) Returns the module has not been restarted since the last reading (status)
 - Command: \$015(CrLf) Reads the reset status of module address 01
 - Response: !011(CrLf) Returns the module has been restarted since the last reading.

7.4.15. \$AAF

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527

- **Description:** Reads the firmware version number
- **Syntax:** \$AAF[chk](CrLf)
 - \$ Delimiter character
 - AA 2-character module address, ranged from 00 to FF in HEX format
 - F Command code
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Response:** valid command: !AA[number][chk](CrLf)
 - invalid command: ?AA[chk](CrLf)
 - no response: syntax error, communication error, or address error
 - ! Delimiter character indicating a valid command
 - ? Delimiter character indicating an invalid command
 - AA 2-character module address in HEX format
 - number the firmware version (4 ~ 5 characters)
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Example:** (takes two I-7523 modules with address 01 and 02 as examples)
 - Command: \$01F(CrLf) Reads the firmware version for module address 01
 - Response: !01A2.0(CrLf) Returns the firmware version is A2.0
 - Command: \$02F(CrLf) Reads the firmware version for module address 02
 - Response: !02A3.0(CrLf) Returns the firmware version is A3.0

7.4.16. \$AAM

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527

- **Description:** Reads the model number
- **Syntax:** \$AAM[chk](CrLf)
 - \$ Delimiter character
 - AA 2-character module address, ranged from 00 to FF in HEX format
 - M Command code
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Response:** valid command: !AA[name][chk](CrLf)
 - invalid command: ?AA[chk](CrLf)
 - no response: syntax error, communication error, or address error
 - ! Delimiter character indicating a valid command
 - ? Delimiter character indicating an invalid command
 - AA 2-character module address in HEX format
 - name model number(4 ~ 5 characters)
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Example:** (takes I-7521 with address 01 and I-7523 with address 02 as examples)
 - Command: \$01M(CrLf) Reads the name of module address 01
 - Response: !017521(CrLf) Returns the module name is I-7521
 - Command: \$02M(CrLf) Reads the name of module address 01
 - Response: !027523(CrLf) Returns the module name is I-7523

7.4.17. \$AAC[delimiter]

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527

- **Description:** Reads/sets the delimiter for command to bypass data to COM-1/3/4/5/6/7/8. The default delimiter is a colon “:”, different delimiter can be set for each COM port. The following characters cannot be used as a delimiter: \$, ~, #, @, %, >, !, ?, Cr & Lf .

\$AAC[chk](CrLf) - Reads the delimiter for bypass data command

\$AAC[delimiter][chk](CrLf) - Sets the delimiter for bypass data command

- **Syntax:** \$AAC[delimiter][chk](CrLf)

\$ Delimiter character

AA 2-character COM port address, ranged from 00 to FF in HEX format

C Command code

[delimiter] The delimiter, default is a colon “:”, characters that cannot be used including : \$, ~, #, @, %, >, !, ?, Cr & Lf.

[chk] 2-character checksum. If the checksum is disabled -> no [chk]

(CrLf) End character

- **Response:** valid command: !AA[[delimiter][chk](CrLf)
invalid command: ?AA[chk](CrLf)
no response: syntax error, communication error, or address error

! Delimiter character indicating a valid command

? Delimiter character indicating an invalid command

AA 2-character COM port address in HEX format

[chk] 2-character checksum. If the checksum is disabled -> no [chk]

(CrLf) End character

- **Example:** (takes I-7523 with module address 01 as an example)

Command: \$01C(CrLf) Reads delimiter of bypass command for COM1

Response: !01:(CrLf) Returns the delimiter of bypass command is " : "

Command: \$02C(CrLf) Reads delimiter of bypass command for COM3

Response: !02:(CrLf) Returns the delimiter of bypass command is " : "

Command: \$03C*(CrLf) Sets the delimiter of bypass command to " * "

Response: !03(CrLf) Complete message from COM4 (address 03)

7.4.18. \$AAD

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527

- **Description:** Reads the delimiter for command to bypass data to COM-1/3/4/5/6/7/8
- **Syntax:** \$AAD[chk](CrLf)
 - \$ Delimiter character
 - AA 2-character COM port address, ranged from 00 to FF in HEX format
 - D Command code
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Response:** valid command: !AA[delimiter][chk](CrLf)
invalid command: ?AA[chk](CrLf)
no response: syntax error, communication error, or address error
 - ! Delimiter character indicating a valid command
 - ? Delimiter character indicating an invalid command
 - AA 2-character COM port address in HEX format
 - (delimiter) Delimiter character, the default delimiter is " : "
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Example:** (takes I-7523 with module address 01 as an example)

Command: \$01D(CrLf)	Reads delimiter of bypass command for COM1
Response: !01:(CrLf)	Returns the delimiter of bypass command is " : "
Command: \$02D(CrLf)	Reads delimiter of bypass command for COM3
Response: !02:(CrLf)	Returns the delimiter of bypass command is " : "
Command: \$03D(CrLf)	Reads delimiter of bypass command for COM4
Response: !03*(CrLf)	Returns the delimiter of bypass command is " * "

7.4.19. [delimiter]AA[bypass]

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527

- **Description:** Bypasses the data string to COM-1/3/4/5/6/7/8
- **Syntax:** (delimiter)AA(bypass)[chk](CrLf)
 - (delimiter) Delimiter character
 - AA 2-character COM port address, ranged from 00 to FF in HEX format
 - (bypass) The data string to bypass to COM-1/3/4/5/6/7/8
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Response:** response from serial device
- **Example:** (takes I-7523 with module address 01 as an example, sets the delimiters for COM1/3/4 are colon (:), semicolon (;) and asterisk (*))
 - Command: :01abcde(CrLf) Sends "abcde" to COM1
 - Response: device response Returns the data received from COM1
 - Command: ;02123456789(CrLf) Sends "123456789" to COM3
 - Response: device response Returns the data received from COM3
 - Command: *03test(CrLf) Sends "test" to COM4
 - Response: device response Returns the data received from COM4

7.4.20. \$AAJN[timeout]

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527, firmware version 3.0 or later

- **Description:** Reads/sets the timeout settings for the end of communication
\$AAJN[chk](CrLf) - reads the timeout settings
\$AAJN[timeout][chk](CrLf) - sets the timeout for the end of a command/response
- **Syntax:** \$AAJN[timeout][chk](CrLf)
 - \$ Delimiter character
 - AA 2-character COM port address, ranged from 00 to FF in HEX format
 - J Command code
 - N=0 Reads/sets the timeout0 for COM2
 - N=1 Reads/sets the timeout1 for COM 1/3/4/5/6/7/8, the default value is 1000ms
 - N=2 Reads/sets the timeout2 for COM 1/3/4/5/6/7/8
 - [timeout] timeout value, ranged from 0 to 4294967259 (ms)
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Response:** valid command: !AA[timeout value][chk](CrLf)
invalid command: ?AA[chk](CrLf)
no response: syntax error, communication error, or address error
 - ! Delimiter character indicating a valid command
 - ? Delimiter character indicating an invalid command
 - AA 2-character COM port address in HEX format
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Example:** (takes I-7522 with module address 01 as an example)
 - Command: \$01J01000 (CrLf) Sets timeout0 for COM2 to 1000ms.
 - Response: !01(CrLf) Complete message from COM2 (address 01)
 - Command: \$01J11500 (CrLf) Sets timeout1 for COM1 to 1500ms
 - Response: !01(CrLf) Complete message from COM1 (address 01)
 - Command: \$01J1 (CrLf) Reads timeout1 for COM1
 - Response: !011500(CrLf) Returns the timeout1 for COM1 is 1500ms

- **Notes:**

(1) Timeout0 for COM2:

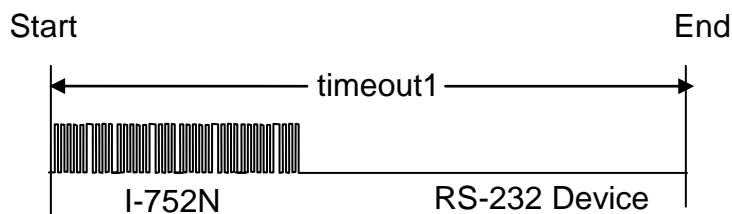
The timeout0 for COM2 is valid only when the CfLfmode is set to 4 (no end character). If the I-752N series module does not receive data from the host in the timeout0 interval, it will determine that the data transfer is ended, and then execute the command.

(2) Timeout1 for COM 1/3/4/5/6/7/8:

I-752N will bypass the data to the specified COM port and then wait for a response from the connected device, while it receives a bypass data command from COM2. If no data is received from the serial device before the "timeout1" period has elapsed, the I-752N will terminate the communication and return to standby mode.

Any data received beyond the waiting time will be placed in COM port buffer, command "\$AAU", "\$AAUR", "\$AAUA" are provided for reading the data in buffer.

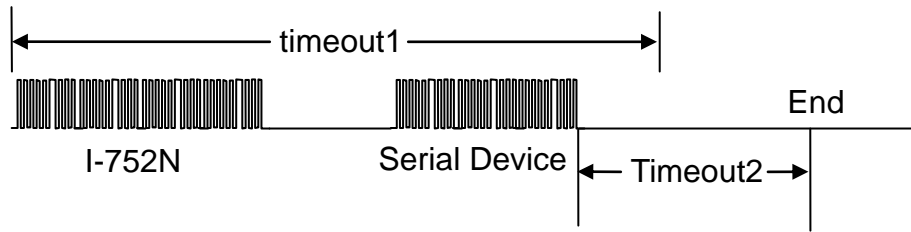
Timeout1 period starts from when the I-752N series module bypasses data to the specified RS-232 port, so the time used to transmit data to the RS-232 device must be taken into account when setting timeout1.



(3) Timeout2 for COM 1/3/4/5/6/7/8

If any data is received before the timeout1 period has elapsed, and CrLfmode is set to one in range of 0 to 3, the received data will be sent to COM2 (to the Host) after the End character is arrived.

If no end character is arrived before the timeout1 period elapsed, and no new data received in the timeout2 period, the I-752N series module will terminate the communication and send the received data to COM2 (to the Host)



When the CrLfmode is set to 4 (no end character), the data transfer is complete if no data arrived in the timeout2 period. The I-752N series module will bypass the data received to COM2 (Host).

The timeout1 must be greater than timeout2, in order to prevent the I-752N series module from ending the communication too early.

Each time the RS-232 port receives a character from the connected device, timeout2 starts to count from 0 again. If the data transfer is fine, no transfer fails or any error, you can keep these settings without modification.

7.4.21. \$AAGN[trigger-level]

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527

- **Description:** Reads/sets the trigger level for a COM port buffer.
You do not have to modify it if the data transfer is fine. If you often receive incomplete messages, try to increase the length of trigger level. Otherwise, if the data is sometimes lost, you can reduce the trigger level.

\$AAGN[chk](CrLf) - reads the trigger level for a buffer

\$AAGN[trigger-level][chk](CrLf) - sets the trigger level

- **Syntax:** \$AAGN[trigger-level][chk](CrLf)
 - \$ Delimiter character
 - AA 2-character COM port address, ranged from 00 to FF in HEX format
 - G Command code
 - N=0 Reads the trigger level for the buffer of COM2
 - N=1 Reads/sets the trigger level for the buffer of COM1/3/4/5/6/7/8
 - [trigger level] Available values are 1/4/8/14 (characters) for buffer of COM3/4/5/6/7/8, (the default value is 8)
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Response:** valid command: !AA[trigger-level][chk](CrLf)
invalid command: ?AA[chk](CrLf)
no response: syntax error, communication error, or address error
 - ! Delimiter character indicating a valid command
 - ? Delimiter character indicating an invalid command
 - AA 2-character COM port address in HEX format
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Example:** (takes I-7523 with module address 01 as an example)
 - Command: \$01G0(CrLf) Reads the trigger level for COM2
 - Response: !011(CrLf) Returns the trigger level for COM2 is 1
 - Command: \$02G14(CrLf) Sets the trigger level for COM3 to 4
 - Response: !02(CrLf) Complete message from COM3 (address 02)
 - Command: \$01G18(CrLf) Sets the trigger level for COM1 to 8
 - Response: ?01(CrLf) Returns the command is invalid for COM1

- **Notes:**
The trigger level for COM1 and COM2 is fixed to 1 and cannot be changed.

7.4.22. \$AAEV

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527, firmware version 3.05 or later

- **Description:** Reads/sets the address prefix status on the response.

Response without COM port address information can speed up communication. The I-752N series module does not add COM port address to the respond by default, but it provides an option to enable the function. The address prefix is useful to confirm COM port from which the data is received, and to maintain the consistency of writing DCON communication code.

- **Syntax:** \$AAEV[chk](CrLf)

\$ Delimiter character
AA 2-character COM port address, ranged from 00 to FF in HEX format
E Command code
V=0 Disables the address prefix on the response
V=1 Enables the address prefix on the response
[chk] 2-character checksum. If the checksum is disabled -> no [chk]
(CrLf) End character

- **Response:** valid command: !AA[V][chk](CrLf)
 invalid command:?[chk](CrLf)
 no response: syntax error, communication error, or address error

! Delimiter character indicating a valid command
? Delimiter character indicating an invalid command
AA 2-character COM port address in HEX format
[chk] 2-character checksum. If the checksum is disabled -> no [chk]
(CrLf) End character

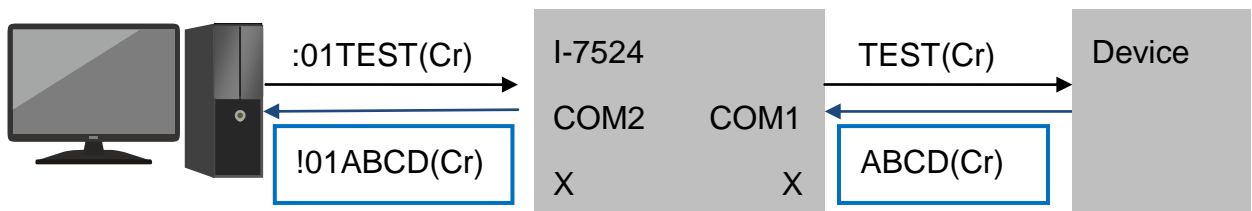
- **Example:** (takes I-7521 with module address 01 as an example)

Command:	\$01E(CrLf)	Reads the address prefix setting for COM1.
Response:	!010(CrLf)	Returns the address prefix of COM1 is disabled.
Command:	\$01E1(CrLf)	Enables the address prefix function for COM1
Response:	!01(CrLf)	Complete message from COM1 (address 01)

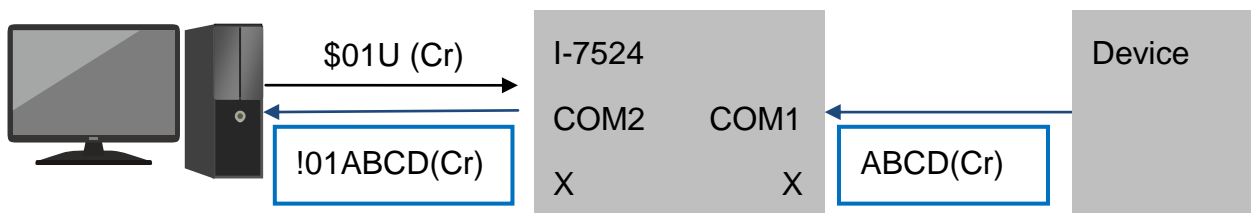
- **Notes:**

If the address prefix is enabled for a COM port, both the responses of [delimiter]AA[bypass data] and \$AAU commands will be prefixed with the COM port address in hexadecimal format like “!AA”.

Example1. Uses “:01TEST(Cr)” ([delimiter]AA[bypass data]) command to send “TEST(Cr)” to COM1, the response is prefixed with “!01”.



Example2. Reads the data in buffer of COM1, the response is prefixed with “!01”.



7.4.23. \$AAHV

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527, firmware version 3.08 or later

- **Description:** Reads/sets the status for bypassing data to COM2.

By default, the I-752N series module receives commands from the host and transfers these commands to serial device to obtain data. Sometimes, you may use devices like electronic scales, barcode scanners, which output data with needless of any request command. The "\$AAHV" command can be used to set the I-752N serial module to bypass the data from RS-232 ports to COM2 (to the host).

When you would like to use the function of bypass data to COM2, it is recommended to enable the function for all COM ports on the I-752N series module. Since RS-485 uses half-duplex communication, you can only connect one I-752N series module on a RS-485 network, and note that the host computer will not send data to the RS-485 network.

- **Syntax:** \$AAHV[chk](CrLf)

\$ Delimiter character

AA 2-character COM port address, ranged from 00 to FF in HEX format

H Command code

V=0 Disables the function of bypass data to COM2

V=1 Enables the function of bypass data to COM2

[chk] 2-character checksum. If the checksum is disabled -> no [chk]

(CrLf) End character

- **Response:** valid command: !AA[V][chk](CrLf)

invalid command:?[chk](CrLf)

no response: syntax error, communication error, or address error

! Delimiter character indicating a valid command

? Delimiter character indicating an invalid command

AA 2-character COM port address in HEX format

[chk] 2-character checksum. If the checksum is disabled -> no [chk]

(CrLf) End character

- **Example:** (takes I-7522 with module address 01 as an example)

Command: \$02H1(CrLf) Enables the mode of bypass data string to COM2

Response: !02(CrLf) Complete message from COM3 (address 02)

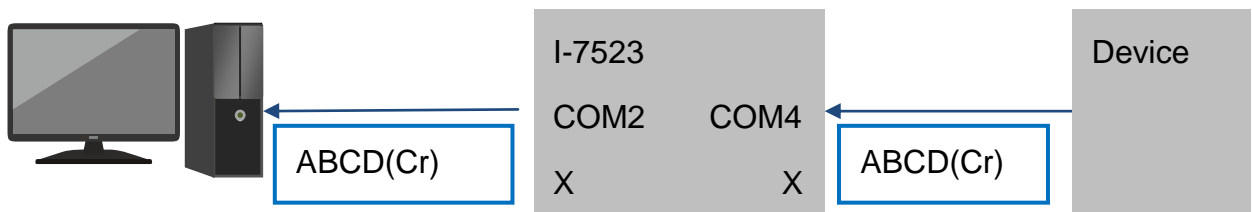
Command: \$02H(CrLf) Reads the mode of bypass data string to COM2

Response: !021(CrLf) Returns the mode is enabled.

● **Notes:**

1. If the function of bypass data to COM2 is used, it is recommend to enable the function for all the COM ports on the same I-752N series module.
2. To avoid data collision, it is best to use only one I-752N series module on the same RS-485 network.
3. Using the \$AAEV command (section 7.4.22) to enable the address prefix function for responses can help you to identify which COM port (device) each data comes from.

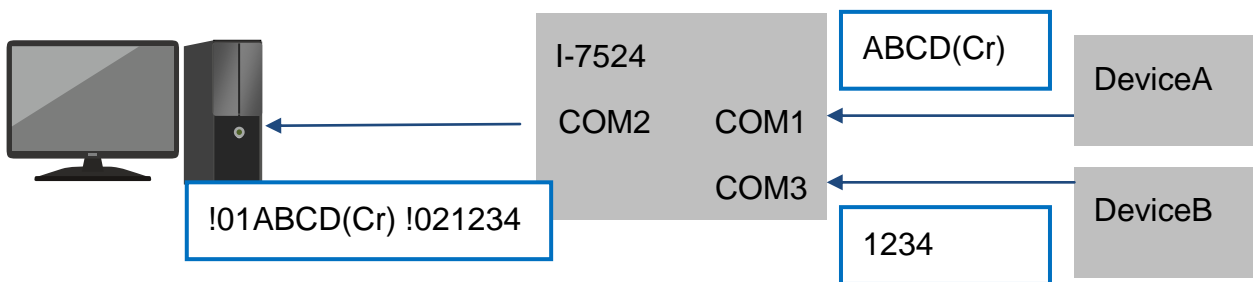
Example1. The host receives the device data from the I-7523 module without a request command when the bypass data function is enabled.



Example2. Using the \$AAEV command to enable the address prefix for responses, then you can easy to identify which COM port (device) sends the data.

- \$01E1(CrLf) Enables the address prefix for response from COM1
- \$01H1(CrLf) Enables the bypass data to COM2 function for COM1
- \$02E1(CrLf) Enables the address prefix for response from COM3
- \$02H1(CrLf) Enables the bypass data to COM2 function for COM3

The data transferred to the Host is prefixed with COM address like "!01" and "!02".



7.4.24. \$AAU

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527

- **Description:** Reads data from the RS-232 COM port buffer. No response if there is no data in the buffer.

When the I-752N series module receive the command ":AAxxxxxx", it will bypass the string "xxxxxx" to the specified COM port, waits for the response from connected device, and returns the response to the host. Any data that is not received in the waiting period will be placed in the COM port buffer. "\$AAU", "\$AAUR", "\$AAUA" commands are provided for reading the data in buffer.

- **Syntax:** \$AAU[chk](CrLf)
\$ Delimiter character
AA 2-character COM port address, ranged from 00 to FF in HEX format
U Command code
[chk] 2-character checksum. If the checksum is disabled -> no [chk]
(CrLf) End character

- **Response:** valid command: [data][chk](CrLf)
 invalid command: ?AA[chk](CrLf)
 no response: syntax error, communication error, or address error
? Delimiter character indicating an invalid command
AA 2-character COM port address in HEX format
[chk] 2-character checksum. If the checksum is disabled -> no [chk]
(CrLf) End character

- **Example:** (takes I-7523 with module address 01 as an example)
Command: \$01U(CrLf) Reads data from buffer of COM1
Response: data1(CrLf) Returns the data - data1
Command: \$02U(CrLf) Reads data from buffer of COM3
Response: No response There is no data in the buffer of COM3

- **Notes:**

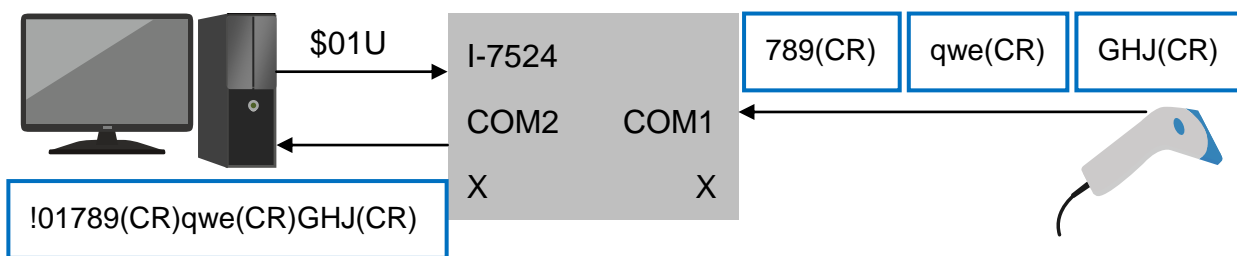
- (1) If the CrLfmode is set to 4 (no end character), all the data will be read back. If the CrLfmode is set to one in range of 0 to 3, the end character will be deleted from the string read.
- (2) When the connected RS-232 device is designed to use 0x0D as end character · the CrLfmode can be set to 0 or 4 only. Other settings may result in the inability to read back the complete data in buffer.
- (3) The data read by the \$AAU command will end at the first end character where the CrLfmode is set to 0/1/2/3. You may need to send \$AAU command several times to read all data in buffer.

- **Reading all data in buffer:** set CrLfmode for COM1 to 4 – no end character

In the scenario of

- (1) The barcode scanner scans three times and transfers the data with end character (CR): "789(CR)", "qwe(CR)", GHJ(CR)"
- (2) The CrLfmode is set to 4 for COM1 and COM2
- (3) The function of address prefix is enabled by the "\$01E1" command

When the host sends "\$01U" command to read data in the buffer of COM1, all the data is returned as "!01789(CR)qwe(CR)GHJ(CR)".

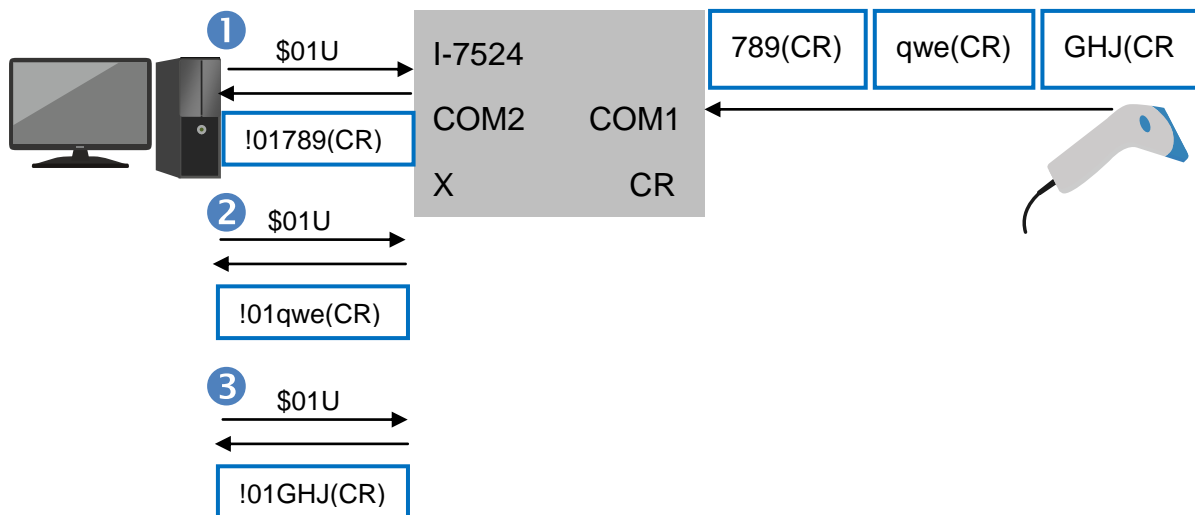


- **Reading one message at a time:** set CrLfmode for COM1 to 0 - CR end character

In the scenario of

- (1) The barcode scanner scans three times and transfers the data with end character (CR): "789(CR)", "qwe(CR)", GHJ(CR)"
- (2) The CrLfmode is set to 0 for COM1
- (3) The CrLfmode is set to 4 for COM2
- (4) The function of address prefix is enabled by the "\$01E1" command

Each time the host sends the "\$01U" command will read one message ended with (CR).



- ◇ \$AAUR and \$AAUA are new commands provided in the firmware version V4 and later. **\$AAUR** is similar to \$AAU but returns "N/A" if there is no data in the buffer. **\$AAUA** is used to read one message from buffer of every RS-232 port.

7.4.25. \$AAUR

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527, firmware version 4 and later

- **Description:** Reads data from RS-232 COM port buffer. "N/A" returned if no data in buffer. This command is valid when the minimum length for response in

7.4.29. \$AAUL[minlength] is set to 0.

- **Syntax:** \$AAUR[chk](CrLf)
 - \$ Delimiter character
 - AA 2-character COM port address, ranged from 00 to FF in HEX format
 - UR Command code
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Response:**
 - valid command: [data][chk](CrLf)
 - invalid command: ?AA[chk](CrLf)
 - no response: syntax error, communication error, or address error
 - ? Delimiter character indicating an invalid command
 - AA 2-character COM port address in HEX format
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Example:** (takes I-7523 with module address 01 as an example)
 - command: \$01UR(CrLf) Reads data in buffer of COM1
 - response: data1(CrLf) Returns the data in buffer
 - command: \$02UR(CrLf) Reads data in buffer of COM3
 - response: N/A Returns "N/A" if there is no data in buffer

7.4.26. \$AAUA

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527, firmware version 4 and later

- **Description:** Reads one message from buffer of every RS-232 port. The messages are arranged in the order of COM port number. (COM1, COM3, COM4 ...) One message is defined as a string received consecutively. If the time interval between 2 characters received exceeds 4 characters transmission time, the previous field will be divided to one message. The address parameter in command may be assigned to address for any com port.

- **Syntax:** \$AAUA[chk](CrLf)

\$ Delimiter character

AA 2-character COM port address, ranged from 00 to FF in HEX format

UA Command code

[chk] 2-character checksum. If the checksum is disabled -> no [chk]

(CrLf) End character

- **Response:** valid command: [data][chk](CrLf)

invalid command: ?AA[chk](CrLf)

no response: syntax error, communication error, or address error

? Delimiter character indicating an invalid command

AA 2-character COM port address in HEX format

[chk] 2-character checksum. If the checksum is disabled -> no [chk]

(CrLf) End character

- **Example:** (takes I-7523 with module address 01 as an example)

Command: \$01UA(CrLf) Reads one message from buffer of every COM port

Response: data1(CrLf) Returns the data - data1

Command: \$02UA(CrLf) Reads one message from buffer of every COM port

Response: data2(CrLf) Returns the data - data2

- **Notes:**

1. Messages are arranged in the order of COM port number. (COM1, COM3, COM4 ...)
2. It is unavailable to confirm if one or more COM port buffer has no data to return.
3. The command “**\$AAUS[seperator]**” (section 7.4.31) can be used to set a separator for data. Setting a separator like comma for messages can help you to make sure if one buffer has no data to return while two consecutive commas are read.

4. **“\$AAUL”** and **“\$AAUD”** commands (sections 7.4.29 and 7.4.30) are used to set the minimum length for a message and the pad character for filling a message to its minimum length.

In this way, you can easily identify one message from others by reading a fixed-length message from a fixed position in the string.

5. The address prefix setting by **“\$AAEV”** (section 7.4.22) command is not available for data read by **“\$AAUA”** command.

7.4.27. \$AAUC

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527, firmware version 4 and later

- **Description:** Clears data in COM port buffer
- **Syntax:** \$AAUC[chk](CrLf)
 - \$ Delimiter character
 - AA 2-character COM port address, ranged from 00 to FF in HEX format
 - UC Command code
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Response:** valid command: !AA[chk](CrLf)
 - invalid command: ?AA[chk](CrLf)
 - no response: syntax error, communication error, or address error
 - ! Delimiter character indicating a valid command
 - ? Delimiter character indicating an invalid command
 - AA 2-character COM port address in HEX format
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Example:** (takes I-7523 with module address 01 as an example)
 - Command: \$01UC(CrLf) Clears the data in buffer of COM1
 - Response: !01(CrLf) Complete message from COM1 (address 01)
 - Command: \$02UC(CrLf) Clears the data in buffer of COM3
 - Response: !02(CrLf) Complete message from COM3 (address 02)

7.4.28. \$AAUN

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527, firmware version 4 and later

- **Description:** Reads the message quantity in COM port buffer.
One message is defined as a string received consecutively. If the time interval between 2 characters received exceeds 4 characters transmission time, the previous field will be divided to one message.
- **Syntax:** \$AAUN[chk](CrLf)
 - \$ Delimiter character
 - AA 2-character COM port address, ranged from 00 to FF in HEX format
 - UN Command code
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Response:** valid command: !AAN[chk](CrLf)
invalid command: ?AA[chk](CrLf)
no response: syntax error, communication error, or address error
 - ! Delimiter character indicating a valid command
 - ? Delimiter character indicating an invalid command
 - AA 2-character COM port address in HEX format
 - N The message number in buffer
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Example:** (takes I-7523 with module address 01 as an example)
 - Command: \$01UN(CrLf) Reads the number of messages in buffer of COM1
 - Response: !012(CrLf) Returns the number is 2
 - Command: \$02UN(CrLf) Reads the number of messages in buffer of COM3
 - Response: !020(CrLf) Returns the number is 0

7.4.29. \$AAUL[minlength]

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527, firmware version 4 and later

- **Description:** Reads/sets the minimum length of a message returned to the host. If the length of a message is less than the minimum length requirement, the pad character set by \$AAUD[padchar] will be used to fill the message to meet its minimum length.

\$AAUL[chk](CrLf) – reads the minimum length setting

\$AAUL[minlength][chk](CrLf) – sets the minimum length for a message returned to host

- **Syntax:** \$AAUL[minlength] [chk](CrLf)

\$ Delimiter character

AA 2-character COM port address, ranged from 00 to FF in HEX format

UL Command code

[minlength] the minimum length of a message, a 3-digit decimal value ranged from 000 to 255. The default is 000 to disable the function.

[chk] 2-character checksum. If the checksum is disabled -> no [chk]

(CrLf) End character

- **Response:** valid command: !AA[minlength][chk](CrLf)
invalid command: ?AA[chk](CrLf)
no response: syntax error, communication error, or address error

! Delimiter character indicating a valid command

? Delimiter character indicating an invalid command

AA 2-character COM port address in HEX format

[chk] 2-character checksum. If the checksum is disabled -> no [chk]

(CrLf) End character

- **Example:** (takes I-7523 with module address 01 as an example)

Command: \$01UL(CrLf) Reads the min length of response set for COM1

Response: !01020(CrLf) Returns the min length is 20 characters

Command: \$02UL030(CrLf) Sets the min length for COM3 to 30 characters

Response: !02(CrLf) Complete message from COM3 (address 02)

7.4.30. \$AAUD[padchar]

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527, firmware version 4 and later

- **Description:** Reads/sets the pad character for filling up a message to its minimum length (set by command \$AAUL).

\$AAUD[chk](CrLf) - reads the pad character

\$AAUD[padchar][chk](CrLf) - sets the pad character

- **Syntax:** \$AAUD[padchar] [chk](CrLf)

\$ Delimiter character

AA 2-character COM port address, ranged from 00 to FF in HEX format

UD Command code

[padchar] the pad character represented by 2-character ASCII code in hexadecimal format. The default is 0x20, a space character.

[chk] 2-character checksum. If the checksum is disabled -> no [chk]

(CrLf) End character

- **Response:** valid command: !AA[padchar][chk](CrLf)
invalid command: ?AA[chk](CrLf)
no response: syntax error, communication error, or address error

! Delimiter character indicating a valid command

? Delimiter character indicating an invalid command

AA 2-character COM port address in HEX format

[chk] 2-character checksum. If the checksum is disabled -> no [chk]

(CrLf) End character

- **Example:** (takes I-7523 with module address 01 as an example)

Command: \$01UD(CrLf) Reads the pad character for COM1

Response: !012A(CrLf) Returns the pad character for COM1 is ""

Command: \$02UD2A(CrLf) Sets the pad character for COM3 to ""

Response: !02(CrLf) Complete message from COM3 (address 02)

7.4.31. \$AAUS[seperator]

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527, firmware version 4 and later

- **Description:** Reads/sets the seperator character for reading one message from buffer of every COM port, the response of \$AAUA command. Address in this command may be assigned to any one of COM port address.

\$AAUS[chk](CrLf) - reads the seperator character

\$AAUSN[seperator][chk](CrLf) - sets the seperator character

- **Syntax:** \$AAUSN[seperator] [chk](CrLf)
 - \$ Delimiter character
 - AA 2-character COM port address, ranged from 00 to FF in HEX format
 - US Command code
 - N 0 = mode0, no seperator used (the default value)
 - 1 = mode1, uses the end character set by CrLfmode (section 7.4.9. AATN)
 - 2 = mode2, uses one byte seperator character
 - 3 = mode3, uses two bytes seperator character
 - [seperator] the seperator is represented by 2 characters (while N is set to 2) or 4 characters (while N is set to 3) ASCII code in hexadecimal format.
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Response:** valid command: !AAN[seperator][chk](CrLf)
 - invalid command: ?AA[chk](CrLf)
 - no response: syntax error, communication error, or address error
 - ! Delimiter character indicating a valid command
 - ? Delimiter character indicating an invalid command
 - AA 2-character COM port address in HEX format
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Example:** (takes I-7523 with module address 01 as an example)
 - Command: \$01US(CrLf) Reads the seperator character
 - Response: !010(CrLf) Returns there is no delimiter used.
 - Command: \$01US22C(CrLf) Sets the seperator character to “,”
 - Response: !01(CrLf) Complete message from COM1 (address 01)

Command:	\$01US32A2A(CrLf)	Sets the seperator character to "***"
Response:	!01(CrLf)	Complete message from COM1 (address 01)
Command:	\$01US (CrLf)	Reads the seperator character
Response:	!0132A2A(CrLf)	Returns the delimiter is "***"

7.4.32. \$AAN[buffermode]

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527, firmware version 4 and later

- **Description:** Reads/sets the usage mode for buffer of RS-232 port.
\$AAN[chk](CrLf) - reads the usage mode for buffer.
\$AAN[buffermode][chk](CrLf) - sets the usage mode for buffer.
- **Syntax:** \$AAN[buffermode] [chk](CrLf)
\$ Delimiter character
AA 2-character COM port address, ranged from 00 to FF in HEX format
N Command code
[buffermode] 0 = Continuously writes data to the buffer, and stops while the buffer is full (default)
 1 = Keeps the latest message only
[chk] 2-character checksum. If the checksum is disabled -> no [chk]
(CrLf) End character
- **Response:** valid command: !AA[buffermode][chk](CrLf)
 invalid command: ?AA[chk](CrLf)
 no response: syntax error, communication error, or address error
! Delimiter character indicating a valid command
? Delimiter character indicating an invalid command
AA 2-character COM port address in HEX format
[chk] 2-character checksum. If the checksum is disabled -> no [chk]
(CrLf) End character
- **Example:** (takes I-7523 with module address 01 as an example)
Command: \$01N(CrLf) Reads the buffer usage mode for COM1
Response: !010(CrLf) Returns the mode is stop writing if the buffer is full
Command: \$01N1(CrLf) Set the mode to 1 - keep the latest message only
Response: !01(CrLf) Complete message from COM1 (address 01)

7.4.33. \$AAS[lastdatahdl]

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527, firmware version 4 and later

- **Description:** Reads/sets the retention mode of the latest message in buffer.

\$AAS[chk](CrLf) - reads the retention mode of the latest message in buffer

\$AAS[lastdatahdl][chk](CrLf) - sets the retention mode of the latest message in buffer

- **Syntax:** \$AAN[lastdatahdl] [chk](CrLf)

\$ Delimiter character

AA 2-character COM port address, ranged from 00 to FF in HEX format

S Command code

[buffermode] 0 = Clears the buffer when the last message is read (default)

1 = Keeps the last message in buffer and replies it to the reading command every time.

[chk] 2-character checksum. If the checksum is disabled -> no [chk]

(CrLf) End character

- **Response:** valid command: !AA[lastdatahdl][chk](CrLf)

invalid command: ?AA[chk](CrLf)

no response: syntax error, communication error, or address error

! Delimiter character indicating a valid command

? Delimiter character indicating an invalid command

AA 2-character COM port address in HEX format

[chk] 2-character checksum. If the checksum is disabled -> no [chk]

(CrLf) End character

- **Example:** (takes I-7523 with module address 01 as an example)

command: \$01S(CrLf) Reads the retention mode for buffer of COM1

response: !010(CrLf) Returns the retention mode is clearing the buffer while the last message is read.

command: \$01S1(CrLf) Sets the buffer to keep the last message if no new data arrives

response: !01(CrLf) Complete message from COM1 (address 01)

7.4.34. DI/DO Data Bit Mapping

The data bit used to represent channel number varies from module to module. Refer to the following tables to confirm the data bit information for accessing status of DI/DO channels.

The corresponding table for DI channel number and the reading value

	Bit0	Bit1	Bit2	Bit3	Bit4
I-7521	-	DI2	DI3	-	-
I-7522	-	DI2	DI3	-	-
I-7523	-	DI2	-	-	-
I-7522A	DI	DI1	DI2	DI3	DI4
I-7524	DI	-	-	-	-
I-7527	DI	-	-	-	-

The corresponding table for DO channel number and the setting value

	Bit0	Bit1	Bit2	Bit3	Bit4
I-7521	DO1	DO2	DO3	-	-
I-7522	DO1	-	-	-	-
I-7523	-	-	-	-	-
I-7522A	DO	DO1	DO2	DO3	DO4
I-7524	DO	-	-	-	-
I-7527	DO	-	-	-	-

7.4.35. \$AAYN

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527

- **Description:** Reads one channel of the onboard DIs
- **Syntax:** \$AAYN[chk](CrLf)
 - \$ Delimiter character
 - AA 2-character module address, ranged from 00 to FF in HEX format
 - Y Command code
 - N 1 = reads Bit0 · 2 = reads Bit1 · 3 = reads Bit2 ·
4 = reads Bit3 · 5 = reads Bit4
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Response:** valid command: !AAS[chk](CrLf)
invalid command: ?AA[chk](CrLf)
no response: syntax error, communication error, or address error
 - ! Delimiter character indicating a valid command
 - ? Delimiter character indicating an invalid command
 - AA 2-character module address in HEX format
 - S=0 The DI status is low
 - S=1 The DI status is high (If the DI wiring is floating, the status is high, too.)
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Example:** (takes I-7521 with module address 01 as an example)
 - Command: \$01Y2(CrLf) Reads the DI2 (Bit1) status
 - Response: !011(CrLf) Returns the DI2 status is high
 - Command: \$01Y3(CrLf) Read the DI3 (Bit2) status
 - Response: !010(CrLf) Returns the DI3 status is low

7.4.36. \$AAZNV

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527

- **Description:** Reads/sets one channel of the onboard DOs
\$AAZN[chk](CrLf) – reads one DO channel status
\$AAZNV[chk](CrLf) - sets one DO channel status
- **Syntax:** \$AAZNV[chk](CrLf)
 - \$ Delimiter character
 - AA 2-character module address, ranged from 00 to FF in HEX format
 - Z Command code
 - N 1= DO1 (Bit0) · 2= DO2 (Bit1) · 3= DO3 (Bit2) ·
4= DO4 (Bit3) · 5= DO5 (Bit4)
 - V=0 Sets DO status to OFF
 - V=1 Sets DO status to ON
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Response:** valid command: !AAS[chk](CrLf)
invalid command: ?AA[chk](CrLf)
no response: syntax error, communication error, or address error
 - ! Delimiter character indicating a valid command
 - ? Delimiter character indicating an invalid command
 - AA 2-character module address in HEX format
 - S=0 The DO status is off
 - S=1 The DO status is on
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Example:** (takes I-7521 with module address 01 as an example)
 - Command: \$01Z10(CrLf) Sets the DO1 status to off
 - Response: !01(CrLf) Complete message from module address 01
 - Command: \$01Z3(CrLf) Reads the DO3 status
 - Response: !010(CrLf) Returns the DO3 status is off

- **Notes:**

If the status of Host Watchdog timeout is not cleared, the DO command will be ignored, and an exclamation mark “!” will be returned.

7.4.37. #**

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527

- **Description:** Simultaneous sampling of DI channels on multiple modules.
When “#**” command is sent to the RS-485 network, all I-752N series modules on the same RS-485 network will read the DI status at the same time, and then using “\$AA4” command to read the synchronized DI data from every module.
- **Syntax:** #**[chk](CrLf)
 - # Delimiter character
 - ** Command code
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Response:** no response
- **Example:** (takes 3 I-7521 modules with address 01, 02, 03 as examples)

Command:	#** (CrLf)	Sends a broadcast command to synchronize sampling of DI signals on multiple I-752N modules.
Response:	No response	No response expected
Command:	014(CrLf)	Reads the synchronized DI from I-7521 with address 01
Response:	!0117(CrLf)	Returns the DI2 and DI3 on the I-7521 are both 1 (high)
Command:	024(CrLf)	Reads the synchronized DI from I-7521 with address 02
Response:	!0213(CrLf)	Returns the DI2 is 1 (high) and DI3 is 0 (low)
Command:	!034(CrLf)	Reads the synchronized DI from I-7521 with address 03
Response:	!0311(CrLf)	Returns the DI2 and DI3 are both 0 (low)

7.4.38. \$AA4

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527

- **Description:** Reads the synchronized DI data
- **Syntax:** \$AA4[chk](CrLf)
 - \$ Delimiter character
 - AA 2-character module address, ranged from 00 to FF in HEX format
 - 4 Command code
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Response:** valid command: !AASV[chk](CrLf)
 - invalid command: ?AA[chk](CrLf)
 - no response: syntax error, communication error, or address error
 - ! Delimiter character indicating a valid command
 - ? Delimiter character indicating an invalid command
 - AA 2-character module address in HEX format
 - S=1 Indicates that it is the first reading since the “#**” command is sent
 - S=0 Indicates that it is not the first reading since the “#**” command is sent
 - V Bit0 = DI1 · Bit1 = DI2 · Bit2 = DI3 · Bit3 = DI4 · Bit4 = DI5
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Example:** (takes 3 I-7521 modules with address 01, 02, 03 as examples)

Command: #** (CrLf)	Sends a broadcast command to synchronize sampling of DI signals on multiple I-752N modules.
Response: No response	No response expected
Command: 014(CrLf)	Reads the synchronized DI from I-7521 with address 01
Response: !0117(CrLf)	Returns the DI2 and DI3 on the I-7521 are both 1 (high)
Command: 024(CrLf)	Reads the synchronized DI from I-7521 with address 02
Response: !0213(CrLf)	Returns the DI2 is 1 (high) and DI3 is 0 (low)
Command: 034(CrLf)	Reads the synchronized DI from I-7521 with address 02
Response: !0311(CrLf)	Returns the DI2 and DI3 are both 0 (low)

7.4.39. \$AAL[data]

Available for I-7522A, firmware version 3.01 and later

- **Description:** Sets DO-1/2/3/4 on the I-7522A module. (on expansion board)
- **Syntax:** \$AALbbbb[chk](CrLf): sets 4 channels with ON/OFF status for each
 \$AALcb[chk](CrLf): set one channel status by specifying the channel number and its ON/OFF status
 \$AALh[chk](CrLf): set 4 channels with one hexadecimal character

\$ Delimiter character

AA 2-character module address, ranged from 00 to FF in HEX format

L Command code

Value	0	1	2	3
b	Set DO to OFF	Set DO to ON		
c	DO1 (pin25)	DO2 (pin26)	DO3 (pin27)	DO4 (pin28)

h Sets the state of DO1 ~ DO4 with one hexadecimal numbers, valid values are 0 ~ 9, a~f, A~F.

bit0= DO1, bit1= DO2, bit2= DO3, bit3 = DO4.

1= ON, 0= OFF.

[chk] 2-character checksum. If the checksum is disabled -> no [chk]

(CrLf) End character

- **Response:** valid command: !AA[chk](CrLf)
 invalid command: ?AA[chk](CrLf)
 no response: syntax error, communication error, or address error

! Delimiter character indicating a valid command

? Delimiter character indicating an invalid command

AA 2-character module address in HEX format

[chk] 2-character checksum. If the checksum is disabled -> no [chk]

(CrLf) End character

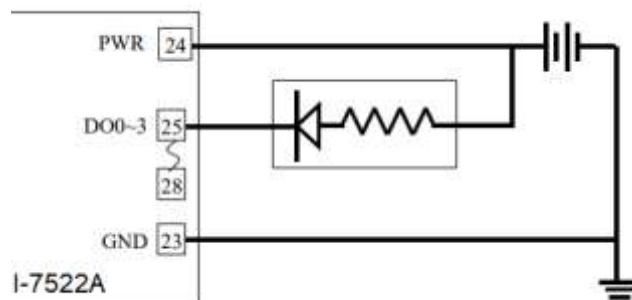
- **Example:** (takes I-7522A with address 01 as an example)

Command:	\$01L1000(CrLf)	Sets DO4= ON · DO3= DO2= DO1= OFF
Response:	!01(CrLf)	Complete message from module address 01
Command:	\$01L21(CrLf)	Sets to ON, no change for other channels
Response:	!01(CrLf)	Complete message from module address 01
Command:	\$01L30 (CrLf)	Sets DO4 to OFF, no change for other channels
Response:	!01(CrLf)	Complete message from module address 01
Command:	\$01LE (CrLf)	Sets DO4, DO3, DO2 to ON, and DO1 to off.
Response:	!01(CrLf)	Complete message from module address 01

- **Notes:**

(1) If the status of Host Watchdog timeout is not cleared, the DO command will be ignored, and an exclamation mark “!” will be returned.

(2) DO wiring diagram for I-7522A



7.4.40. \$AAR

Available for I-7522A, firmware version 3.01 and later

- **Description:** Reads DI-1/2/3/4 on the I-7522A module. (on expansion board)

- **Syntax:** \$AAR[chk](CrLf)

\$ Delimiter character

AA 2-character module address, ranged from 00 to FF in HEX format

R Command code

[chk] 2-character checksum. If the checksum is disabled -> no [chk]

(CrLf) End character

- **Response:** valid command: !AAS[chk](CrLf)
invalid command: ?AA[chk](CrLf)
no response: syntax error, communication error, or address error

! Delimiter character indicating a valid command

? Delimiter character indicating an invalid command

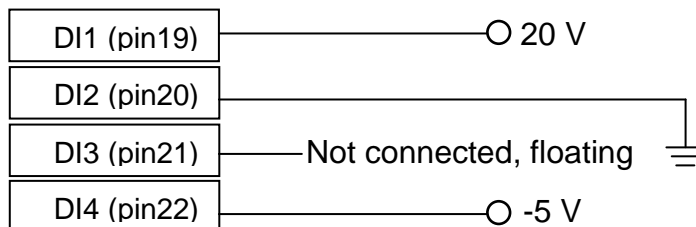
AA 2-character module address in HEX format

S The state of DI1 ~ DI4, indicated by one hexadecimal character from 0 to F
bit0= DI1 (pin19), bit1= DI2 (pin20), bit2= DI3 (pin21), bit3 = DI4 (pin 22)
1= high, 0= low. (The status will be high if the DI wiring is floating)

[chk] 2-character checksum. If the checksum is disabled -> no [chk]

(CrLf) End character

- **Example:** (takes I-7522A with module address 01 as an example)



command: \$01R(CrLf)

Reads DI status from module address 01

response: !015(CrLf)

Returns the DI status is 5 (0101), meaning that
DI1 = DI3= 1 (high) · DI2 = DI4= 0 (low)

7.4.41. @AA[data]

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527, firmware version 3.0 and later

- **Description:** Reads/sets the status of onboard DI/DO-1/2/3/4/5
 - @AA[chk](CrLf) – reads the status of DO and DI channels
 - @AAh[chk](CrLf) –sets the onboard DO 1/2/3 on the I-7521/7522/7523
 - @AAhh[chk](CrLf) –sets the onboard DO 1/2/3/4/5 on the 7522A/7524/7527
- **Syntax:** @AA[chk](CrLf)
 - @AAh[chk](CrLf)
 - @AAhh[chk](CrLf)

@ Delimiter character

AA 2-character module address, ranged from 00 to FF in HEX format

h 4-bit DO value in hexadecimal format, valid values are 0 ~ 9, a~f, A~F
bit0= DO1, bit1= DO2, bit2= DO3, bit3 = DO4.
1= ON, 0= OFF.

hh 5-bit DO value in HEX format, valid values are 00 ~ 1F
bit0= DO1, bit1= DO2, ... bit4 = DO5 ° 1= ON, 0= OFF °

[chk] 2-character checksum. If the checksum is disabled -> no [chk]

(CrLf) End character
- **Response:**
 - @AA valid command: >AAbbcc[chk](CrLf)
invalid command:?[chk](CrLf)
no response: syntax error, communication error, or address error
 - > Delimiter character indicating a valid command
 - ? Delimiter character indicating an invalid command
 - AA 2-character module address in HEX format
 - bb 2-character DO status in HEX format, valid values are 00 ~ 1F
bit0= DO1, bit1= DO2, ... bit4 = DO5 ° 1= ON, 0= OFF
 - cc 2-character DI status in HEX format, valid values are 00 ~ 1F
bit0= DI1, bit1= DI2, ,, , bit4 = DI5 ° 1=High, 0= Low
(The status will be high if the DI wiring is floating)
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character

@AAh/hh valid command: >[chk](CrLf)
invalid command:?[chk](CrLf)
no response: syntax error, communication error, or address error

> Delimiter character indicating a valid command
? Delimiter character indicating an invalid command

[chk] 2-character checksum. If the checksum is disabled -> no [chk]
(CrLf) End character

- **Example:**

Takes I-7522A with module address 01 as an example:

Command:	@0108(CrLf)	Sets DO1, DO2, DO3, DO5 to OFF, DO4 to ON
Response:	>(CrLf)	Complete message from module
Command:	@01(CrLf)	Reads the DI and DO status from module address 01
Response:	>01081F(CrLf)	Returns DO1, DO2, DO3 and DO5 are OFF, DO4 is ON, DI1, DI2, DI3, DI4, DI5 are high

Takes I-7521 with module address 01 as an example:

Command:	\$017 (CrLf)	Sets DO1, DO2 and DO3 to ON
Response:	>(CrLf)	Complete message from module

- **Notes:**

If the status of Host Watchdog timeout is not cleared, the DO command will be ignored, and an exclamation mark “!” will be returned.

7.4.42. #AABBHH

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527, firmware version 3.0 and later

- **Description:** Sets multiple DO channels (onboard)
- **Syntax:** #AABBHH[chk](CrLf)
 - # Delimiter character
 - AA 2-character module address, ranged from 00 to FF in HEX format
 - BB 00/0A
 - HH 2-character DO value in HEX format, valid values are 00 ~ 1F
bit0= DO1, bit1= DO2, ... bit4 = DO5 · 1= ON, 0= OFF
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Response:**
 - valid command: >[chk](CrLf)
 - invalid command:?[chk](CrLf)
 - no response: syntax error, communication error, or address error
 - > Delimiter character indicating a valid command
 - ? Delimiter character indicating an invalid command
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Example:** (takes I-7521 with module address 01 as an example)
 - Command: #010003(CrLf) Sets DO1, DO2 to ON, DO3 to OFF
 - Response: >(CrLf) Complete message from module
 - Command: #010A02(CrLf) Sets DO1, DO3 to OFF, DO2 to ON
 - Response: >(CrLf) Complete message from module

- **Notes:**

If the status of Host Watchdog timeout is not cleared, the DO command will be ignored, and an exclamation mark “!” will be returned.

7.4.43. #AABCDD

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527, firmware version 3.0 and later

- **Description:** Sets one DO channel
- **Syntax:** #AABCDD [chk](CrLf)
 - # Delimiter character
 - AA 2-character module address, ranged from 00 to FF in HEX format
 - B 1 or A
 - C 0= DO1 · 1= DO2 · 2= DO3 · 3= DO4 · 4= DO5
 - DD 00= OFF · 01= ON
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Response:** valid command: >[chk](CrLf)
 - invalid command:?[chk](CrLf)
 - no response: syntax error, communication error, or address error
 - > Delimiter character indicating a valid command
 - ? Delimiter character indicating an invalid command
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Example:** (takes I-7521 with module address 01 as an example)
 - Command: #011201(CrLf) Sets DO3 to ON
 - Response: >(CrLf) Complete message from module
 - Command: #01A000(CrLf) Sets DO1 to ON
 - Response: >(CrLf) Complete message from module

- **Notes:**

If the status of Host Watchdog timeout is not cleared, the DO command will be ignored, and an exclamation mark “!” will be returned.

7.4.44. ~**

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527

- **Description:** Broadcasts the “Host is OK” message for the modules on the RS-485 network. The module with host watchdog enabled will reset the timer after receiving the message.

- **Syntax:** ~**[chk](CrLf)

~ Delimiter character

** Command code

[chk] 2-character checksum. If the checksum is disabled -> no [chk]

(CrLf) End character

- **Response:** No response expected

- **Example:**

(The broadcast command is valid for all Host Watchdog-enabled modules on the RS-485 network)

command: ~**(CrLf) Sends a broadcast message “Host OK”

response: no response

7.4.45. ~AA0

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527

- **Description:** Reads the status of Host Watchdog. 0x04 means that the Host Watchdog timeout event has occurred, and all DO/AO output commands will not be executed, until the timeout status is cleared by “~AA1”command.

- **Syntax:** ~AA0[chk](CrLf)

~ Delimiter character

AA 2-character module address, ranged from 00 to FF in HEX format

0 Command code

[chk] 2-character checksum. If the checksum is disabled -> no [chk]

(CrLf) End character

- **Response:** valid command: !AASS[chk](CrLf)

invalid command: ?AA[chk](CrLf)

no response: syntax error, communication error, or address error

! Delimiter character indicating a valid command

? Delimiter character indicating an invalid command

AA 2-character module address in HEX format

SS 2-character host watchdog status in HEX format.

Bit_0/ Bit_1: reserved

Bit_2: 0= OK, 1= Host watchdog timeout occurred

Bit_7: 1=Host watchdog enabled

[chk] 2-character checksum. If the checksum is disabled -> no [chk]

(CrLf) End character

- **Example:**

Command: ~010(CrLf) Reads the Host Watchdog status from module 01

Response: !0100(CrLf) Returns the Host Watchdog is disabled

Command: ~020(CrLf) Reads the Host Watchdog status from module 02

Response: !0204(CrLf) Returns the module is in alarm status due to Host Watchdog timeout occurred. .

Command: ~020(CrLf) Reads the Host Watchdog status from module 02

Response: !0280(CrLf) Returns the Host Watchdog is enabled and no watchdog timeout occurred

7.4.46. ~AA1

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527

- **Description:** Clears the Host Watchdog timeout status. All DO/AO output commands will not be executed if the Host Watchdog timeout has occurred, until the timeout status is cleared by this command.
- **Syntax:** ~AA1[chk](CrLf)
 - ~ Delimiter character
 - AA 2-character module address, ranged from 00 to FF in HEX format
 - 1 Command code
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Response:** valid command: !AA[chk](CrLf)
invalid command: ?AA[chk](CrLf)
no response: syntax error, communication error, or address error
 - ! Delimiter character indicating a valid command
 - ? Delimiter character indicating an invalid command
 - AA 2-character module address in HEX format
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Example:** (takes I-7521 with module address 01 as an example)

Command:	~010(CrLf)	Reads the Host Watchdog status of module address
Response:	!0104(CrLf)	Returns the Host Watchdog timeout occurred
Command:	\$01Z11 (CrLf)	Sets DO1 on the module with address 01 to ON
Response:	!(CrLf)	The output command is ignored due to the Host Watchdog timeout status is not cleared yet.
Command:	~011(CrLf)	Clears the Host Watchdog timeout of module address 01
Response:	!01(CrLf)	Complete message from module address01
Command:	~010(CrLf)	Reads the Host Watchdog status of module address 01
Response:	!0100(CrLf)	Returns the Host Watchdog status is 00, the timeout status is cleared
Command:	\$01Z11 (CrLf)	Sets DO1 on the module with address 01 to ON
Response:	!01(CrLf)	Complete message from module address01

7.4.47. ~AA2

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527

- **Description:** Reads the Host Watchdog status and timeout setting
- **Syntax:** ~AA2[chk](CrLf)
 - ~ Delimiter character
 - AA 2-character module address, ranged from 00 to FF in HEX format
 - 2 Command code
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Response:** valid command: !AASTT[chk](CrLf)
 - invalid command: ?AA[chk](CrLf)
 - no response: syntax error, communication error, or address error
 - ! Delimiter character indicating a valid command
 - ? Delimiter character indicating an invalid command
 - AA 2-character module address in HEX format
 - S=0 Host Watchdog is disabled
 - S=1 Host Watchdog is enabled
 - TT 2-character timeout value in HEX format, valid values are 00 ~ FF (0,1sec)
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Example:** (takes I-7521 with module address 01 as an example)
 - Command: ~012(CrLf) Reads the Host Watchdog settings for module address01
 - Response: !01000(CrLf) Returns the Host Watchdog is disabled and timer is set to 0 second for module with address 01
 - Command: ~022(CrLf) Reads the Host Watchdog settings for module address02
 - Response: !02164(CrLf) Returns that the Host Watchdog is enabled, and the watchdog timer is set to 64 in HEX, or 100 in decimal, equal to 100 * 0.1 second = 10 seconds.

7.4.48. ~AA3ETT

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527

- **Description:** Sets the Host Watchdog including the timeout value
- **Syntax:** ~AA3ETT[chk](CrLf)
 - ~ Delimiter character
 - AA 2-character module address, ranged from 00 to FF in HEX format
 - 3 Command code
 - E=0 Disables the Host Watchdog
 - E=1 Enables the Host Watchdog
 - TT 2-character timeout value in HEX format, valid values are 00 ~ FF (0,1sec)
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Response:** valid command: !AASTT[chk](CrLf)
 - invalid command: ?AA[chk](CrLf)
 - no response: syntax error, communication error, or address error
 - ! Delimiter character indicating a valid command
 - ? Delimiter character indicating an invalid command
 - AA 2-character module address in HEX format
 - S=0 Host Watchdog is disabled
 - S=1 Host Watchdog is enabled
 - TT 2-character timeout value in HEX format, valid values are 00 ~ FF (0,1sec)
 - [chk] 2-character checksum. If the checksum is disabled -> no [chk]
 - (CrLf) End character
- **Example:** (takes two I-7523 modules with address 01 and 02 as examples)
 - Command: ~013000(CrLf) Disables the Host Watchdog and sets timer to 0 sec.
 - Response: !01000(CrLf) Complete message from module address01, the Host Watchdog is disabled and timer is set to 0 sec.
 - Command: ~023164(CrLf) Enables the Host Watchdog and sets timer to 10 sec.
 - Response: !02164(CrLf) Complete message from module address01, the Host Watchdog is enabled and timer is set to 10 sec.

7.4.49. ~AA4P / ~AA4S

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527

- **Description:** Reads the power-on value and safe value for DO channels. Power-on value and safe value are designed for avoiding device failure by incorrect output commands.
 - (1) All DO and AO channels will output their power-on values when the module is powered on or reset by the module watchdog. At this point the output commands can be used to change the status of channels.
 - (2) All DO and AO channels will output their safe values since the Host Watchdog timeout is occurred. The output commands will not be executed until the timeout status is cleared.

~AA4P[chk](CrLf) - reads power-on value for DO channels

~AA4S[chk](CrLf) - reads safe value for DO channels

- **Syntax:** ~AA4P[chk](CrLf)/ ~AA4S[chk](CrLf)

~ Delimiter character

AA 2-character module address, ranged from 00 to FF in HEX format

4 Command code

P Reads the power-on value for DO channels

S Read the safe value for DO channels

[chk] 2-character checksum. If the checksum is disabled -> no [chk]

(CrLf) End character

- **Response:** valid command: !AAV[chk](CrLf)

invalid command: ?AA[chk](CrLf)

no response: syntax error, communication error, or address error

! Delimiter character indicating a valid command

? Delimiter character indicating an invalid command

AA 2-character module address in HEX format

V Bit0 = DO1 \ Bit1 = DO2 \ Bit2 = DO3 \ Bit3 = DO4 \ Bit4 = DO5

1= ON \ 0= OFF

[chk] 2-character checksum. If the checksum is disabled -> no [chk]

(CrLf) End character

- **Example:** (takes I-7521 with module address 01 as an example)

Command: ~014P(CrLf) Reads the power-on value set for module address 01

Response: !017(CrLf) Returns the power-on value is DO1=DO2=DO3=1

Command: ~014S(CrLf) Reads the safe value set for module address 01

Response: !010(CrLf) Returns the safe value is DO1=DO2=DO3=0

7.4.50. ~AA5P / ~AA5S

Available for I-7521/I-7522/I-7522A/I-7523/I-7524/I-7527

- **Description:** Sets the power-on value and safe value for DO channels.

This function sets the current state of DO channels as a power-on/safe value. Power-on and Safe values are used to keep the state of output channels in the pre-set state when the module or host fails to function properly.

- (1) All DO and AO channels will output their power-on values when the module is powered on or reset by the module watchdog. At this point the output commands can be used to change the status of channels.
- (2) All DO and AO channels will output their safe values since the Host Watchdog timeout is occurred. The output commands will not be executed until the timeout status is cleared.
- (3) Sets the DO channels to the status of power-on value or safe value, and then sends “~AA4P” or “~AA4S” to complete setting.

~AA5P[chk](CrLf) - sets the power-on value for DO channels

~AA5S[chk](CrLf) – sets the safe value for DO channels

- **Syntax:** ~AA5P[chk](CrLf)/ ~AA5S[chk](CrLf)

~ Delimiter character

AA 2-character module address, ranged from 00 to FF in HEX format

5 Command code

P Sets the power-on value for DO channels

S Sets the safe value for DO channels

[chk] 2-character checksum. If the checksum is disabled -> no [chk]

(CrLf) End character

- **Response:** valid command: !AAV[chk](CrLf)

invalid command: ?AA[chk](CrLf)

no response: syntax error, communication error, or address error

! Delimiter character indicating a valid command

? Delimiter character indicating an invalid command

AA 2-character module address in HEX format

V Bit0 = DO1 \ Bit1 = DO2 \ Bit2 = DO3 \ Bit3 = DO4 \ Bit4 = DO5

1= ON \ 0= OFF

[chk] 2-character checksum. If the checksum is disabled -> no [chk]

(CrLf) End character

- **Example:** (takes I-7521 with module address 01 as an example)

Command:	#010007(CrLf)	Sets DO1=DO2=DO3=1
Response:	>(CrLf)	Complete message from module
Command:	~015P(CrLf)	Sets the power-on value to current DO status
Response:	!017(CrLf)	Returns the power-on value is set to DO1=DO2=DO3=1
Command:	#010000(CrLf)	Sets DO1=DO2=DO3=0
Response:	>(CrLf)	Complete message from module
Command:	~015S(CrLf)	Sets the safe value to current DO status
Response:	!010(CrLf)	Returns the safe value is set to DO1=DO2=DO3=0

Notes:

If the status of Host Watchdog timeout is not cleared, the DO command will be ignored, and an exclamation mark “!” will be returned.

Revision History

Revision	Date	Description
3.0.0	2022/08	Added description for new functions in firmware version 4.x.