EX952N Series (Addressable RS422/485 to RS232 converter)

EX952N Quick Manual

Introduction

EX952N is an Addressable RS422/485 to RS232 converter that it is designed to connect RS422/485 devises to an RS232 network.

On most computer systems, industrial device, PLCS, measurement equipment are unbalanced transmission for communication distance, transmission speed, network capability. The EX952N addressable converter solves the problem and let you easily build up an RS485 network with your RS232 devices.

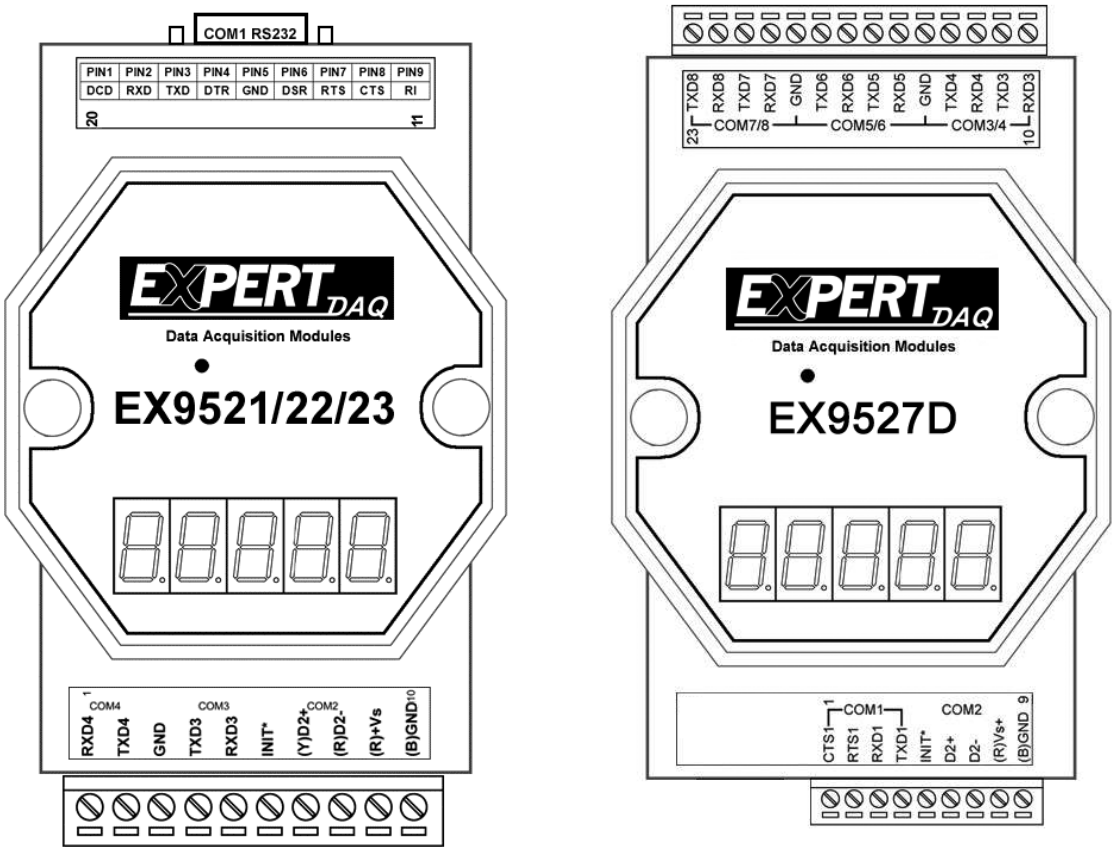
Features & Spec.

- Built in "Addressable RS485 to RS232 Converter" firmware
- Watchdog timer provides fault tolerance and recovery
- CPU 80188, 40MHZ
  - SRAM 256KB
  - FLASH ROM 512KB
  - EEPROM 2KB
  - EMBEDDED OS RomDos(Datalight)
  - Communication speed: 115.2K bps max
  - Operating temperature: -25℃ to +75℃
  - Storage temperature: -40℃ to +80℃
  - Power requirement: Unregulated 10~30 VDC power
- Series EX9521D, EX9522D, EX9523D, EX9527D

Model No.	EX9521D	EX9522D	EX9523D	EX9527D
COM1	RS232(5wire)	RS232(5wire)	RS232(5wire)	RS232(5wire)
COM2	RS485	RS485	RS485	RS485
COM3	x	RS232(3wire)	RS232(3wire)	RS232(3wire)
COM4	X	X	RS232(3wire)	RS232(3wire)
COM5	X	X	X	RS232(3wire)
COM6	X	X	X	RS232(3wire)
COM7	X	X	X	RS232(3wire)
COM8	X	X	X	RS232(3wire)

COM4:Program download RS232 (3wire) EX9521D/22D/23D.

COM1:Program download RS232 (5wire) EX9527D.



Pin assignment of EX9521/22/23's COM1 connector (DB-9 Male):

Pin	Name	Description
1	DCD	Data Carrier Detect
2	RXD	Receive Data
3	TXD	Transmit Data
4	DTR	Data Terminal Ready
5	GND	Signal ground
6	DSR	Data Set Ready
7	RTS	Request To Send
8	CTS	Clear To Send
9	RI	Ring Indicator

Use COM4(EX9521/22/23)/COM1(EX9527) for downloading program

Power on the system when EX952N module's INIT* pin is wired to ground and COM4(EX9521/22/23)/COM1(EX9527) is connected to PC. The disk image can be downloaded from PC to flash ROM of 952N module under Hyper-Terminal (Com port setting: baudrate **57600**, Databit **8**, Parity **None**, Stopbit**1,** Flow control **None**) by clicking "transfer", "receive file", then choose Xmodem as the protocol and key in the file name and path. If the update is not successful, then repeat the process. If users want to debug the system from COM4(EX9521/22/23)/COM1(EX9527), just power on the system with INIT* floating. **Note: INIT* must be disconnected from ground immediately before the updating disk image completes, otherwise the system will hang.**

Quick Start

■ Initial pin & GND:

Each EX952N module has a build-in EEPROM to store configuration information such as address, type, baudrate and other information. Sometimes, user may forget the configuration. Therefore, the EX952N have a special mode named "INIT mode" , to help user to resolve the problem. The "INIT mode" is setting as **Address=00, Baudrate=9600bps, Data bit=8, Parity=none, Stop bit=1, No checksum**

To enable INIT mode, please following these steps:

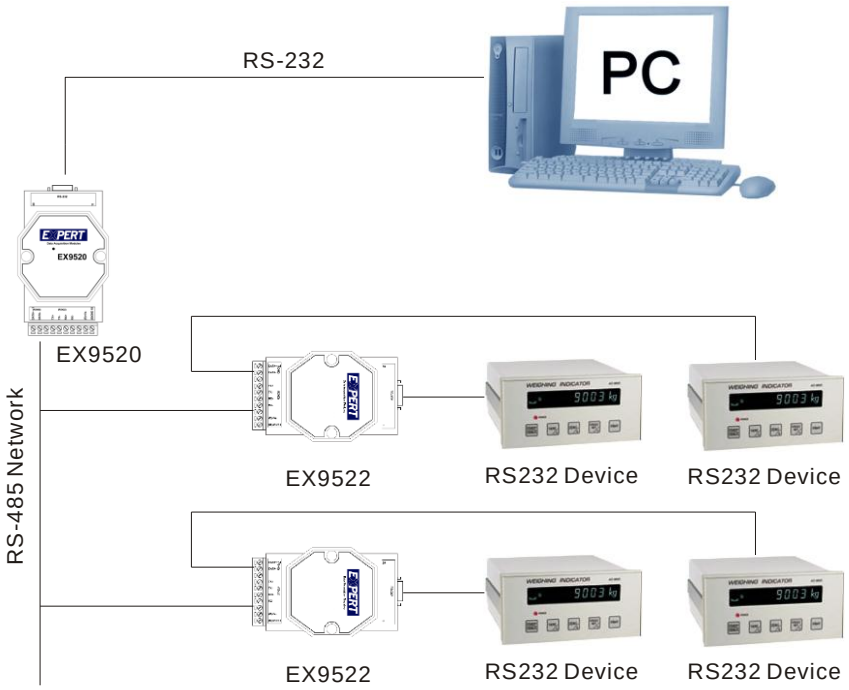
Step1. **Power on the 952N.**

Step2. **Touch the Init* to GND after power on then release the Init* with GND**

■ Data format setting (COM port & status):

■ Address setting:

■ Wire connection:



Command Sets

Command	Response	Description
$AAA(addr)	!AA	Read/Set the Module Address
$AABN(baud rate)	!AA(baud rate)	Read/Set Baud Rate of COM-1/2/3/4/5/6/7/8
$AADN(data-bit)	!AA(data-bit)	Read/Set the Data-Bit of COM-1/2/3/4/5/6/7/8
$AAPN(parity-bit)	!AA(parity-bit)	Read/Set the Parity-Bit of COM-1/2/3/4/5/6/7/8
$AAON(stop-bit)	!AA(stop-bit)	Read/Set the Stop-Bit of COM-1/2/3/4/5/6/7/8
$AA6(ID)	!AA	Set the ID-string of COM-1/3/4/5/6/7/8
$AA7	!AA(ID)	Read the ID-string of COM-1/3/4/5/6/7/8
$AAC(delimiter)	!AA(delimiter)	Read/Set the delimiter of COM-1/3/4/5/6/7/8
$AAD	!AA(delimiter)	Read the delimiter of COM-1/3/4/5/6/7/8
(delimiter) AA (bypass)	!AA	Bypass data-string to COM-1/3/4/5/6/7/8
$AAKN(CrLfMode)	!AA(CrLfMode)	Read/Set the checksum-status of COM2(RS485)
$AATN(CrLfMode)	!AA(CrLfMode)	Read/set the end-char of COM-1/2/3/4
$AAXV	!AA	Set the RTS state of COM1
$AA5	!AAS	Read the resent status
$AAF	!AA(number)	Read the firmware version number
$AAM	!AA(name)	Read the module name
$AA2	!AABDPK	Read the configuration of COM2(RS485)
$AAW	!AAS	Read the CTS status of COM1
$AAU	(data)	Read data from RS232 COM port buffer

Address mapping

	9521	9522	9523	9527
COM1(RS232)	AA	AA	AA	AA
COM2(RS485)	AA	AA	AA	AA
COM3(RS232)	X	AA+1	AA+1	AA+1
COM4(RS232)	X	X	AA+2	AA+2
COM5(RS232)	X	X	X	AA+3
COM6(RS232)	X	X	X	AA+4
COM7(RS232)	X	X	X	AA+5
COM8(RS232)	X	X	X	AA+6

1. User must send command to COM2(RS-485), if the command is used to set configuration of 952N, the 952N will directly echo message to COM2. If the command is used to bypass data to RS-232 device, the data will be sent to COM1/3/4/5/6/7/8(RS-232). Any message come from COM1/3/4/5/6/7/8(RS-232) will be echo to COM2(RS-485) at any time.

2. The COM port address of 9521 is given as following:
   COM1=RS-232 --> RS-232 Address=AA
   COM2=RS-485

3. The COM port address of 9522 is given as following:
   COM1=RS-232 --> RS-232 Address=AA
   COM2=RS-485
   COM3=RS-232 --> RS-232 Address=AA+1

9522 is used as two 9521 with address AA, AA+1.
The COM port must use the same ModeCrLf485 and the ModeCrLf232 can be different. The delimiters for two COM port can be different.

4. The COM port address of 9523 is given as following:
   COM1=RS-232 --> RS-232 Address=AA
   COM2=RS-485
   COM3=RS-232 --> RS-232 Address=AA+1
   COM4=RS-232 --> RS-232 Address=AA+2

9523 is used as three 9521 with address AA, AA+1,AA+2.
The three COM port must use the same ModeCrLf485 and the ModeCrLf232 can be different. The delimiters for these three COM port can be different.

5. The COM port address of 9527 is given as following:

COM1=RS-232 --> RS-232 Address=AA
COM2=RS-485
COM3=RS-232 --> RS-232 Address=AA+1
COM4=RS-232 --> RS-232 Address=AA+2
COM5=RS-232 --> RS-232 Address=AA+3
COM6=RS-232 --> RS-232 Address=AA+4
COM7=RS-232 --> RS-232 Address=AA+5
COM8=RS-232 --> RS-232 Address=AA+6

9527 is used as seven 9521 with address AA,AA+1,AA+2,AA+3,AA+4,AA+5,AA+6. The three COM port must use the same ModeCrLf485 and the ModeCrLf232 can be different. The delimiters for these three COM port can be different.

1. $AAA(addr)

Description: Read/Set the module address

Syntax: $AAA(addr)[chk](Cr)
$ is a delimiter character
AA module address 00 to FF

Response: valid command !AA
Invalid command ?AA

Example:
command: $01A02 receive: !02
address 01 is changed to 02
command: $02AA0 receive: !A0
address 02 is changed to 0xA0
command: $00A receive: !02
address stored in EEPROM=02

2. $AABN(baud rate)

Description: Read/Set the baud rate of COM-1/2/3/4/5/6/7/8

Syntax: $AABN(baud rate)[chk](Cr)
$ is a delimiter character
AA module address 00 to FF
N 0=Read/Set baud rate of RS485, 1=Read/Set baud rate of RS232
(baud rate) 300/600/1200/2400/4800/9600/19200/38400/57600/115200

Response: valid command !AA(baud rate)
Invalid command ?AA

Example: (Assume the AA of EX9523 is 01)
command: $01B0115200 receive: !01
Changes RS485(COM2) to 15200 BPS
command: $01B19600 receive: !01
Changes RS232(COM1) to 9600 BPS
command: $02B138400 receive: !02
Changes RS232(COM3) to 38400 BPS
command: $03B157600 receive: !03
Changes RS232(COM4) to 57600 BPS

SHORT-CODE FOR BAUD RATE:
300=1, 600=2, 1200=3, 2400=4, 4800=5, 9600=6, 19200=7, 38400=8, 57600=9,115200=A. The short-code of baud rate will be shown in the 7-SEGMENT **LED3**.

3. $AADN(data-bit)

Description: Read /Set the Data-Bit of COM-1/2/3/4/5/6/7/8

Syntax: $AADN(data-bit)[chk](Cr)
$ is a delimiter character
AA module address 00 to FF
N 0=Read/Set the data-bit of RS485, 1=Read/Set the data-bit of RS232
(data-bit): 7 or 8

Response: valid command !AA(data-bit)
Invalid command ?AA

Example: (Assume the AA of EX9523 is 01)
command: $01D08 receive: !01
Changes data-bit of RS485(COM2) to 8
command: $01D17 receive: !01
Changes data-bit of RS232(COM1) to 7
command: $02D17 receive: !02
Changes data-bit of RS232(COM3) to 7
command: $03D17 receive: !03
Changes data-bit of RS232(COM4) to 7

VALID DATA-BIT:

	COM1	COM2	COM3	COM4	COM5	COM6	COM7	COM8
9521	7/8	7/8	X	X	X	X	X	X
9522	7/8	7/8	7/8	X	X	X	X	X
9523	7/8	7/8	7/8	7/8	X	X	X	X
9524	7/8	7/8	7/8	7/8	X	X	X	X
9527	8	8	7/8	7/8	7/8	7/8	7/8	7/8

4. $AAPN(data-bit)

Description: Read/Set the parity-bit of COM-1/2/3/4/5/6/7/8

Syntax: $AAPN(parity-bit)[chk](Cr)
$ is a delimiter character
AA module address 00 to FF
N 0=Read/Set the parity-bit of RS485, 1=Read/Set the parity-bit of RS232
(parity-bit): 0=NONE, 1=EVEN, 2=ODD

Response: valid command !AA(data-bit)
Invalid command ?AA

Example: (Assume the AA of EX9523 is 01)
command: $01P00 receive: !01
Changes parity-bit of RS485(COM2) to NONE
command: $01P10 receive: !01
Changes parity-bit of RS485(COM1) to NONE
command: $02P11 receive: !02
Changes parity-bit of RS485(COM3) to EVNE
command: $03P12 receive: !03
Changes parity-bit of RS485(COM4) to ODD

VALID PARITY-BIT:

	9521	9522	9523	9527
COM1 (RS232)	N/E/O	N/E/O	N/E/O	N
COM2 (RS485)	N/E/O	N/E/O	N/E/O	N
COM3 (RS485)	X	N/E/O	N/E/O	N/E/O
COM4 (RS232)	X	X	N/E/O	N/E/O
COM5 (RS232)	X	X	X	N/E/O
COM6 (RS232)	X	X	X	N/E/O
COM7 (RS232)	X	X	X	N/E/O
COM8 (RS232)	X	X	X	N/E/O

N: None E: Even O: Odd

5. $AAON(stop-bit)

Description: Read/Set the stop-bit of COM-3/4/5/6/7/8

Syntax: $AAPN(stop-bit)[chk](Cr)
$ is a delimiter character
AA module address 00 to FF
N 0=Read/Set the parity-bit of RS485, 1=Read/Set the parity-bit of RS232
(stop-bit) 1 for COM1/2, 1/2 for COM3/4

Response: valid command !AA(data-bit)
invalid command ?AA

Example: (Assume the AA of 9523 is 01)
command: $02O12 receive: !02
Changes the stop-bit of RS232 (COM3) to 2
command: $03O12 receive: !03
Changes the stop-bit of RS232 (COM4) to 2

Valid stop-bit:

	9521	9522	9523	9527
COM1 (RS232)	1 OR 2	1 OR 2	1 OR 2	1
COM2 (RS485)	1 OR 2	1 OR 2	1 OR 2	1
COM3 (RS232)	X	1	1	1 OR 2
COM4 (RS232)	X	X	1	1 OR 2
COM5 (RS232)	X	X	X	1 OR 2
COM6 (RS232)	X	X	X	1 OR 2
COM7 (RS232)	X	X	X	1 OR 2
COM8 (RS232)	X	X	X	1 OR 2

6. $AA6(ID)

Description: Set the ID-string of COM-1/3/4/5/6/7/8, max=50 Characters

Syntax: $AA6(ID)[chk](Cr)
$ is a delimiter character
AA module address 00 to FF
(ID): ID-string, max. 50 characters

Response: valid command !AA
invalid command ?AA

Example: (Assume the AA of 9523 is 01)
command: $016Temperature! receive: !01
ID of RS232 (COM1) is Temperature1
command: $026HP34401A-1 receive: !02
ID of RS232 (COM3) is HP3440A-1
command: $036HP34401A-2 receive: !03
ID of RS232 (COM4) is HP3440A-2

7. $AA7

Description: Read the ID=-String of COM-1/3/4/5/6/7/8

Syntax: $AA7[chk](Cr)
$ is a delimiter character
AA module address 00 to FF

Response: valid command !AA(ID)
invalid command ?AA
(ID): ID-String, max. 50 characters.

Example: (Assume the AA of 9523 is 01)
command: $017 receive: !01 Temperature1
ID of RS232 (COM1) is Temperature1
command: $027 receive: !02 HP34401A-1
ID of RS232 (COM3) is HP34401A-1
command: $037 receive: !03 HP34401A-2
ID of RS232 (COM4) is HP3440A1-2

8. $AAC(delimiter)

Description: Read/Set the delimiter of COM-1/3/4/5/6/7/8

Syntax: $AAC(delimiter)[chk](Cr)
$ is a delimiter character
AA module address 00 to FF
(delimiter): default delimiter is :

Response: valid command !AA (delimiter)
invalid command ?AA

Example: (Assume the AA of 9523 is 01)
command: $01C receive: !01:
Reads the delimiter of RS232 (COM2) :
command: $02C receive: !02:
Reads the delimiter of RS232 (COM3) :
command: $03C* receive: !03:
Changes the delimiter of RS232 (COM4) :

Note:
Note1: The delimiter of COM1/3/4/5/6/7/8 can be different.
Note2: The default delimiter is :
Note3: The delimiter cannot be $, ~, #, @, %, CR & LF

9. $AAD

Description: Read the delimiter of COM-1/3/4/5/6/7/8

Syntax: $AAD[chk](Cr)
$ is a delimiter character
AA module address 00 to FF

Response: valid command !AA (delimiter)
invalid command ?AA
(delimiter): default delimiter is:

Example: (Assume the AA of 9523 is 01)
command: $01D receive: !01:
Reads the delimiter of RS232 (COM1) :
command: $02D receive: !02:
Reads the delimiter of RS232 (COM3) :
command: $03D receive: !03*
Changes the delimiter of RS232 (COM4) *

Note:
Note1: The delimiter of COM1/3/4/5/6/7/8 can be different.
Note2: The default delimiter is :

10. (delimiter)AA(bypass)

Description: Bypass data-string to COM-1/3/4/5/6/7/8

Syntax: (delimiter) AA (bypass)[chk](Cr)
AA module address 00 to FF
(bypass): data-string send to COM-1/3/4

Response: valid command !AA
invalid command ?AA

Example: (Assume the AA of 9523 is 01, the delimiters for COM1/3/4 are **: / ;/***)
command: :01abcde receive: !01
Sends **abcde** to COM1
command: :02123456789 receive: !02
Sends **123456789** to COM3
command: *03 test receive: !03
Sends **test** to COM4

11. $AAKN(CrLfMode)

Description: Read/Set the checksum-status

Syntax: $AAKN(CrLfMode)[chk](Cr)
$ is a delimiter character
AA module address 00 to FF
N 0= checksum disable, 1= checksum enable

Response: valid command !AA(CrLfMode)
invalid command ?AA
N=0 checksum disable, N=1 checksum enable

Example: (Assume the AA of 9523 is 01, the other AA of 9523 is 04)
command: $01K0 receive: !01
Disables checksum
command: $04K1 receive: !04
The checksum is enabled

Note: The checksum enable/disable is valid to COM2 only.

12. $AATN(CrLfMode)

Description: Read/Set the end-char of command string

Syntax: $AAK(CrLfMode)[chk](Cr)
$ is a delimiter character
AA module address 00 to FF
N 0= Read/Set the parity-bit of RS485, 1= Read/Set the parity-bit of RS232
(CrLfMode):
0=(CrLf)→0x0D
1= (CrLf)→0x0D+0x0A
2= (CrLf)→0x0A
3= (CrLf)→0x0A+0x0D
4= No end-char

Response: valid command !AA(data-bit)
invalid command ?AA

Example: (Assume the AA of 9523 is 01)
command: $01T0 receive: !014
The end-char of COM2 is no end character
command: $01T1 receive: !011
The end-char of COM1 is 0x0D+0x0A
command: $02T1 receive: !022
The end-char of COM3 is 0x0A
command: $03T1 receive: !033
The end-char of COM4 is 0x0A+0x0D

Note: The default CrLfMode = 4 then the default (CrLf) = NONE for all port.

13. $AAXV(CrMode)

Description: Set the RTS-state of COM1

Syntax: $AAXV[chk](Cr)
$ is a delimiter character
AA module address 00 to FF
V 0= set RTS inactive, 1= set RTS to active_HIGH state

Response: valid command !AA
invalid command ?AA

Example: (Assume the AA of 9523 is 01)
command: $01X0 receive: !01
Sets the RTS of COM1 to inactive state
command: $02X1 receive: !02
Sets the RTS of COM3 to active-HIGH state

	COM1(RS232)	COM2(RS485) / COM3 / COM4
9521	AA	AA / X / Download
9522	AA	(AA/AA+1) / (AA+1) /Download
9523	AA	(AA/AA+1/AA+2)/(AA+1)/(AA+2/Download)

	COM1(RS232)	COM2(RS485)/ COM3/COM4/COM5/COM6/COM7/COM8
9527	AA/Download	(AA/AA+1/AA+2/AA+3/AA+4/AA+5/AA+6)/ (AA+1)/(AA+2)/(AA+3)/(AA+4)/(AA+5)/(AA+6)

Note: The RTS-state is valid for COM1 & COM3

14. $AA5

Description: Read resets status.

Syntax: $AA5[chk](Cr)
$ is a delimiter character
AA module address 00 to FF

Response: valid command !AAS
invalid command ?AA
Syntax error or communication error or address error
S=0 it has not been reset since the last reset status read
S=1 it has been reset since the last reset status read

Example:
command: $015 receive: !011
It is first time power-on reset
command: $015 receive: !010
It is normal
command: $015 receive: !010
It is normal
command: $015 receive: !011
This module has been “reset” by module hardware watchdog. Therefore all output is going to its **start-values** now.

15. $AAF

Description: Reads the firmware version number.

Syntax: $AAF[chk](Cr)
$ is a delimiter character
AA module address 00 to FF

Response: valid command !AA(number)
invalid command ?AA
number = 4- character for version number

Example:
command: $01F receive: !01A2.0
Module 01 version 2.0
command: $02F receive: !02A3.0
Module 02 version 3.0

16. $AAM

Description: Reads the module name.

Syntax: $AAM[chk](Cr)
$ is a delimiter character
AA module address 00 to FF

Response: valid command !AA(name)
invalid command ?AA
number = 4- character or 5-character for module name

Example:
command: $01F receive: !019521
Name of module 01 is 9521
command: $02F receive: !029523
Name of module 02 is 9523

17. $AA2

Description: Reads the configuration code of COM2 (RS485) stored in EEPROM

Syntax: $AA2[chk](Cr)

$ is a delimiter character

AA module address 00 to FF

Response: valid command !AA40BDPK

invalid command ?AA

B short-code for baud rate

Baudrate	300	600	1200	2400	4800	9600	19200	38400	57600	115200
short-code	1	2	3	4	5	6	7	8	9	A

D data-bit: 7 or 8

P party-bit: 0=NONE, 1=EVEN, 2=ODD

K checksum status: 0= checksum disable, 1= checksum enable

Example: (Assume IN/T*=GND)

command: $002 receive: !01406800

Address 01 is 9521 series module, 9600 BPS, N81, checksum disable

command: $002 receive: !0240A801

Address 02 is 9521 series module, 115200 BPS, N81, checksum enable

18. $AAW

Description: Read the CTS-STATUS of COM1

Syntax: $AAW[chk](Cr)
$ is a delimiter character
AA module address 00 to FF

Response: valid command !AAS
invalid command ?AA
S: 0→ CTS is inactive now , 1→ CTS is active-High now
AA= 2-character HEX module address

Example: (Assume the AA of 9523 is 01)
command: $01W receive: !010
The CTS of COM1 is inactive now
command: $02W receive: !021
The CTS of COM3 is active –HIGH now

	COM1(RS232)	COM2(RS485) / COM3 / COM4
9521	AA	AA / X / Downloard
9522	AA	(AA/AA+1) / (AA+1) /Download
9523	AA	(AA/AA+1/AA+2)/(AA+1)/(AA+2/Downlord)

	COM1(RS232)	COM2(RS485)/ COM3/COM4/COM5/COM6/COM7/COM8
9527	AA/Download	(AA/AA+1/AA+2/AA+3/AA+4/AA+5/AA+6)/ (AA+1)/(AA+2)/(AA+3)/(AA+4)/(AA+5)/(AA+6)

Note : The CTS-status is valid for COM1 & COM3

19. $AAU

Description: Read data from the COM port buffer

Any 232 device should obey the rules of request-reply constitution. In other word, 232 devices are passive. If they have not received any commands, they will not send any message out. However, since the active device frequently appear, our controller is designed with a buffer to receive these message in this situation.
Buffer operation rules:
Rule 1: buffer is enabled after power-on.
Rule 2: (delimiter) AA command (ref. Sec.3.10) disables buffer operation of that port
Rule 3: after disabling buffer, the first incoming message will transfer to COM2. Then controller waits for 10 seconds. If no message has arrived, the buffer is enabled again.

Syntax: $AAU[chk](CrLf) -> read first data in buffer
$ is a delimiter character
AA 2-character HEX module address, from 00 to FF
[chk] 2-character checksum, if checksum disable -> no [chk]
(CrLf) End-Char

Response: valid command (data)[chk](CrLf)
invalid command ?AA[chk](CrLf)
No response: buffer is empty or
syntax error or
communication error or
address error

! is a delimiter character indicating a valid command
? is a delimiter character indicating an invalid command
AA=2-character HEX module address
[chk]=2-character checksum, if checksum disabled -> no [chk]
(CrLf)=End-Char

Example:

Command: $01U(CrLf) Response: data1(CrLf)

Retrieves “data1” from buffer

Command: $01U(CrLf) Response: data2

Retrieves another data “data2” from buffer

Command: $02U(CrLf) Response:

No data is in buffer

Warning:

(1)Change CrLf mode will corrupt the integrity of unread data in the buffer

(2)Repeat this command several times to ensure the buffer is empty.