Interfacing With API's Line Of Instrumentation

1.0 General

The CPP-3794 supports serial interfaces with the following API instruments;

Model 100, 100A - SO_2 analyzer,
Model 200, 200A - NOx analyzer,
Model 300, 300A - CO analyzer,
Model 400, 400A - O_3 analyzer,
Model 360 - CO_2 analyzer,
Model 700 - Calibrator

API has delivered instruments using two operating software programs. Older instruments have the older operating system, and newer instruments (A model numbers) have the newer or AMX operating system. The CPP can collect data from both types, but it must be set up properly. The CPP uses the Baud rate to determine the operating system. If the Baud rate in the instrument can be set to 9600 Baud, then it is the AMX operating system, and must be set to 9600 Baud. The older operating system is limited to 2400 Baud, and should be set to 2400 Baud.

The CPP polls the instrumentation at a minimum of once per minute. The API instruments require several interrogations. The first poll retrieves the present data reading. The second interrogation is to retrieve the instruments operating mode. The remaining polls deal with retrieving and clearing instrument alarming status. Retrieved status information is decoded by the CPP and appended to the data value status word. Internal operational alarms are contained in the received status and are detected by the CPP, which flags the data point with an I modifier. Reference the CPP manual for a more detailed discussion of status indicators.

The API instruments support a password feature that is designated as a LOGON feature. This is not supported by the CPP and should be disabled in the set up of the instrument. The optimum communicating set up is to place the instrument into the quiet/computer mode, with the LOGON feature disabled. This is accomplished in the VARS menu. To get to the variable menu, press SETUP-MODE-VARS-ENTR and scroll to RS-232_MODE by pressing NEXT, then press EDIT. The possible values are:

Decimal Value	Description
1	Turns on quiet mode(messages suppressed)
2	Places analyzer in computer mode(no echo of chars)
4	Enables security features(LOGON)
8	Enables RS-232 menus display
16	Enables alternate protocol set up
32	Enables multi-drop support for RTS

To enter the correct value, add the decimal values of the features that you want to enable. The proper selection is eleven (11), 1+2+8=11. This turns on quiet mode, selects computer mode, and disables the LOGON feature. The factory default is 8, which works with the CPP as well.

The operator can also communicate directly with the instrumentation. This feature is presented in the Operator/Instrument Interface section of this appendix.

2.0 Connections

The newer API instruments have a rear panel switch that allows the unit to operate electrically as either a DCE device or a DTE device as defined in the IEEE RS-232 standards. The older instruments are DTE only. The factory DCE setting allows the unit to be connected directly to the comm port of a PC.

Although API instruments can be assigned an ID and addressed, they do not tri-state their outputs, therefore some external multiplexing device is required to multidrop API instruments. The IMD offered by H2NS will provide this multiplexing function. In a multi-drop configuration, all instrumentation is "daisy chained" together with 9 pin RS-232 cables. Then one 9 pin RS-232 cable is connected to a comm port on the CPP.

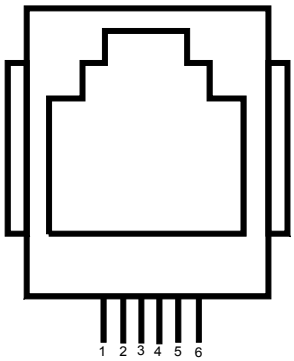
A better approach when connecting API instrumentation is to run a separate cable from separate comm ports on the CPP to each API instrument. Additionally, any combination of the above two methods can be utilized.

In normal operation the CPP controls the clocks in the instrumentation. The CPP contains a very long term, accurate clock. Following a power up condition, or a time change, and then periodically, the CPP downloads the time and date to all connected instrumentation.

The newer API instruments default to 9600 Baud operation, the older units default to 2400 Baud. This should not be changed. The API instrumentation RS-232 word length should always be set for 8 bits, no parity, and one stop bit.

RJ-11 Comm Port Connections

H2NS installs a 6 pin, RJ11, telephone jack connector into the CPP rear panel. The majority of applications only use four of the 6 pins, which are pins 2,3,4,5. A null modem can be readily implemented by either twisting or not twisting the telephone cable, which is readily detectable because the cable is color-coded. **Standard, purchased telephone cables are twisted.** Presented below are the pin assignments set up by H2NS for the RJ11 connector.



1 RTS Output
2 TX+ Output of CPP
3 Gnd (TX- in RS-422/485)
4 Gnd (RX- in RS-422/485)
5 RX+ Input to CPP
6 CD Input

DB9 to RJ-11 Connectors

H2NS can also provide RS-232 connectors that are terminated in an RJ11 connector. One of the RS-232 connectors can be connected to an instrument and then connected with an RJ11 cable to the CPP. The wire colors for the 9 pin, DB male and female connectors are given below. These colors apply only to the connector delivered by H2NS. Other manufacturers of these type connectors may use a different color scheme.

Pin	9 Pin Male	9 Pin Female
1	Blu	Wht
2	Yel	Blk
3	Grn	Red
4	Red	Grn
5	Blk	Yel
6	Wht	Blu

API DB9 to RJ-11 Connector

The output of newer API instruments can be switch selected to be either a DCE or DTE configuration in regards to pins 2 & 3, Rx & Tx. The rear panel connector is a 9 pin female DB connector. Select the DCE configuration. The proper wiring for the mating 9 pin male DB to RJ-11 is given below. The pin assignments assume that a standard, twisted telephone cable is used. These connectors can be ordered from H2NS. When connecting to an older API instrument, the telephone cable should be fabricated as a non-twist cable.

RJ-11 to 9 Pin DBM to API

API-9 Pin	Color	RJ-11 Pin	RJ-11 Signal/Pin @ CPP
2 Tx	Blk	5	2 – Rx
3 Rx	Yel	2	5 – Tx
5 Gnd	Grn	3	3 – Gnd

3.0 Initialization

The CPP is initialized via the operator port, comm port #4. Entering an **I<cr>** results in the CPP bringing up a Main Menu selection. Selecting the channel initialization results

in the CPP asking a series of questions as presented below. Operator responses are in **bold**, and **<cr>** means a carriage return. The numbers 1.), 2.) and so forth are used for discussion purposes only and are not printed in an actual system initialization.

1.) Channel # = **1<cr>**
2.) Name = NAME **NOX<cr>**
3.) Units = UNITS **PPB<cr>**
4.) Full Scale = 100.0 **500.0<cr>**
5.) Zero = 0.0 **0.0<cr>**

6.) Instrument Manufacturer
00 = Other
01 = API
02 = CLI
03 = ML
04 = Thermo
05 = Vaisala
7.) Setting = 00 **1<cr>**

8.) API Model #
01 = SO2 - 100A
02 = NOX - 200A
03 = NO2 - 200A
04 = NO - 200A
05 = CO - 300A
06 = O3 - 400A
07 = 700 - Cal

9.) Setting = 01 **02<cr>**
10.) Comm Port (3-00 Thru 3-08) = 3-01 3- **02<cr>**
11.) Baud Rate
1 = 300, 2 = 1200, 3 = 2400, 4 = 4800, 5 = 9600, 6 = 19.2K, 7 = 38.4K = 5**<cr>**

12.) Instrument ID = 0000 **<cr>**
13.) IMD Installed = 00 **<cr>**
14.) # Points For Interim Avg = 01 **04<cr>**
15.) # Points For Final Avg = 01 **9<cr>**
16.) Unip(U)/Bip(B) = U **<cr>**
17.) Unip(U)/Bip(B) Cals = B **<cr>**
18.) Channel # = **<cr>**

Line one simply asks which channel is to be initialized. A carriage return here exits back to the Main Menu. In other steps, a carriage return input leaves the setting unchanged and the CPP goes to the next question. In cases where the operator must provide an input, the CPP asks the same question again.

Lines 2 through 5 request information that is not required for proper operation when interfaced to an instrument over the serial port. It is, however, good operating procedures to set these inputs to match those in the instrument. The correct full scale

and zero values are required if data stored in the CPP is being output to some other device (for example, Odessa Engineering's Envicom program) in a binary format. This is also true of channels set up as ADC input channels.

Channel names and the units can be up to six characters in length. The full scale and zero values can be four digits in length and the number of digits after the decimal point (if a decimal point is used) must be the same in the full scale and zero inputs. The inputs can also have a minus sign.

Line 6 lists all of the manufacturer drivers that are resident in this software version. In the example, five manufacturers are presented. As the manufacturers are alphabetized, the numbers delineating the various manufacturers will change depending on how many drivers are installed.

In Line 7 of our example, the operator selected API as the manufacturer. The CPP then prints a list of the model numbers that are supported for the API instrument line.

In Lines 8 & 9, the operator input a **02** which selects the NOx analyzer.

In Line 10 the CPP asks to which comm port this analyzer is going to be connected. This is the connector on the CPP into which the cable from the instrument is plugged. The allowable inputs are 00-08, which is nine comm ports. As discussed above, with the use of an IMD, all API instrumentation can be plugged into the same comm port. In our example, the operator selected comm port 3-02.

The comm port nomenclature should not be made overly complicated or confusing. Although comm Ports 3-00 through 3-08 are independent serial ports, internally the comm port 3 designation references interfacing to the instrumentation. Comm port 3-00 addresses comm port #3 physically located on the CPP printed circuit board and labeled comm #3 on the CPP back panel. Comm ports 3-01 through 3-08 address comm ports physically located on the expansion printed circuit board and are labeled comm 3-01 through comm 3-08 on the CPP back panel.

It should be noted that comm port #2 can be used to interface to some instrumentation. The API line of instrumentation is **not** included in that group, and should always be assigned to a comm port number 3 designation. Comm port #2 is usually used with instruments that broadcast, which is normally meteorological instrumentation.

In line 11, the CPP is asking what baud rate should be used to communicate with this instrument. This setting must be 9600 for the AMX operating system, and 2400 Baud for the older instruments.

Line 12 asks if an instrument ID is required. An ID is not required with the API instruments. It is recommended that an ID not be used. If an ID is set up, Input the appropriate digits ID. In the example, the user did not input an ID.

Line 13 is asking if an IMD (Intelligent Multidrop Device) is installed in this comm port cabling. The IMD, manufactured by H2NS, allows a number of different instruments to be connected to a single comm cable. The instruments can have different protocols and even different baud rates. This is very useful when a collection of instrumentation is located remotely from the CPP. The IMD allows only one cable to be run between the CPP and the instruments, with the IMD providing the instrument multiplexing locally at the instruments. The IMD supports addresses from 01 through 60. An address of 00 indicates that an IMD is not installed in this comm cabling.

Lines 14 & 15 are questions regarding data validity qualifiers. The first (line 14) asks how many valid one minute samples the CPP must have to qualify the interim average as valid. Question 14 asks the same in regards to the final average.

Lines 16 & 17 This allows negative data values to either be accepted or set to zero. Usually, negative values are set to zero, except during calibrations.

4.0 Calibrations

To set up this channel as a calibration channel reference the CPP manual. Calibrations can be set up to be controlled by the API instrument operating from it's internal clock, or by the clock in the CPP, or by an external controller. The CPP can be configured (by channel) to detect external status bits as calibration indicators, or the CPP can be configured to generate either contact closures to control calibrations or to send RS-232 commands to the instruments to sequence calibrations.

5.0 Internal/External Status

Please reference appropriate sections of the CPP manual for a more detailed discussion of internal and external control and status features. These quite often tie into the overall system calibration schemes. On an individual basis, with each data poll the CPP also sends commands that return the instruments status. Flags indicating the operational mode, either sampling, zero, span, or diagnostics are retrieved and decoded by the CPP. Additional polls check for the instruments operating condition and any potentially abnormal condition is flagged with an I modifier.

6.0 Communicating directly with API Instrumentation

The CPP supports two modes in which an operator can communicate directly with the connected instruments. One is an on line mode, in which the CPP interleaves operator and polling commands, and the second is an off line, unabated pass through mode.

On Line Communications

Via comm port #4, the CPP has a menu that defines the appropriate commands associated with the instrument selected. Via comm port #1, enter SETQ to switch comm port #1 into comm port #4. Input **D43<cr>** and the CPP responds with a list of instrumentation connected to the unit, as shown below;

```
Chn#     Manu        Mdl#          Type
01 =   THERMO      SO2-43C       - SO2
02 =   API         NOX-200A      - NOX
03 =   API         NO2-200A      - NOX
04 =   API         NO- 200A      - NOX
04 =    CLI        WS            - WS
05 =    CLI        WD            - WD

Selection = 2<cr>
02 =  API  NOX-200A  -  NOX  Selected

CMD> ?<cr>
T =  Test Measurement
D =  Diagnostics
W = Warnings
V  = Variables
```

C = Calibrations

CMD> = **NT? <cr>**
#NT01 = NOx Reading (%T NOX)
#NT02 = NO2 Reading (%t NO2)
#NT03 = NO Reading (%T NO)
#NT04 = Sample Input Range (%T RANGE)
MORE>

Note - this is an example, the actual list is much longer. The MORE> indicates that more commands are available. At this point commands can be sent to the instruments as well as from the CMD> prompt. A carriage return input presents more commands. The N is required as it ensures that the system is communicating with an NOx instrument. This would be as S when communicating with an SO2 instrument, an O when communicating with an Ozone analyzer, an a C when communicating with a CO Analyzer.

The CPP will provide a menu of the available commands for the selected instrument. Referencing the above example, the operator can set retrieve a data reading in two ways. The first is to send the command as specified by the instrument manufacturer, *e.g.,* [**%T NOX<cr>**].

The brackets [ ] are for clarity only, and **should not be** entered.. The percent sign (**%**) is required as it apprises the CPP that the string input by the operator is to be sent to the instrument. This string command must be used if parameters are to be sent to the instrument.

Alternately, the operator can input [**#NF01<cr>**] and the CPP will assemble and send the command to the instrument. In both cases instrument responses are presented. A "No Response" message is printed if the instrument does not respond. These abbreviated commands can only be use to interrogate the instruments, not to send parameters to the instruments.

If the operator had input [**#NF01?<cr>**] the response would have been;

#NF01 = NOx Reading (%T NOX)

CMD>

or a definition of just this command as opposed to definitions of all available commands.

The first character following the #, in this example N, identifies that the operator is communicating with an NOX instrument. Had the SO2 instrument been selected, the commands would have a format of **#SF01**. The ozone instrument would be **#OF01**, and the CO instrument would be **#CF01**.

In the pass through mode, strings can then be sent to the instrument with the **%[string]<cr>** inputs. The CPP remains in the pass through mode with this channel until it is exited as discussed below, or the pass through times out from inactivity. To select another channel, the pass through mode can be exited and reentered with another **D43xx** input, or at the CMD> prompt, enter an **M<cr>**. The CPP will present a listing of all instruments connected and ask for a selection. Select the channel desired.

To terminate the pass through mode input [**^<cr>**] at the CMD> prompt. If a character is not input for two minutes, the CPP times out and the pass through mode is exited automatically.

The CPP remains on line collecting data even in the pass through mode. Operator commands are interlaced with the CPP normal polling or interrogation commands.

Off Line Pass Through

The CPP offers an off line, unabated pass through mode to connected instruments. Over comm port #1, a command is entered commanding the CPP to connect one of the comm port to comm port #1 and back off. Information can be passed via the instrument and the user without adhering to any CPP protocol. Manufacturer diagnostics, such as APICOM, can be conducted remotely. Only the channel being passed through is off line. Data is being collected from all other channels.

More discussion of this is provided in Tech Note 36.

Off Line Pass Through

The rear panel of the CPP has three RJ-11 Pass through connectors. Invoking this feature removes the CPP from the loop, and allows a user to communicate directly, unabated, with the instrument. This requires that the RS-232 cable from the instrument be connected to the comm port over which the CPP is collecting data, and also connected to one of the pass through connectors. More discussion of this is provided in Tech Note 36.

7.0 Error Log

The CPP maintains a running error log that lets the user determine if communications are occurring properly. After connecting instrumentation clear the error log by inputting an **EC<cr>**. After some period of time (two to three minutes) check the error log (**E<cr>**) for errors. In proper operation, there should be no errors. The Error Log lists the following.

No Response	- No response from the instrument to an interrogation
Response Error	- CPP received response but detected an error in the data field
Unknown Resp	- CPP received a response but could not decode the response
Model # Error	- CPP received a response but not from the type of instrument it is configured to interrogate on this channel

The three most common causes of errors are, the cable does or does not need a null modem, or the cable is connected to the wrong comm port on the CPP, or the baud rate set in the CPP does not match that set in the instrument. With API instrumentation, another source of potential error is that the instrument is set up in with the LOGON feature enabled.