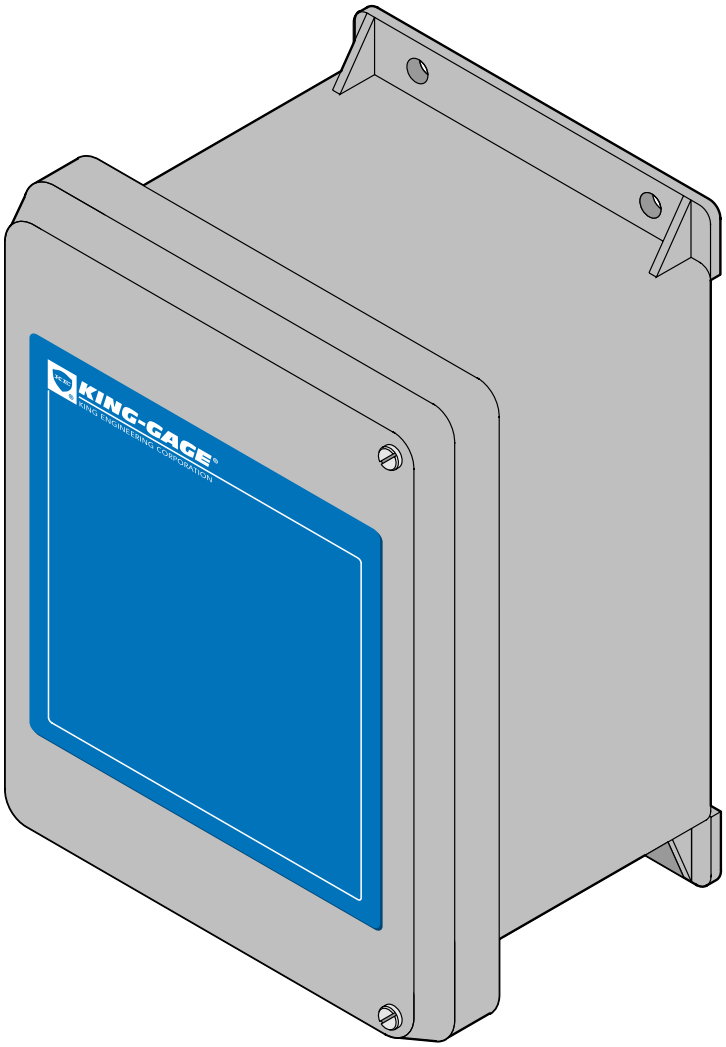
Effective: May, 2000 EX-1823

LevelPRO™

Optional Network Communications



DeviceNet Gateway

Part No. : 5390-1

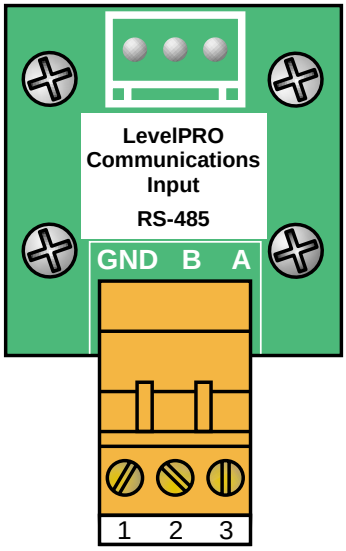
KING-GAGE®
Since 1937 KING ENGINEERING CORPORATION

Revisions:

May, 2000 – Original Release.

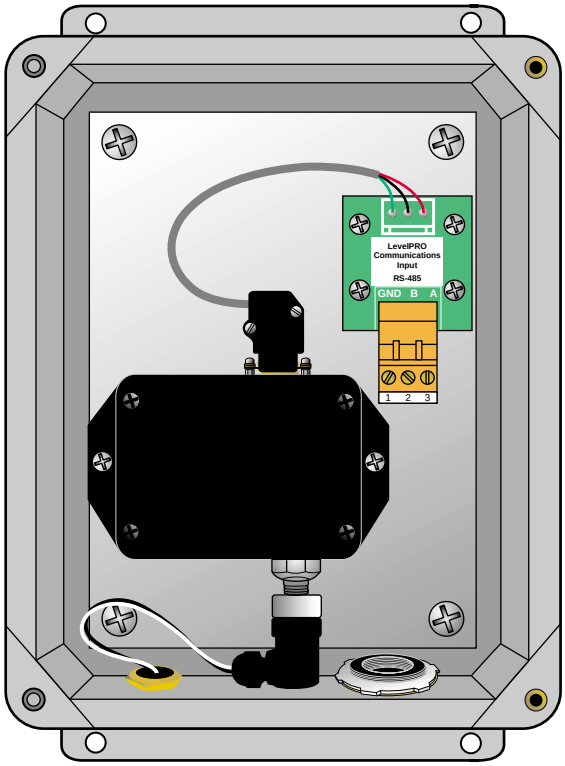
LevelPRO Tank Level Processors can be used on a DeviceNet communications network by employing the **CDN067 RS422/485 DeviceNet adapter** ("gateway") **manufactured by DIP Products**. This provides for the physical connection between the RS-485 serial communications network linking the LevelPRO processor(s) and the DeviceNet network.

Actual data communications depends upon programming the "host" to query the LevelPRO processor(s) and interpret the subsequent data response. Please refer to the ***King Bus ASCII Communications*** detailed in the LevelPRO processor manual for the specific query-response format.

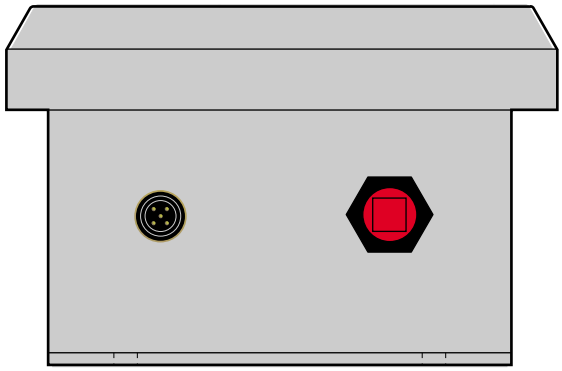


DEVICENET GATEWAY
Part No. 5390-1

Internal Connections (RS-485 Interface)



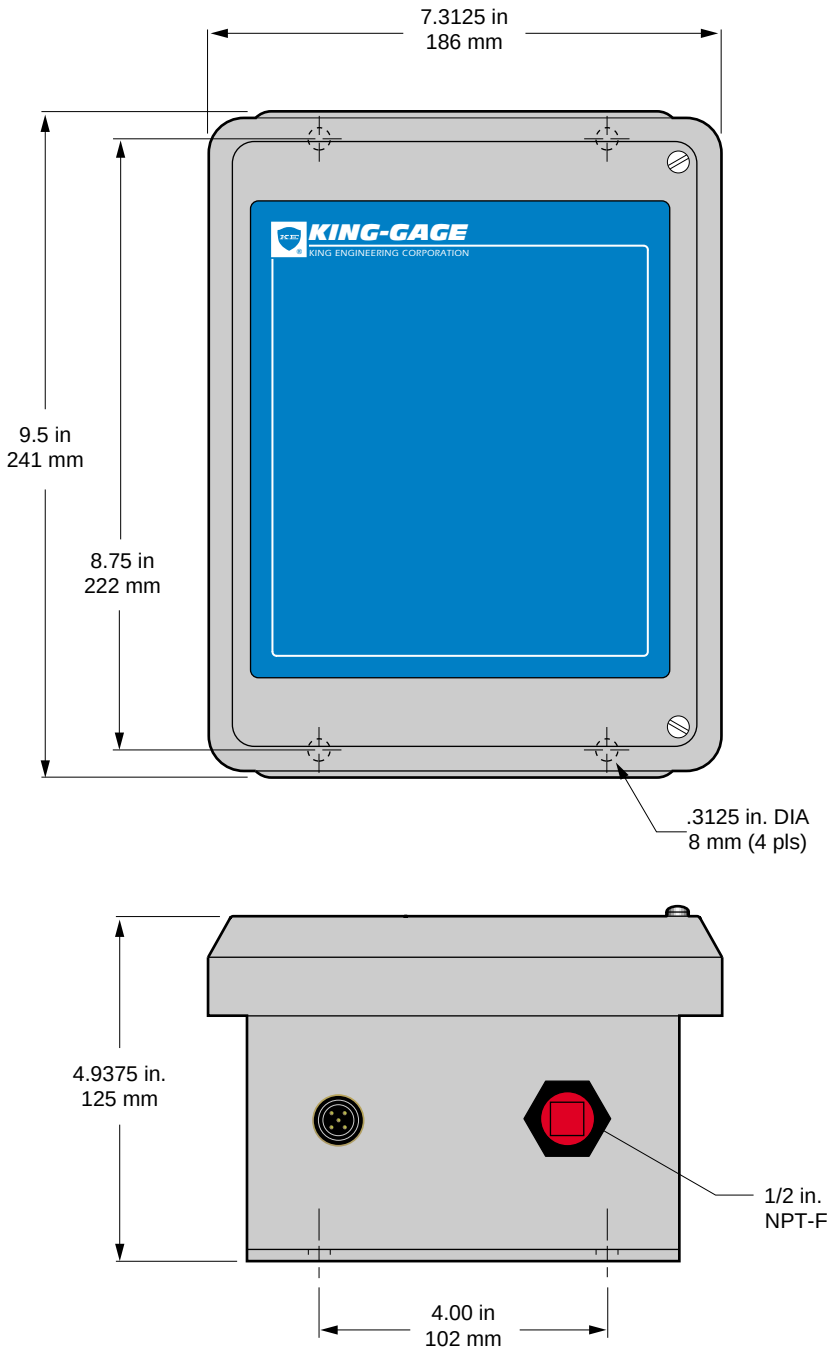
Front View - Model 5390-1



Bottom View - Model 5390-1

DEVICENET GATEWAY
Part No. 5390-1

Mounting Dimensions (NEMA 4 enclosure)



CONFIGURING THE DEVICENET GATEWAY

The electrical hook-up is uncomplicated. The CDN067 has a unique DeviceNet address (MAC ID) and is connected to the DeviceNet network. It receives necessary power directly from the DeviceNet wires. A standard watertight DeviceNet connector is provided. A male 9-pin D-shell connector provides the external connection to the LevelPRO RS-485 network. One or more physical LevelPRO devices may be connected to the network.

In order to achieve useful communications with the LevelPRO device(s), there is a small amount of programming that has to be done on the DeviceNet side. The exact method for this programming will depend on what "smart" device is operating your specific DeviceNet application (PC, PLC, etc.). The basic operation is that a valid LevelPRO request is assembled and written to a register (DeviceNet attribute) in the CDN067. After a short delay, the LevelPRO response can be read from a different register. It is then up to the programmer to parse this response as necessary to extract the desired information. The request and response are in the format detailed in the LevelPRO manual.

Initially, the CDN067 RS-485 side must be configured. The configuration is stored in non-volatile memory and should be pre-set from King Engineering Corporation. The configuration parameters are set in the User Defined Object, Class Code 64 (0x40), Instance 1, Attributes 3-24. See the table below for configuration values and meanings. Also, the DeviceNet MAC ID must be assigned depending on your specific circumstances.

Attribute	Name	Value/Notes
3	Receive Data (from RS-485 LevelPRO Network)	(1)
4	Transmit Data (to RS-485 LevelPRO Network)	(2)
5	Status	(3)
6	Baud Rate	0/0x06 (4)
7	Parity	0x00
8	Data Size	0x08
9	Stop Bits	0x01
10	Flow Control	0x00
11	Receive Count	N/A
12	Transmit Count	N/A
13	Maximum Receive Size	0x20
14	Data Format	0x02
15	Block Mode	0x00
16	Receive Delimiter	N/A
17	Pad Char	N/A
18	Maximum Transmit Size	0x20
19	Idle String	N/A
20	Fault String	N/A
21	Status Enable	0x00
22	Status Clear Enable	0x00
23	Four Wire	0x00
24	Option Switch	N/A (4)

NOTES:

1. The RS-485 LevelPRO response data can be read from this attribute. A valid response will be 31 bytes long. Depending on the specifics of your control device, DeviceNet overhead items such as message fragmentation, headers and so forth may or may not be transparent.

2. A properly formatted RS-485 LevelPRO request must be written to this attribute. Depending on the specifics of your control device, DeviceNet overhead items such as message fragmentation, headers and so forth may or may not be transparent.

3. Status gives information about RS-485 errors that may have occurred. Writing a zero clears the errors. DeviceNet should periodically clear this attribute.

4. Baud rate should be set for 0x06 (19200, default) or 0x00 (9600) depending on your RS-485 LevelPRO network. In order to write to this attribute, the Option Switch must first be set to a value of nine (9). Attempting to write to the attribute with the Option Switch set to any other value returns a DeviceNet error.

Specific Example

An example showing the actual DeviceNet packets is given below. This should help someone who is familiar with DeviceNet in coding the proper request. This example assumes a Master DeviceNet device with a MAC ID of 1 and a CDN067 with a MAC ID of 63. A LevelPRO device with tanks present on ASCII addresses 001, 002 and 003 is connected to the RS-485 side.

The proper RS-485 ASCII request for LevelPRO information for Tank 1 is

#001*

or viewed as a hex string

23 30 30 31 2A

This must be encoded into a DeviceNet message as a fragmented message, since it will require more than eight bytes. Depending on your master device, this may or may not be transparent. If your device can handle the fragmentation by itself, simply write the string to Class 0x40, Instance 1, Attribute 0x04. If not, it is coded as:

[5FC]: <81 00 10 40 01 04 05 23 30 30 31 2A>

where

[5FC]	is the DeviceNet header containing the source device
81	shows that the message is fragmented
00	is the fragmentation protocol
10	is the command
40	is the class
01	is the instance
04	is the attribute
05	is the number of bytes in the data
23 30 30 31 2A	is the message to be sent out the RS-485 port

What actually transpires on the DeviceNet bus is:

[5FC]: <81 00 10 40 01 04 05 23>	first message fragment
[5FB]: <81 C0 00>	first fragment acknowledge
[5FC]: <81 81 30 30 31 2A>	last fragment
[5FB]: <81 C1 00>	Last fragment acknowledge
[5FB]: <01 90>	

The actual response produced by this request differs depending on LevelPRO configuration and current settings. For this example we'll assume that the LevelPRO returned the ASCII string

001 1.104 R00000150 GALS 04E4<CR><LF>

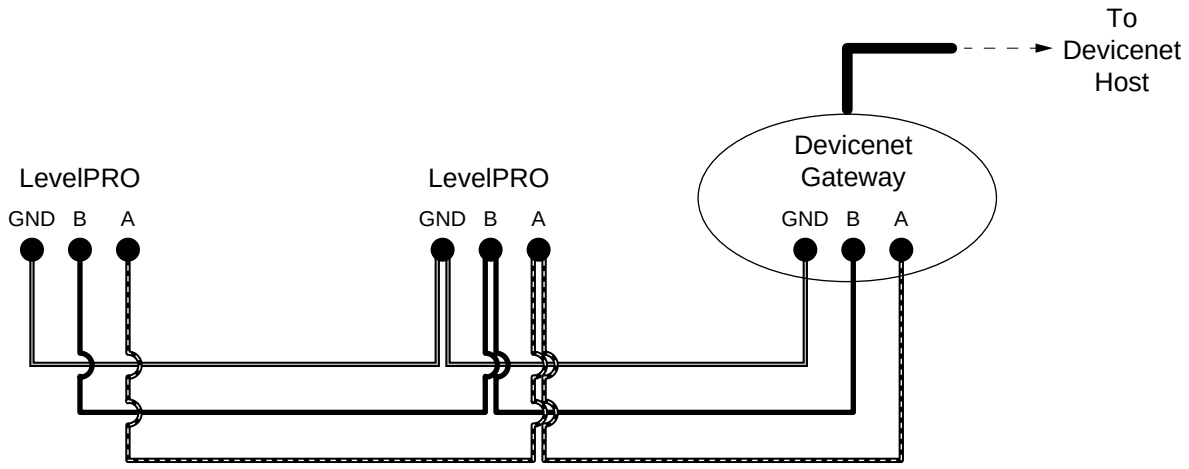
which, as viewed as a hex string is

30 30 31 20 30 2E 31 30 34 20 52 30 30 30 30 30 31 35 30 20 47 41 4C 53 20 30 34 45 34 0D 0A

This string is what would be returned from attribute 0x03 after the request. Depending on your specific DeviceNet Master device, there may be some fragmentation and DeviceNet protocol overhead bytes included. What actually transpires on the DeviceNet bus when a Get Attribute 0x03 is executed is:

[5FC]: <41 0E 40 01 03>	Get Attribute 0x03
[5FB]: <C1 00 8E 1F 30 30 31 20>	First Fragment containing number of bytes (0x1F or 31)
[5FC]: <81 C0 00>	First Fragment Acknowledge
[5FB]: <81 41 31 2E 31 30 34 20>	Second Message Fragment
[5FC]: <81 C1 00>	Second Fragment Acknowledge
[5FB]: <81 42 52 30 30 30 30 30>	Third Message Fragment
[5FC]: <81 C2 00>	Third Fragment Acknowledge
[5FB]: <81 43 31 35 30 20 47 41>	Fourth Message Fragment
[5FC]: <81 C3 00>	Fourth Fragment Acknowledge
[5FB]: <81 44 4C 53 20 30 34 45>	Fifth Message Fragment
[5FC]: <81 C4 00>	Fifth Fragment Acknowledge
[5FB]: <81 85 34 0D 0A>	Last Message Fragment
[5FC]: <81 C5 00>	Last Fragment Acknowledge

The programer should allow a short delay (~500 mSec) after sending the request before reading the attribute 0x03.



Box 1228, Ann Arbor, Michigan 48106-1228 U.S.A.
Phone: (734) 662-5691 ▪ FAX: (734) 662-6652