# DeviceNet Slave Device

# CAN-2018D

# User's Manual

## Warranty

Without contrived damage, all products manufactured by ICP DAS are warranted in one year from the date of delivery to customers.

## Warning

ICP DAS revises the manual at any time without notice. However, no responsibility is taken by ICP DAS unless infringement act imperils to patents of the third parties.

## Copyright



## Trademark

The brand name ICP DAS as a trademark is registered, and can be used by other authorized companies.

## Contents

# 1 Introduction

## 1.1 Overview

DeviceNet is one kind of the network protocols based on the CAN bus and mainly used for the embedded network such as industrial machine control, factory automation, medical equipments control, remote data acquisition, environmental monitoring, and packaging machines control, etc.

The CAN-2018D follows DeviceNet specification Volume I/II, Release 2.0. You can obtain the analog input data and configure the module via standard DeviceNet protocol. To build a connection between DeviceNet master and the CAN-2018D is no more a tough project. The DeviceNet EDS file would help you to achieve this purpose quickly and easily. This module has 8 analog input channels for measuring the thermocouples or voltage values. It can be fit to various applications. By using the DeviceNet masters of ICP DAS, you can simply build a DeviceNet network for your requirements.

## 1.2 Hardware Specifications

**Analog Input:**

- Input Channels: 8
- Input Type: +/- 15mV, +/- 50mV, +/- 100mV, +/- 500mV, +/- 1V, +/- 2.5V, -20mA ~ +20mA (Requires Optional External 125Ω Resistor) Thermocouple(J, K, T, E. R. S, B, N, C)
- Resolution: 16-bit
- Accuracy: +/-0.1% FSR
- Sampling Rate: 10 Samples/ sec (Total)
- Zero Drift: +/- 10μV/ °C
- Span Drift: +/- 25 ppm/ °C
- Common Mode Rejection: 86 dB
- Normal Mode Rejection: 100 dB
- Input Impedance: >400 kΩ
- Individual Channel Configuration: Yes
- Open Thermocouple Detection: Yes
- Over voltage Protection: 240 Vrms
- 4KV ESD Protection: Yes, Contact for each terminal.
- Intra-module Isolation, Field to Logic: 3000 $V_{DC}$

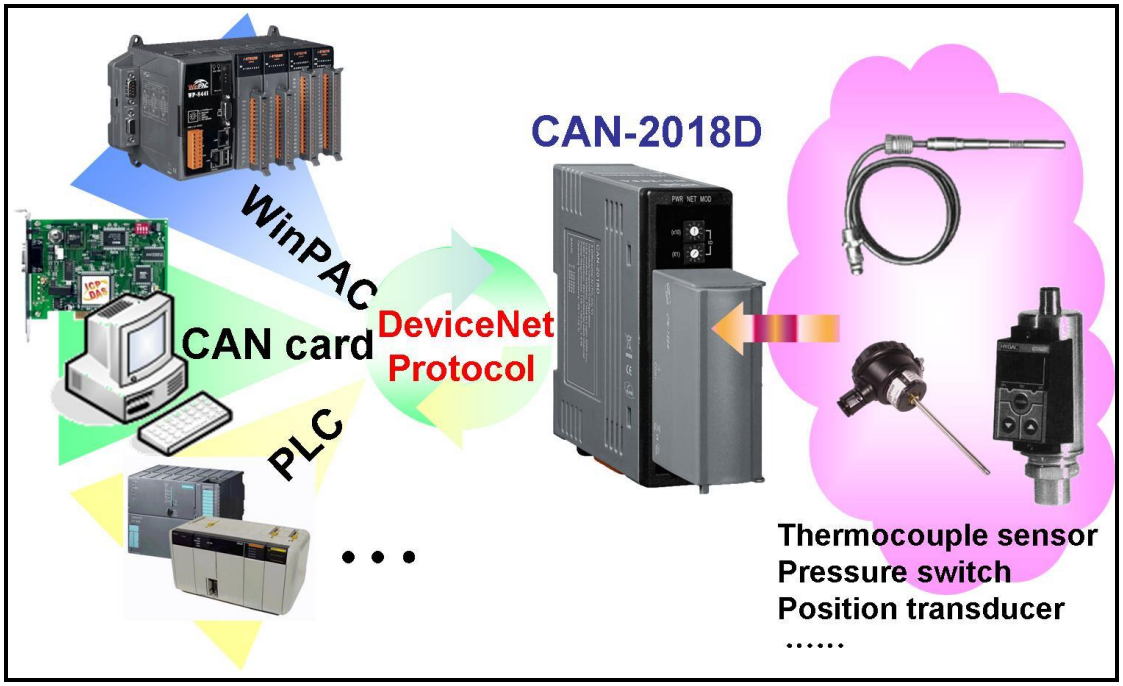
**Others:**

- Power LED: PWR (red)
- DeviceNet Status Led: NET/MOD, green/orange
- Power Supply: Unregulated +10 ~ +30 $V_{DC}$.
- Power Consumption: 1.5 W.
- Operating Temperature: -25 ~ 75 ℃.
- Storage Temperature: -30 ~ 80 ℃.
- Humidity: 10 to 90% RH, Non-condensing.
- Dimensions: 32.3 mm x 99 mm x 78 mm (W x L x H) Detail.

## 1.3 Features

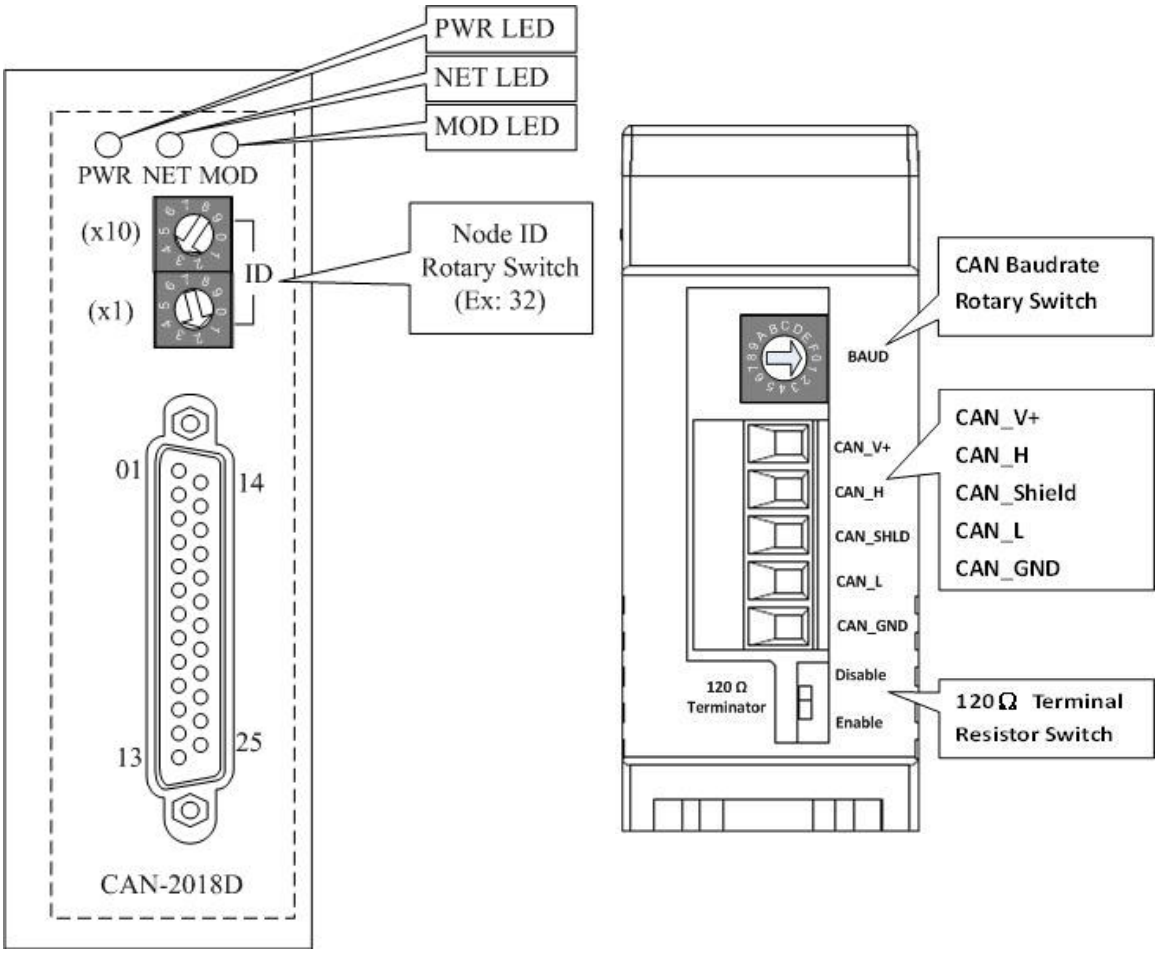
- DeviceNet general I/O slave device
- Comply with DeviceNet specification Volume I, Release 2.0 & Volume II, Release 2.0, Errata 5
- Group 2 Only Server (non UCMM-capable)
- Support Predefined Master/Slave Connection Set
- Connection supported:
  - 1 connection for Explicit Messaging
  - 1 connection for Polled I/O
  - 1 connection for Bit-Strobe I/O connection
- Support DeviceNet heartbeat and shutdown messages
- Provide EDS file for standard DeviceNet master interface.
- NET, MOD and PWR LED indictors

## 1.4 Application



# 2 Hardware

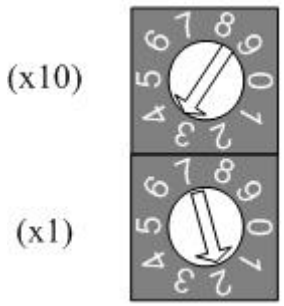
## 2.1 Structure




**(Top View)** **(Bottom View)**

## 2.2 Node ID & Baud Rate Rotary Switch

The rotary switches of node ID configure the node ID of CAN-2018D module. These two switches are numbers in base ten. For example, the node ID value of the following picture is 32.




**Node ID rotary switch**

The rotary switch about the baud rate configuration handles the CAN baud rate of CAN-2018D module. The relationship between the rotary switch value and the practical baud rate is presented in the following table.

**Baud rate rotary switch**

| Rotary Switch Value | Baud rate (kbps) |
|---|---|
| 0 | 125 |
| 1 | 250 |
| 2 | 500 |

## 2.3 LED Description

**PWR LED**

The CAN-2018D is powered by 10 ~ 30 $V_{DC}$. Under a normal connection, a good power supply and a correct voltage range selection, as the unit is turned on, the LED will light up in red.

**NET LED**

The NET LED indicates the current status of the DeviceNet communication link.

| Condition | status | Indicates |
|---|---|---|
| Init Off | Off line | Device is not online |
| Off | Connection timeout | I/O connection timeout |
| Flashing | On line | Device is on line, but not communicating |
| Init solid | Link failed | (Critical) Device has detected an error that has rendered it incapable of communicating on the link; for example, detected a duplicate node address or network configuration error |
| Solid | On line, communicating | Device is online and communicating |

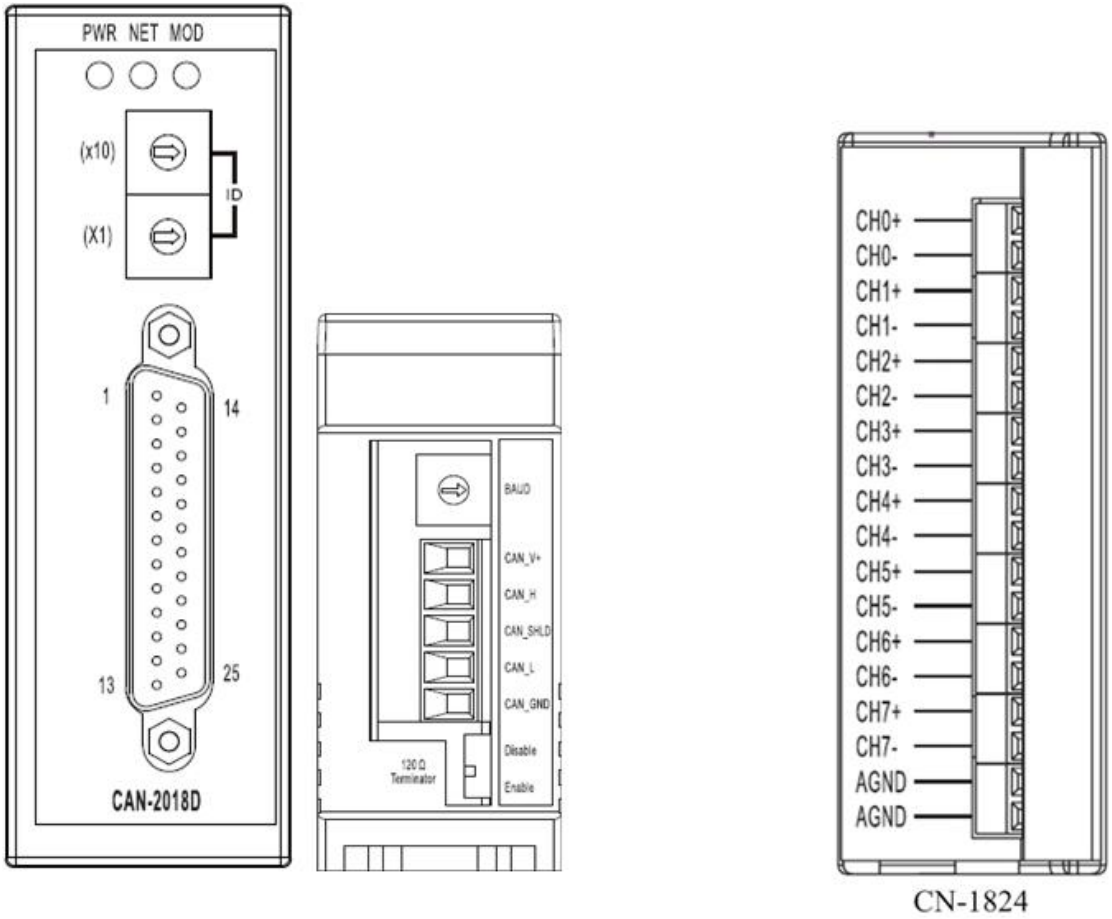
**MOD LED**

This LED provides the devices status. It indicates whether or not the device is operating properly.

| Condition | status | Indicates |
|---|---|---|
| Off | Normal | |
| Solid | Critical fault | Device has unrecoverable fault. |
| Flashing | Non_critical fault | Device has recoverable fault to recover. If users want to fix the problem, reconfiguring device's MAC ID or resetting device may work. |

## 2.4 PIN Assignment

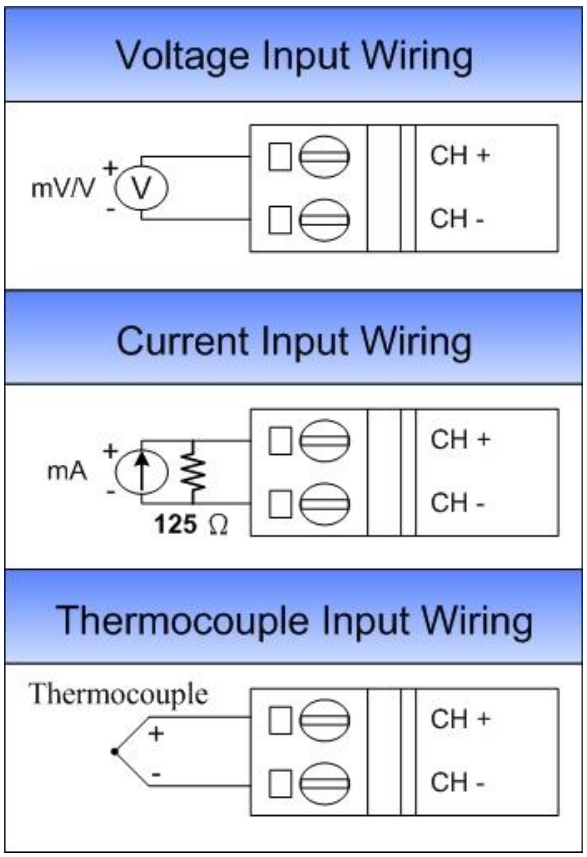
CAN-2018D + CN-1824 (transformation connector)



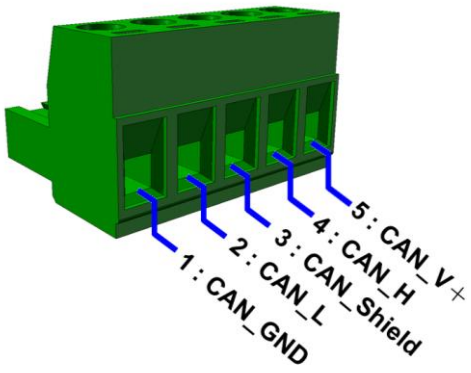

| Pin Assignment Name | Terminal No. | | Pin Assignment Name |
|---|---|---|---|
| +5V | 01 | 14 | DGND |
| CJC | 02 | 15 | CH0+ |
| CH0- | 03 | 16 | CH1+ |
| CH1- | 04 | 17 | CH2+ |
| CH2- | 05 | 18 | CH3+ |
| CH3- | 06 | 19 | CH4+ |
| CH4- | 07 | 20 | CH5+ |
| CH5- | 08 | 21 | CH6+ |
| CH6- | 09 | 22 | CH7+ |
| CH7- | 10 | 23 | N.C. |
| N.C. | 11 | 24 | N.C. |
| N.C. | 12 | 25 | AGND |
| AGND | 13 | | |
| | Shield | | F.G. |

CAN-2018D 25-pin Female D-Sub Connector

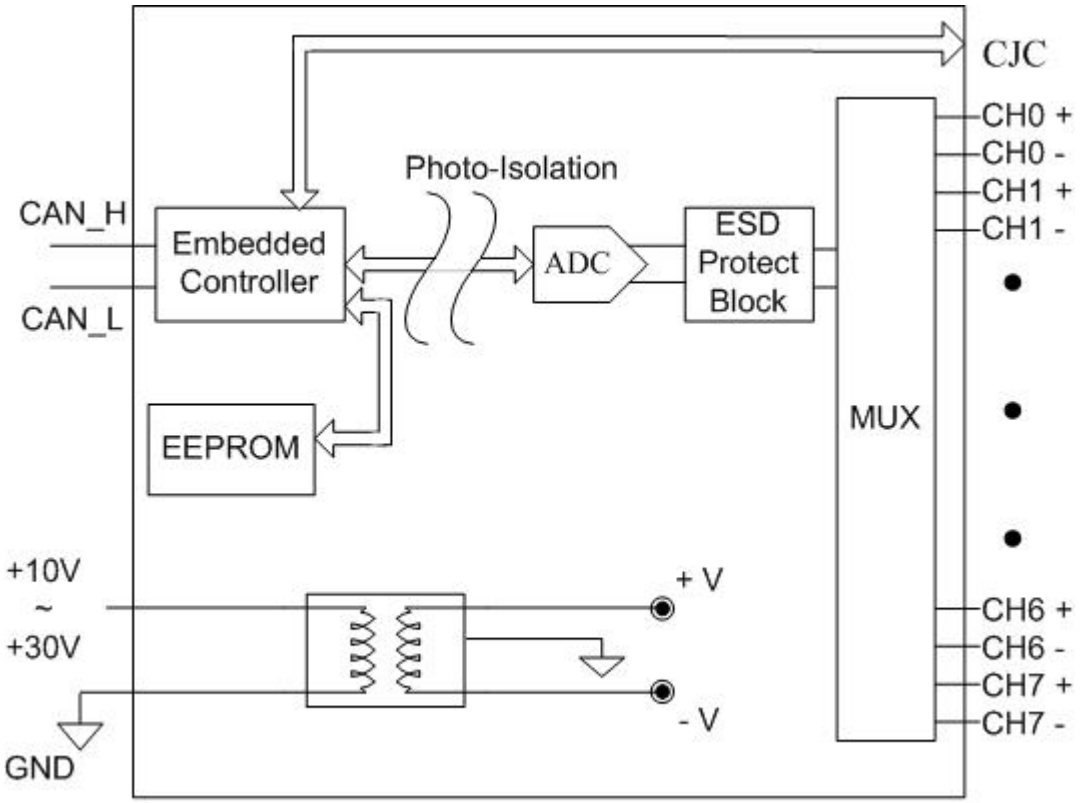
## 2.5 Wire Connection




**5-pin screw terminal block**



## 2.6 Internal I/O Structure




Internal I/O Structure

# 3 DeviceNet Profile Area

This section describes the detailed functions for each object class that is implemented in the CAN-2018D DeviceNet network.

## 3.1 DeviceNet Statement of Compliance

**General Device Data**

| Device Information | Description |
|---|---|
| Version Description of DeviceNet Specification | Volume I, Release 2.0 & Volume II, Release 2.0 |
| Vendor Name | ICP DAS |
| Device Profile Name | CAN-2018D |
| Production Revision | 1.1 |

**DeviceNet Physical Conformance Data**

| Item | Description |
|---|---|
| MAC ID Setting | Switch (0 ~ 63) |
| Communication Baud Rate Setting | Switch (125, 250, 500 kbps) |
| Default Baud Rate | 125 kbps |
| Predefined Master/Slave Connection Set | Group 2 Only Server |

## 3.2 Identity Object (Class ID: 0x01)

This object provides the identification and general information about the device.

**Class Attribute (Instance ID=0)**

| Attribute ID | Attribute name | Data Type | Method | Value |
|---|---|---|---|---|
| 0x01 | Revision | UINT | Get | 0001 |
| 0x02 | Max Instance | UINT | Get | 1 |

**Class Service**

| Service Code | Service name | Support |
|---|---|---|
| 0x0E | Get_Attribute_Single | Yes |

**Instance Attribute (Instance ID=1)**

| Attribute ID | Description | Method | DeviceNet Data Type | Value |
|---|---|---|---|---|
| 1 | Vendor | Get | UINT | 803 |
| 2 | Product type | Get | UINT | 0x00 |
| 3 | Product code | Get | UINT | 0x300 |
| 4 | Major. Minor of firmware version | Get | Struct of USINT USINT | 1.1 |
| 5 | Status | Get | WORD | - |
| 6 | Serial number | Get | UDINT | 1 |
| 7 | Product name | Get | Short_String | CAN-2018D |
| 10 | Heartbeat Interval | Get/Set | USINT | 0(default) |

**Instance Service**

| Service Code | Service name | Support |
|---|---|---|
| 0x0E | Get_Attribute_Single | Yes |
| 0x10 | Set_Attribute_Single | Yes |
| 0x05 | Reset | Yes |

**Note:** Use the Instance Service 0x05 will reboot the device.

## 3.3 Connection Object (Class ID:0x05)

This section presents the externally visible characteristics of the Connection Objects associated with the Predefined Master/Slave Connection Set within slave devices.

The default IO connection path is as follow.

| Connection Path | Class ID | Instance ID | Attribute ID |
|---|---|---|---|
| Poll Produced | 0x04 | 0x64 | 0x03 |
| Poll Consumed | 0x65 | 0x01 | 0x01 |
| Bit Strobe Produced | 0x04 | 0x64 | 0x03 |
| Bit Strobe Consumed | 0x65 | 0x01 | 0x01 |

| Connection Instance ID | Description |
|---|---|
| 1 | References the Explicit Messaging Connection into the Server |
| 2 | References the Poll I/O Connection |
| 3 | References the Bit–Strobe I/O Connection |

## 3.4 Assembly Object (Class ID: 0x04)

The Assembly Object binds attributes of multiple objects, which allows sending / receiving the data from / to each object over a single connection. Assembly objects can be used to bind input data or output data. The terms of ”input” and ”output” are defined from the network’s point of view. An input will produce data on the network and an output will consume data from the network. (The read temperature function is supported in firmware version 1.2 or later)

**Class attribute (Instance ID=0)**

| Attribute ID | Attribute name | Data Type | Method | Value |
|---|---|---|---|---|
| 0x01 | Revision | UINT | Get | 1 |
| 0x02 | Max Instance | UINT | Get | 0x04 |

**Class service**

| Service Code | Service name | Support |
|---|---|---|
| 0x0E | Get_Attribute_Single | Yes |

**Instance ID**

| Instance ID | OUTPUT | INPUT |
|---|---|---|
| 0x64 | | Get values of all AI channels |
| 0x65 | Type codes of all AI channel | Get type codes of all AI channels (***Note**) |
| 0x66 | Offset values of all AI channels | Get offset values of all AI channels |
| 0x67 | | Get single floating-point format value of all AI channels |

***Note:** About the definition of the type codes, please refer to the appendix B for details.

**Contents of Each Assembly Object Instance**

| Instance ID | Description | Type | Method | Default Value |
|---|---|---|---|---|
| 0x64 | Channel 0 ~ 7 AI values **(*Note1)** | UINT<br>…<br>UINT | Get | 0x0000<br>…<br>0x0000 |

| | | | | |
|---|---|---|---|---|
| 0x65 | Channel 0 ~ 7 AI type codes | USINT<br>…<br>USINT | Get/Set | 0x0E<br>…<br>0x0E |
| 0x66 | Channel 0 ~ 7 AI offset values | UINT<br>…<br>UINT | Get/Set | 0x0000<br>…<br>0x0000 |
| 0x67 | Channel 0 ~ 7 AI values **(*Note2)** | REAL<br>…<br>REAL | Get | 0x00000000<br>…<br>0x00000000 |

***Note1:** The range of values for variables of type UINT are defined in appendix B.

***Note2 :** The range of values for variables of type REAL are defined in IEEE 754 for the basic single floating-point format.

**Parameter description of Assembly Object Instance**

| Instance ID | Data Range | Parameter Description |
|---|---|---|
| 0x64 | Refer to Appendix B for the corresponding input values | Channel 0~7 input values **(*Note1)** |
| 0x65 | Refer to Appendix B for the type code definition | Channel 0~7 AI type codes |
| 0x66 | 0x0000 ~ 0xFFFF | Channel 0~7 AI offset value<br>0x0000 ~ 0x7FFF => 0 ~ +32767<br>0xFFFF ~ 0x8000 => -1 ~ -32768 |
| 0x67 | Refer to Appendix B for the corresponding input values | Channel 0~7 input values **(*Note2)** |

***Note1:** The range of values for variables are defined in appendix B.

***Note2 :** The range of values for variables are defined in IEEE 754 for the basic single floating-point format

**Instance attribute (Instance ID=0x64~0x68)**

| Attribute ID | Description | Method | DeviceNet Data Type | Value |
|---|---|---|---|---|
| 0x03 | Data | Get/Set | OUTPUT/ INPUT | Dependent on instance ID |

**Instance service**

| Service Code | Service name | Support |
|---|---|---|
| 0x0E | Get_Attribute_Single | Yes |

| | | |
|---|---|---|
| 0x10 | Set_Attribute_Single | Yes |

## 3.5 Application Object1 (Class ID: 0x64)

Application objects are the interfaces between an application and the DeviceNet Layer. The attributes of application Objects contain the data for the application, which are accessed and exchanged via the DeviceNet protocol. A DeviceNet device accesses application data by invoking read and write functions. These functions need to be provided by an Application Object. The DeviceNet protocol provides Get_Attribute_Single and Set_Attribute_Single to read and write the single AO channel configuration of the CAN-2018D module. (The read temperature function is supported in firmware version 1.2 or later)

**Class attribute (Instance ID=0)**

| Attribute ID | Attribute name | Data Type | Method | Value |
|---|---|---|---|---|
| 0x01 | Revision | UINT | Get | 1 |
| 0x02 | Max Instance | UINT | Get | 0x08 |

**Class service**

| Service Code | Service name | Support |
|---|---|---|
| 0x0E | Get_Attribute_Single | Yes |

**Instance ID**

| Instance ID | Description |
|---|---|
| 0x01 | For accessing channel 0 AI configuration |
| 0x02 | For accessing channel 1 AI configuration |
| 0x03 | For accessing channel 2 AI configuration |
| 0x04 | For accessing channel 3 AI configuration |
| 0x05 | For accessing channel 4 AI configuration |
| 0x06 | For accessing channel 5 AI configuration |
| 0x07 | For accessing channel 6 AI configuration |
| 0x08 | For accessing channel 7 AI configuration |

**Instance attribute (Instance ID=0x01~0x08)**

| Attribute ID | Description | Method | Data Type | Default Value |
|---|---|---|---|---|
| 0x01 | AI value | Get | UINT | 0x0000 |
| 0x02 | AI type code | Get/Set | USINT | 0x0E |
| 0x03 | AI offset value | Get/Set | UINT | 0x0000 |
| 0x04 | AI value | Get | REAL | 0x00000000 |

**Parameter description of Application Object1 attributes**

| Attribute ID | Data Range | Parameter Description |
|---|---|---|
| 0x01 | Refer to Appendix B for the corresponding input values | Channel**x** AI value (***Note1**) |
| 0x02 | Refer to Appendix B for the corresponding output values | Channel**x** AI type code |
| 0x03 | 0x0000 ~ 0xFFFF | Channel**x** AI offset value<br>0x0000 ~ 0x7FFF => 0 ~ +32767<br>0xFFFF ~ 0x8000 => -1 ~ -32768 |
| 0x04 | Refer to Appendix B for the corresponding input values | Channel**x** AI value (***Note2**) |

**Note:** Channel**x** is channel number of module, which is selected by Instance ID

***Note1:** The range of values for variables are defined in appendix B.

***Note2 :** The range of values for variables are defined in IEEE 754 for the basic single floating-point format

**Instance service**

| Service Code | Service name | Support |
|---|---|---|
| 0x0E | Get_Attribute_Single | Yes |
| 0x10 | Set_Attribute_Single | Yes |

## 3.6 Application Object2 (Class ID: 0x65)

Application Object2 not only defines parameters for saving configurations into EEPROM or loading factory default setting but also defines CJC parameters.

**Class attribute (Instance ID=0)**

| Attribute ID | Attribute name | Data Type | Method | Value |
|---|---|---|---|---|
| 0x01 | Revision | UINT | Get | 1 |
| 0x02 | Max Instance | UINT | Get | 0x03 |

**Class service**

| Service Code | Service name | Support |
|---|---|---|
| 0x0E | Get_Attribute_Single | Yes |

**Instance attribute (Instance ID=1)**

| Attribute ID | Description | Method | Data Type | Default Value |
|---|---|---|---|---|
| 0x01 | Save all configurations into EEPROM or using factory default configuration setting | Set | USINT | - |
| 0x02 | Set CJC (cold junction compensation) Offset value or Get CJC (offset) temperature value | Set/Get | UINT | 0x0000 |
| 0x03 | Enable/Disable CJC | Set/Get | USINT | 0x01 |

**Parameter description of Application Object2 attributes**

| Attribute ID | Data Range | Parameter Description |
|---|---|---|
| 0x01 | 0x01: Use default configuration<br>0x02: Save all configurations to EEPROM | 0x01: After restarting the device, all configurations will become default setting.<br>0x02: Save all channels configuration into EEPROM |
| 0x02 | Refer to Appendix B for the CJC (cold junction compensation) temperature values | CJC Enable => Get CJC temperature value<br>CJC Disable => Get CJC offset value |
| 0x03 | 0x00 or 0x01 | 0x00: Disable CJC<br>0x01: Enable CJC |

**Instance service**

| Service Code | Service name | Support |
|---|---|---|
| 0x0E | Get_Attribute_Single | Yes |
| 0x10 | Set_Attribute_Single | Yes |

# 4 Application

Application Object (Class ID:0x64, 0x65) lists all the parameters of the module. Each Instance ID is corresponding to the different channel. By using “Set/Get Attribute Single” service, you can read/write the parameters of each channel.

**Example1:**

Enable/Disable CJC.

(Class ID: **0x65**, Instance ID: **0x01**, Attribute ID **0x03**).

If the node ID of the CAN-2018D is 1, and the master (ID: 0x0A) has completed “Explicit” connection with the device. By setting the value of Attribute ID 0x03 to 0x01, you can set the CJC status to enable mode.

| IDENTIFIER BITS | | | | | | | | | | | RTR | Data Length | 8-byte Data (byte) (HEX) | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Destination MAC ID | | | | | | | | | | | | | | | | | | |
| 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 6 | 0A | 10 | 65 | 01 | 03 | 01 | -- | -- |

**Master** → **Slave (CAN-2018D)**

| IDENTIFIER BITS | | | | | | | | | | | RTR | Data Length | 8-byte Data (byte) (HEX) | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Source MAC ID | | | | | | | | | | | | | | | | | | |
| 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 2 | 0A | 90 | -- | -- | -- | -- | -- | -- |

**Master** ← **Slave (CAN-2018D)**

Set the value 0x01 to the Application Object2 with Instance ID 0x01 and Attribute ID 0x03. After sending the “Set Attribute Single”, the slave device will response 0x90 to mean that the setting is OK. Then the CJC status will become to “Enable” mode. Setting the value 0x00 to this attribute will disable the CJC status.

Via changing the Attribute ID of the Application Object, you can set other parameters of this device.

**Example2:**

Get channel0 AI data

(Class ID: **0x64**, Instance ID: **0x01**, Attribute ID **0x01**).

If the node ID of the CAN-2018D is 1, and the master (ID: 0x0A) has completed “Explicit” connection with the device. By getting the value of the object with attribute ID 0x01, you can get the channel 0 of the AI data.

| IDENTIFIER BITS | | | | | | | | | | | RTR | Data Length | 8-byte Data (byte) (HEX) | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Destination MAC ID | | | | | | | | | | | | | | | | | | |
| 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 5 | 0A | 0E | 64 | 01 | 01 | -- | -- | -- |

**Master** → **Slave (CAN-2018D)**

| IDENTIFIER BITS | | | | | | | | | | | RTR | Data Length | 8-byte Data (byte) (HEX) | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Source MAC ID | | | | | | | | | | | | | | | | | | |
| 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 4 | 0A | 8E | FF | 7F | -- | -- | -- | -- |

**Slave (CAN-2018D)** → **Master**

Get the value of Application Object1 with Instance ID 0x01 and Attribute ID 0x01. After sending the “Get Attribute Single”, the slave device responds the AI data. The value 0x7FFF means that the channel0’s input data is 0x7FFF. About the meaning of the AI value, refer to the appendix B for the AI data and type code definition.

Attribute 0x02 of Application Object1 is used for AI type code. Please refer to the appendix B for the AI data and type code definition. The Attribute 0x03 of Application Object1 can use to shift the AI data. The value 0x0000 ~ 0x7FFF means the AI value plus 0 ~ 32767. The value 0xFFFF ~ 0x8000 means the AI data plus -1 ~ -32768.

Attribute 0x02 of Application Object2 is used for CJC offset value. When enable the CJC, users can get the CJC value from this parameter. When disable the CJC, users can set/get the CJC offset value. Please refer to the appendix B for the range definition. The Attribute 0x03 of Application Object2 is used to enable/disable the CJC.

After modifying the value of the parameters, set attribute 0x01 of Application Object2 to “2” to save all channels’ configuration into the module’s EEPROM. Users also can set this attribute to “1” to be factory setting after rebooting.

**Example3:**

Get channel0 temperature value with K type thermocouple.

(Class ID: **0x64**, Instance ID: **0x01**, Attribute ID **0x04**).

If the node ID of the CAN-2018D is 1, and the master (ID: 0x0A) has completed “Explicit” connection with the device. By getting the value of the object with attribute ID 0x04, you can get the channel 0 of the temperature value.

| IDENTIFIER BITS | | | | | | | | | | | RTR | Data Length | 8-byte Data (byte) (HEX) | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Destination MAC ID | | | | | | | | | | | | | | | | | | |
| 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 5 | 0A | 0E | 64 | 01 | 04 | -- | -- | -- |

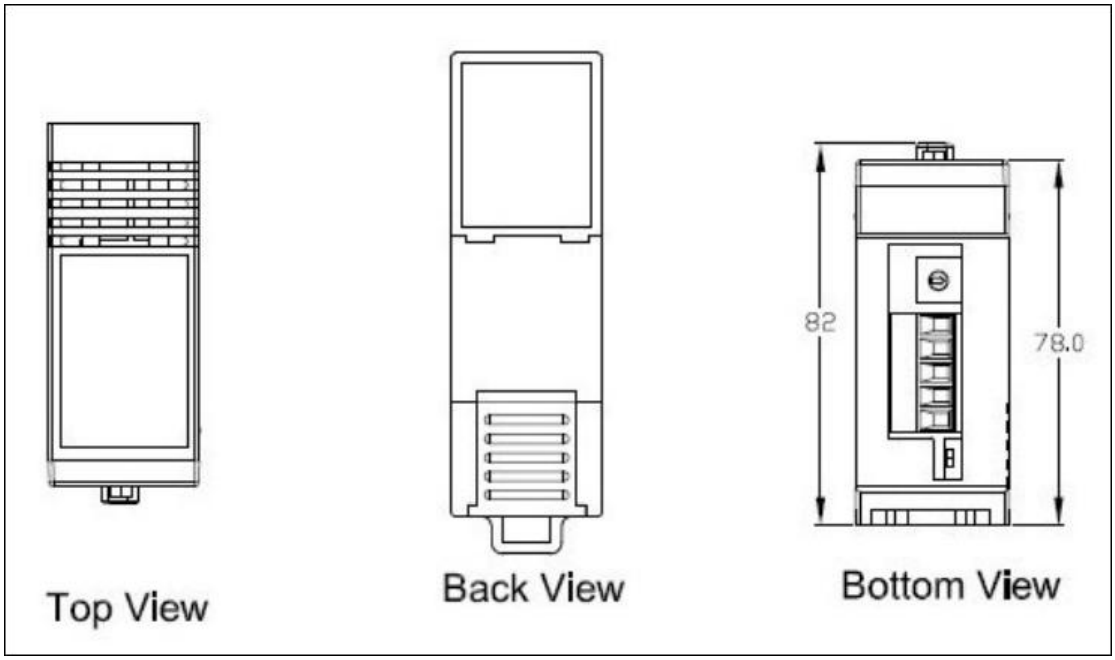
**Master** → **Slave (CAN-2018D)**

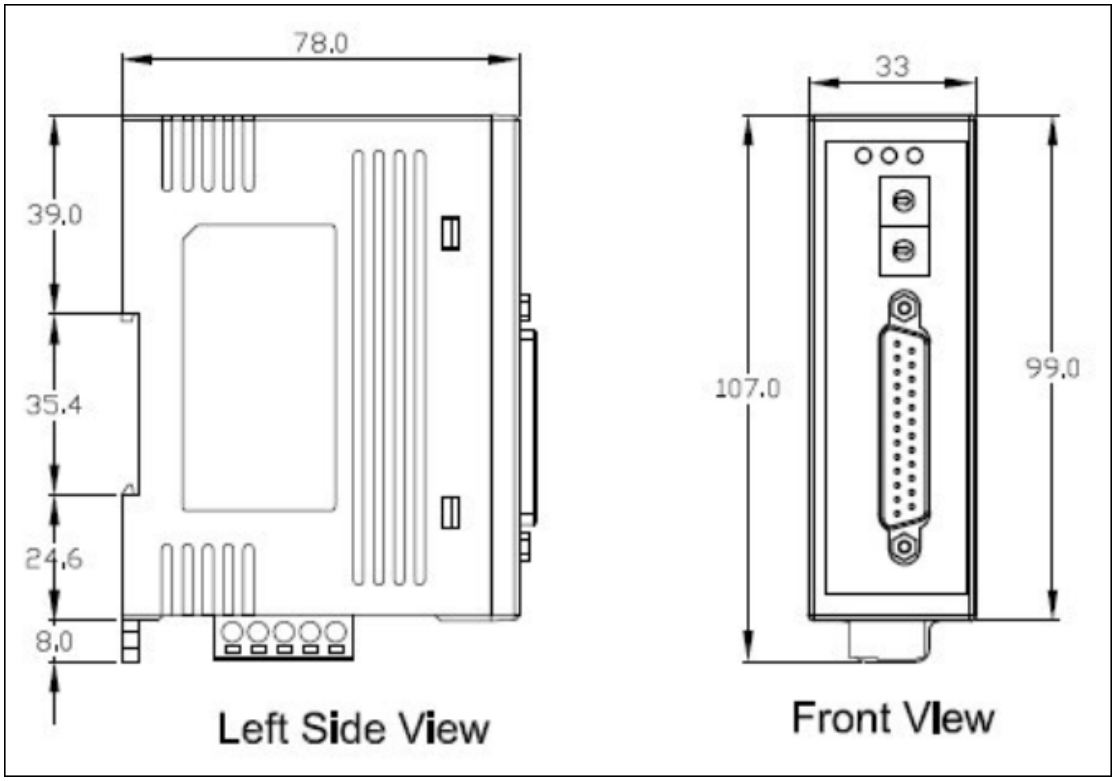
| IDENTIFIER BITS | | | | | | | | | | | RTR | Data Length | 8-byte Data (byte) (HEX) | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Source MAC ID | | | | | | | | | | | | | | | | | | |
| 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 6 | 0A | 8E | 00 | 80 | AB | 44 | -- | -- |

**Master** ← **Slave (CAN-2018D)**

Get the value of Application Object1 with Instance ID 0x01 and Attribute ID 0x04. After sending the “Get Attribute Single”, the slave device responds the temperature data. The value 0x44AB8000 is in IEEE 754 for the basic single floating-point format, means that the channel0’s data is 1372.0 ℃. About the range of the value, refer to the appendix B for the AI data and type code definition. (The read temperature function is supported in firmware version 1.2 or later)

# Appendix A: Dimension




**Unit: mm**

# Appendix B: Type Code Definition

Type Code Definition for CAN-2018D

| Type Code | Input Range | Data Format | Max Value | Min Value |
|---|---|---|---|---|
| 0x00 | -15 to +15 mV | Engineer Unit | +15 | -15 |
| | | 2's Complement HEX | 0x7FFF | 0x8000 |
| 0x01 | -50 to +50 mV | Engineer Unit | +50 | -50 |
| | | 2's Complement HEX | 0x7FFF | 0x8000 |
| 0x02 | -100 to +100 mV | Engineer Unit | +100 | -100 |
| | | 2's Complement HEX | 0x7FFF | 0x8000 |
| 0x03 | -500 to +500 mV | Engineer Unit | +500 | -500 |
| | | 2's Complement HEX | 0x7FFF | 0x8000 |
| 0x04 | -1 to +1 V | Engineer Unit | +1 | -1 |
| | | 2's Complement HEX | 0x7FFF | 0x8000 |
| 0x05 | -2.5 to +2.5 V | Engineer Unit | +2.5 | -2.5 |
| | | 2's Complement HEX | 0x7FFF | 0x8000 |
| 0x06 | -20 to +20 mA (with 125Ω resistor) | Engineer Unit | +20 | -20 |
| | | 2's Complement HEX | 0x7FFF | 0x8000 |
| 0x0E | J Type (℃) | Engineer Unit | +1200 | -210 |
| | | 2's Complement HEX | 0x7FFF | 0xE99A |
| 0x0F | K Type (℃) | Engineer Unit | +1372 | -270 |
| | | 2's Complement HEX | 0x7FFF | 0xE6D0 |
| 0x10 | T Type (℃) | Engineer Unit | +400 | -270 |
| | | 2's Complement HEX | 0x7FFF | 0xA99A |
| 0x11 | E Type (℃) | Engineer Unit | +1000 | -270 |
| | | 2's Complement HEX | 0x7FFF | 0xDD71 |
| 0x12 | R Type (℃) | Engineer Unit | +1765 | -50 |
| | | 2's Complement HEX | 0x7FFF | 0xFC60 |
| 0x13 | S Type (℃) | Engineer Unit | +1765 | -50 |
| | | 2's Complement HEX | 0x7FFF | 0xFC60 |
| 0x14 | B Type (℃) | Engineer Unit | +1820 | 0 |
| | | 2's Complement HEX | 0x7FFF | 0x0000 |
| 0x15 | N Type (℃) | Engineer Unit | +1300 | -270 |
| | | 2's Complement HEX | 0x7FFF | 0xE56B |
| 0x16 | C Type (℃) | Engineer Unit | +2320 | 0 |
| | | 2's Complement HEX | 0x7FFF | 0x0000 |

**Note:**

Getting the 0x7FFF value (UINT formate) or 0x7FFFFFFF value (REAL format) in all of the thermocouple type codes, it means that the thermocouple wires are broken.(The read temperature function is supported in firmware version 1.2 or later)

CJC (cold junction compensation) Definition for CAN-2018D

| Input Range | Data Format | Max Value | Min Value |
|---|---|---|---|
| -30 to +100 (℃) | Engineer Unit | +100 | -30 |
| | 2's Complement HEX | 0x03E8 | 0xFED4 |