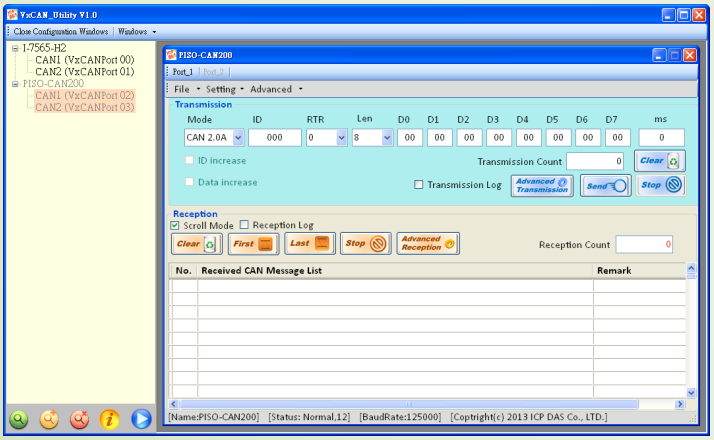
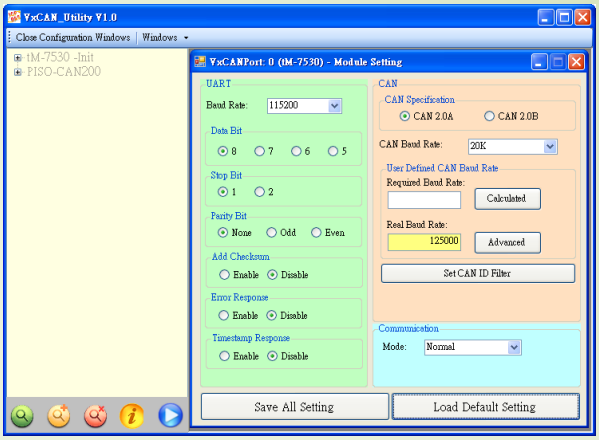
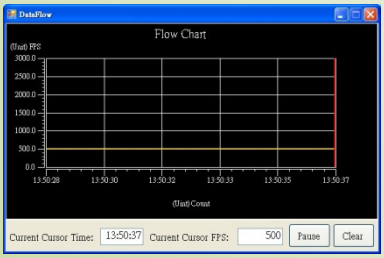
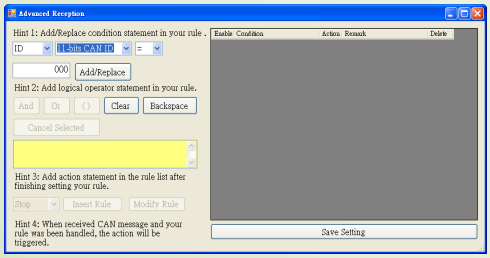
VxCAN Utility

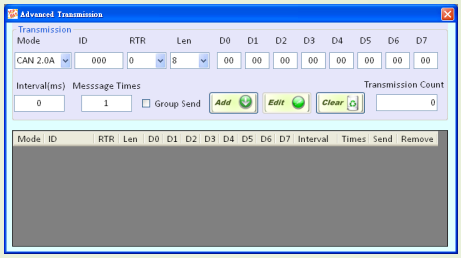
User's manual











- Warranty

  All products manufactured by ICP DAS are under warranty regarding 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 resulting from 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, or for any infringements of patents or other rights of third parties resulting from its use.

- Copyright

- Trademark

  The names used for identification only may be registered trademarks of their respective companies.

Revision & Hardware

Revision

Ver.	Date	Author	Description
1.3.1	2025/01/15	Terry	Fixed known bugs.
1.3	2024/09/16	Terry	Chart feature added. Fixed known bugs.
1.2	2024/07/01	Terry	Added new hardware. Optimized user interface. Fixed known bugs.
1.1	2022/04/28	Johney	Update the searching description.
1.0	2014/07/31	Alan	First edition

Supported Hardware
~~1. PISO-CAN200/400~~
2. PISO-CAN100U/200U/400U/800U
3. PEX-CAN200i
~~4. PCM-CAN200(P)~~
5. I-7530 / I-7530T
6. tM-7530(A)
7. I-7530A
8. I-7530FT
9. I-7530A-MR
10. I-7540D
11. I-7540D-MTCP
12. I-7565
13. I-7565-H1/H2
14. tM-7565
15. I-7565M-HS
16. I-7565M-FD
17. PISO-CAN200U-FD/PISO-CAN400U-FD

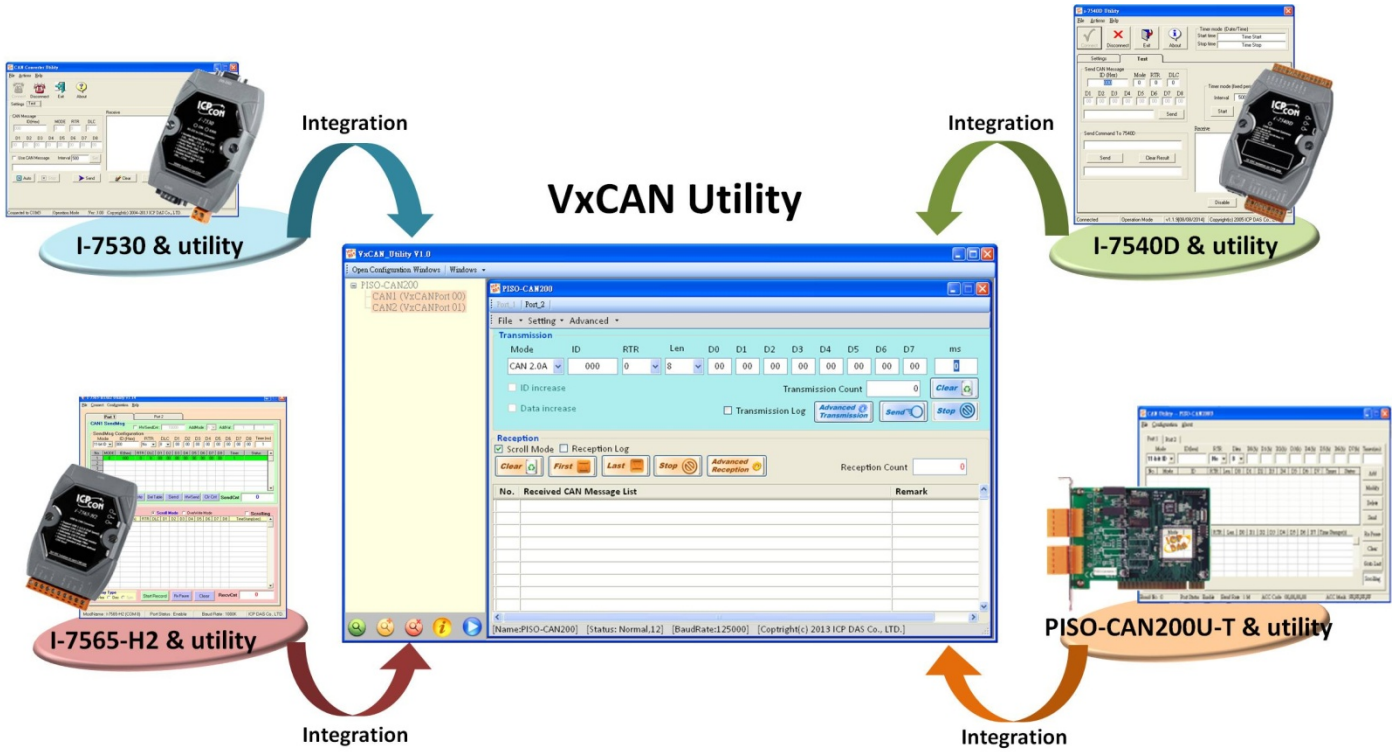
Contents

1. Introduction

1.1 Overview

The VxCAN_Utility, a configuration, diagnostic and test software tool, is designed to unleash the power of all CAN products of ICP DAS. It is based on the Virtual CAN technique which creates the virtual CAN ports to match the physical CAN interfaces of the installed products. By means of the VxCAN_Utility, it is not necessary to adapt the various utilities or software interface due to the different CAN products. Users just need to focus on the project development and network diagnosis without concerning about what CAN product the system uses.

In order to satisfy in various applications, the VxCAN_Utility provides not only the functions of sending and receiving CAN messages, but the functions for message trigger, group transmission and data record. Users can set some trigger conditions in the VxCAN_Utility. If these conditions are reached, it will activate the specific messages, such as stopping the message reception, starting the data log or sending some specific messages. The group transmission function allows users to arrange a group of messages. Users can arrange the CAN messages in advance which must be sent sequentially, and send the group by manual trigger or some specific rules. It is useful when users would like to control or configure the slave devices for testing. The reception messages of the VxCAN_Utility are able to be saved in the file after enabling the function of message record. This can help users to know what happen in the system network clearly so that the system problems would be fixed more efficiently and simply.



1.2 Features

- Integrates all CAN converters in the utility.
- Searches all supported CAN products installed in the system automatically.
- Provides the basic functions to send and receive CAN messages.
- Allows sending a predefined group of CAN messages.
- Provides CAN ID filter to sieve out the CAN messages which are not interested.
- Supports the configuration of the trigger event for starting transmission, stop reception or recording the messages.
- Presents the CAN Bus loading by the trend.
- Shows the status of the corresponding CAN controller of the CAN product.
- CAN Message log functions.
- Supports J1939, CANopen and DeviceNet protocol.

1.3 Applications

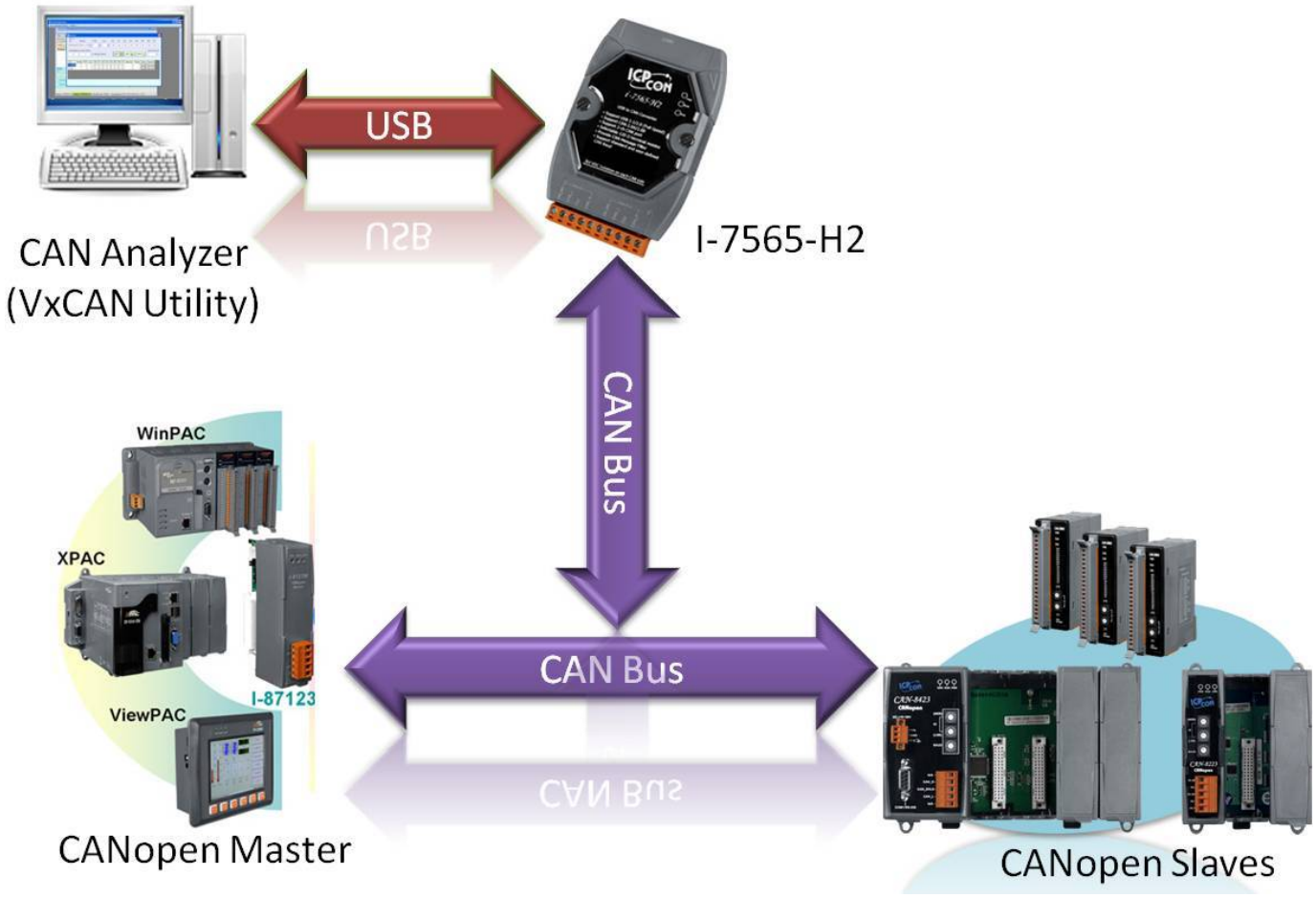
Works as a CAN Master:

The VxCAN_Utility provides basic sending and receiving functions. Through these functions, the CAN product, for example the I-7530, can work as a CAN master to control the CAN slave device as the following figure. Users just need to use the VxCAN_Utility to send the corresponding commands of the CAN slave devices, the responses from the slave devices are shown in the reception list of the utility. If there are

several commands needed to be sent sequentially, users can use the Group Send function with predefined delay time to access the slave devices.

Works as a CAN Analyzer:

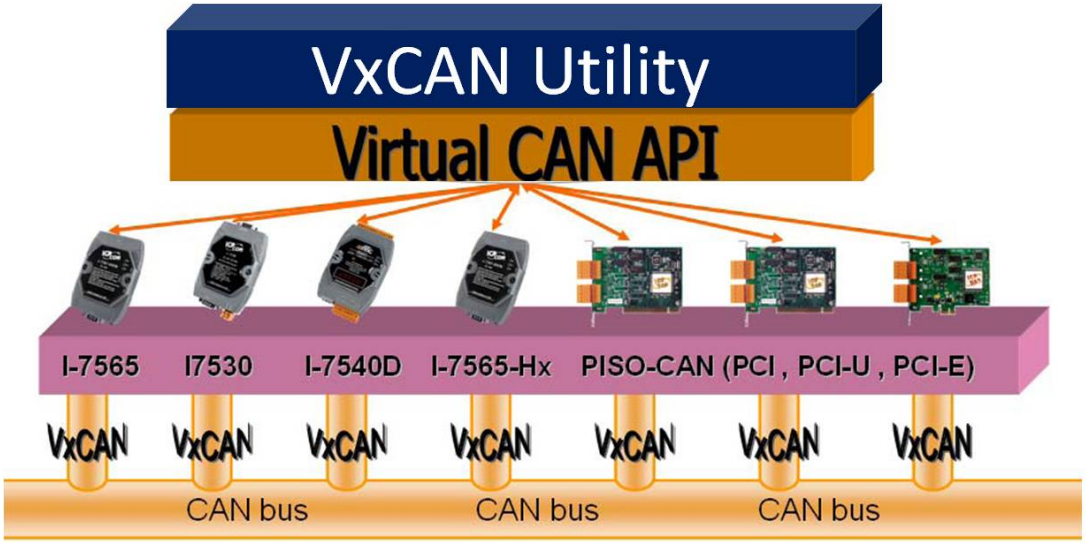
The VxCAN_Utility could be a compact analyzer. When users would like to diagnostic the CAN network, the VxCAN_Utility provides several useful functions, such as event trigger, data log and Bus loading trend. The event trigger can used to trigger the message transmission, data log or stop of messages reception. Users just need to use a CAN product such as I-7565-H2, the utility would be helpful users to monitor the communication of the network, access the CAN nodes, or diagnose the CAN Bus loading.



2. Architecture and Requirements

2.1 Software Architecture

The VxCAN_Utility is designed based on the Virtual CAN driver. Before using the VxCAN_Utility, the Virtual CAN driver must be installed first. It can be downloaded from the following website Virtual CAN Driver. The Virtual CAN driver also provides APIs, demos and users' manual for developing the CAN applications by themselves. Through the Virtual CAN technology, all physical CAN interfaces will be mapped to the corresponding virtual CAN ports. Users can easy deploy the portable software on the different CAN devices. The software application architecture is as follows.



2.2 System Requirements

■ **Hardware**

CPU: Dual Core 2.4GHz or higher.

RAM: 2GB or higher.

Note: The resolution 1280 x 768 pixels of the screen is recommended.

■ **Software**

Runtime: Microsoft .NET Framework 3.5 or later.

Virtual CAN driver: Version 3.0 or later.

Platform: Window7/10/11 32-bit/64-bit.

3. Software Installation and Operation

3.1. Installation

The installation of the .Framework 3.5 (or later) and virtual CAN driver 3.4 (or later) is necessary. Users can download them from following website.

- Microsoft .Net Framework Version 3.5:
  http://www.microsoft.com/en-us/download/details.aspx?id=22
- Virtual CAN driver:
  https://www.icpdas.com/en/download/index.php?nation=US&kind1=&model=&kw=VxCAN

After finishing the installation of the .Net Framework, users can start to install virtual CAN driver. During the installation procedure of virtual CAN driver, the virtual CAN related files including VxCAN_Utility will be installed into PC.

About the related installation method of virtual CAN driver, please refer to the virtual CAN driver user's manual section 1.2 "Virtual CAN Driver Installation".

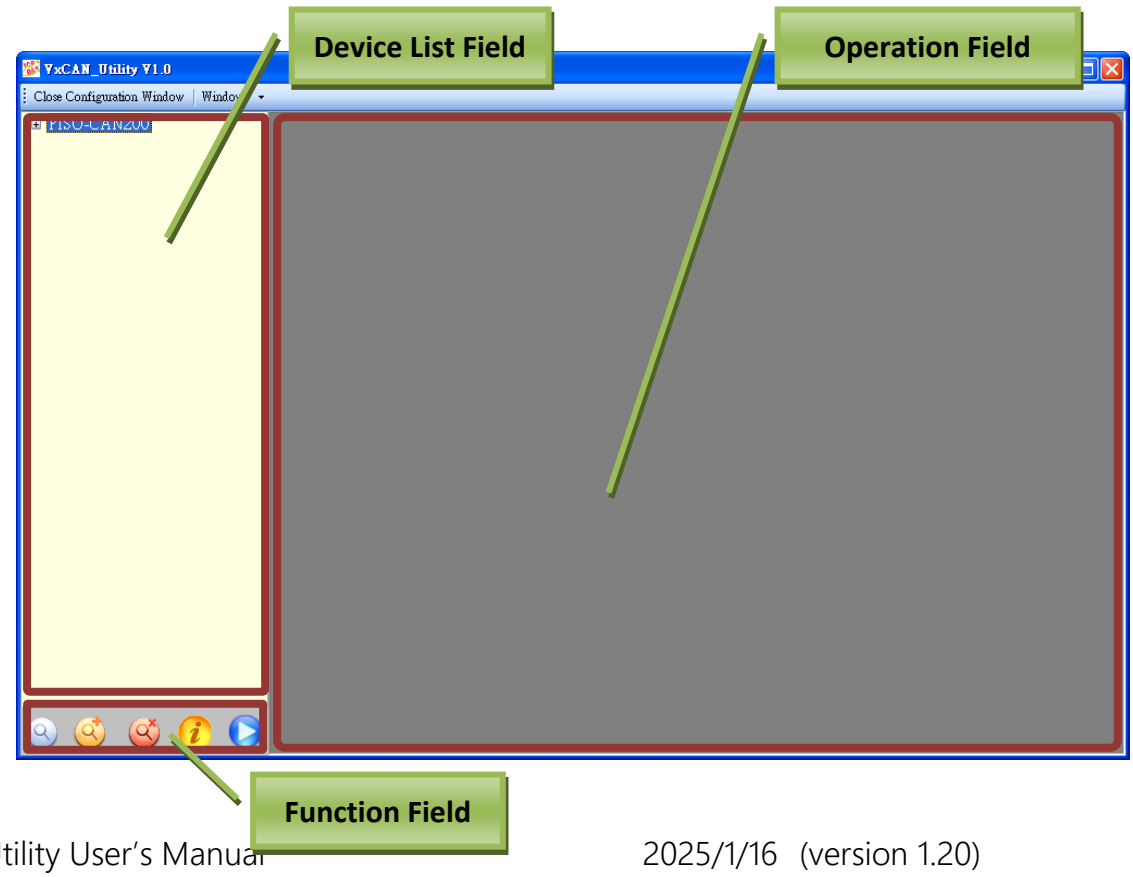
3.2. Software Function Description

After finishing the installation, the default location of the execution file will be as following path. **"C:\ ICPDAS\VxCAN_Utility\ VxCAN_Utility.exe".**

3.2.1. Information Page

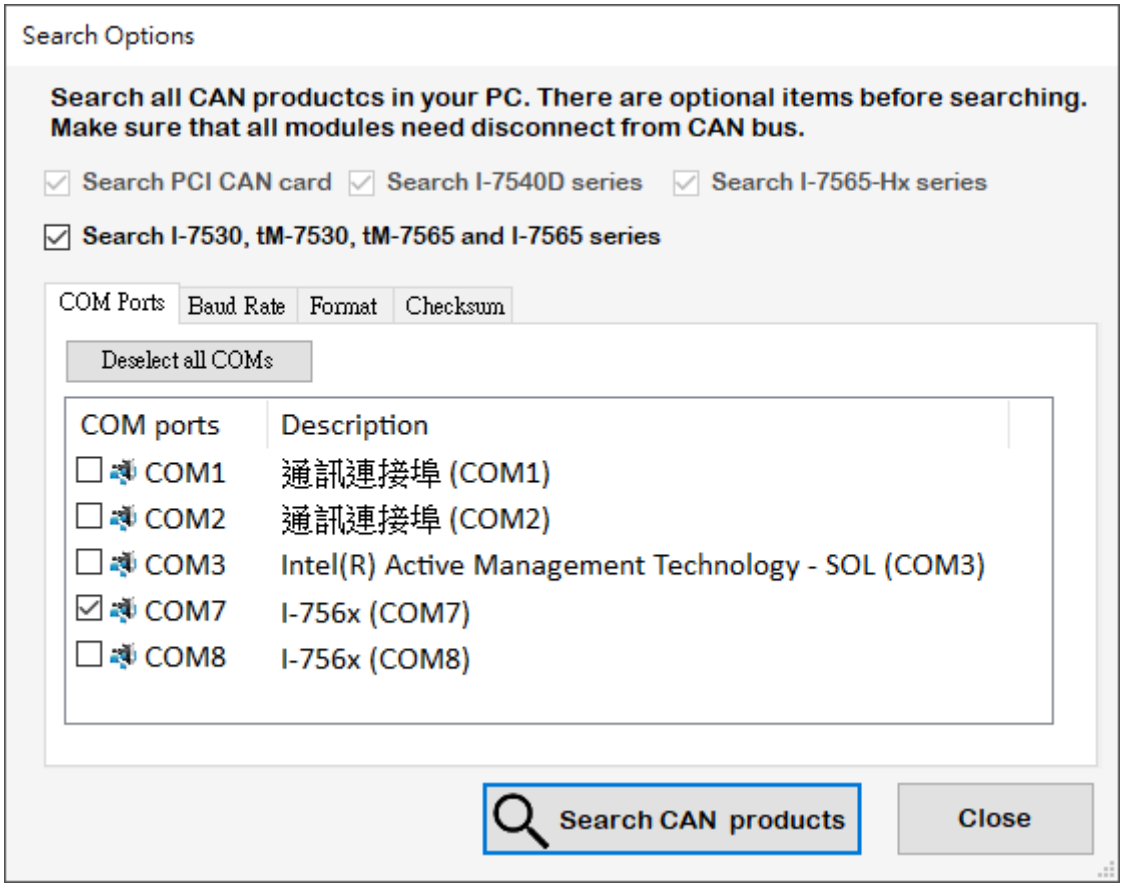
The following figure is the main screen of the VxCAN_Utility. When first time to run the VxCAN_Utility, the VxCAN_Utility will scan the CAN interface connected with the PC. Afterwards, the VxCAN_Utility saves the result, and doesn't scan again when the Utility are opened next time. Therefore, if users add or change the CAN interfaces, uses the search button to scan the available CAN interface again. The main screen is divided into three parts, "Function Field", "Device List Field", and "Operation Field".

- **Function Field:** In this field, there are five buttons. Those buttons are used for searching CAN interfaces, showing information, and setting the advance search options.
- **Device List Field:** This field displays the information of the searched CAN devices. Users also can configure the parameters of the CAN devices here.
- **Operation Field:** After finishing the configuration and enable the CAN ports of the CAN device, this field provides several tools for network diagnosis and monitor. Users can use these to send, receive, record, or display the relation information of the CAN messages.



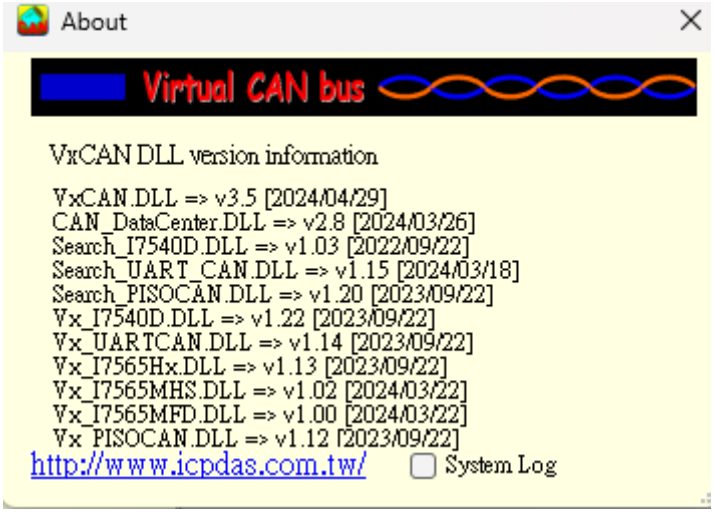
◆ **Function Field:**

➢ **Search** : Click this button to search all of the CAN devices connected to PC if there is any CAN device removed from or connected to the PC first time. After searching, the CAN devices list will be saved in PC. The users can get the list without searching. After clicking the button, the search options dialog is popped up as the below figure. Users can select the proper COM ports , COM port baud rates, Data-bits and etc. When starting to search, the VxCAN_Utility will follow the options to search the CAN devices connected with the PC.



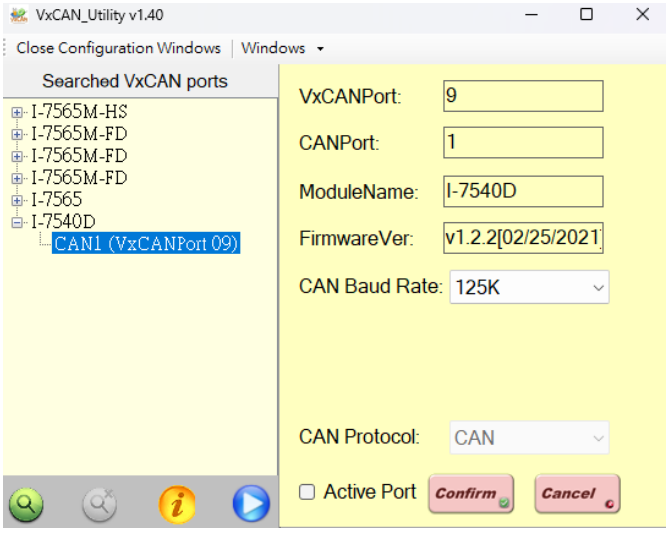
➢ **Stop Searching** : Users can use this function to stop the search process.

➢ **Software Information** : Show the version information of the VxCAN DLL version as the following figure. The check box System Log is used for logging Status of DLL under VxCAN of the VxCAN_Utility. It is helpful for engineer to debug the VxCAN DLL.

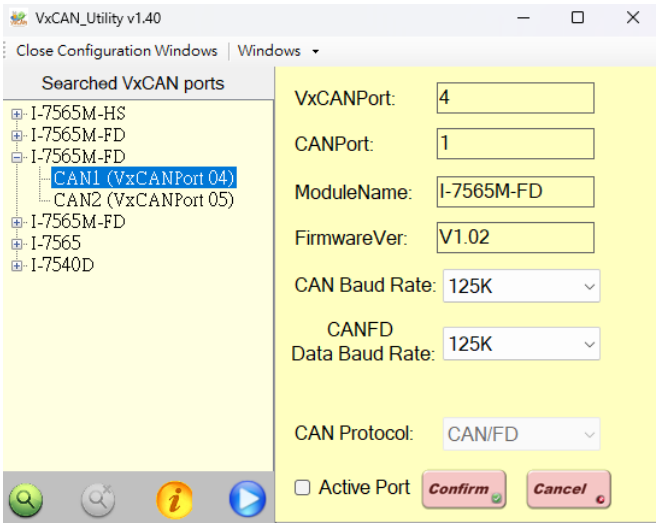


➢ **Run the CAN Port** : When users select the CAN port in the Device List Field (see the following section) done, click this button to run the VxCAN_Utility on this CAN port. Afterwards, the tools for sending, receiving or recording CAN messages will be popped up in the Operation Field.

◆ **Device List Field:**



1. CAN only device



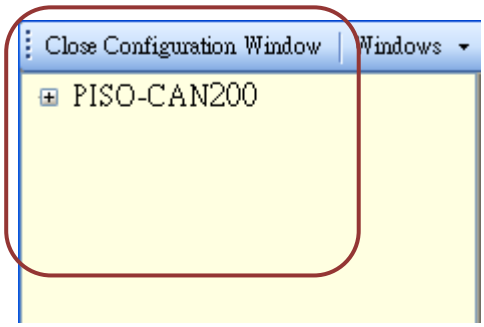
2. CAN FD Device

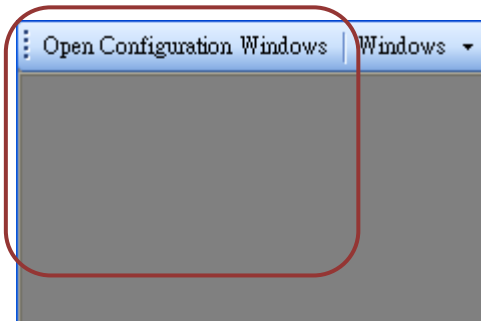
This field lists all modules which had been searched or recorded in the configuration file. Select the CAN port of the listed module to show the configuration information of the port. The details of the configuration information are described as below.

➢ **VxCAN Port:** After finishing the search, the Virtual CAN Driver allocates virtual CAN ports to each searched physical CAN port of the ICP DAS CAN devices. Users can use the corresponding virtual CAN port No. to send/receive CAN messages.

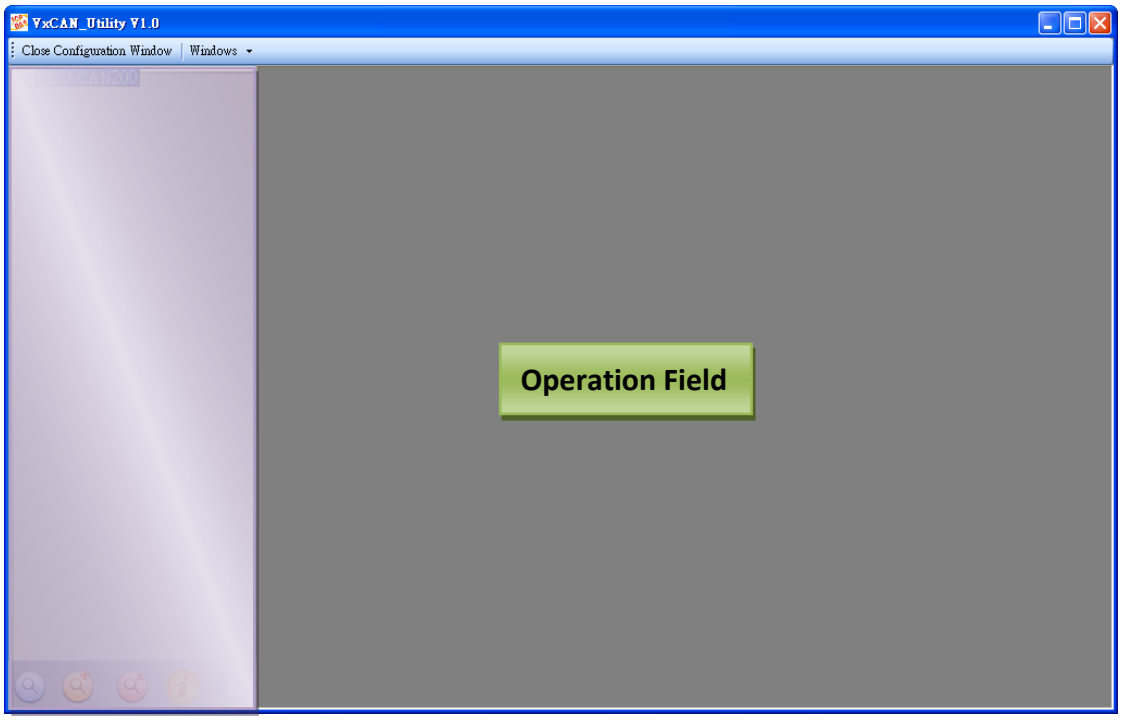
- **CAN Port:** This is a physical CAN port No. of the CAN device.
- **Module Name:** The name of the CAN device.
- **Firmware Version:** The firmware version of the CAN device.
- **CAN Baud Rate:** The configuration of the CAN baud rate to the selected CAN port.
- **CAN FD Baud Rate (CAN FD Data Baud Rate):** Indicates the data field baud rate setting for the selected CAN port.
- **CAN Protocol:** Represents the types of CAN protocols supported by the device. Currently supports CAN and CAN FD.
- **Active Port:** Click the checkbox will activate the selected CAN port. Afterwards, when uses click the button  in the function field, this CAN port can be used to access the CAN network. Users can also activate multi CAN ports for different applications.
- **Confirm button:** Click this button to activate the CAN configuration of the selected CAN port.
- **Cancel button:** Click this button to ignore the CAN configuration of the selected CAN port.

If users would like to adjust the section of the Operation Field, use “Close Configuration Window” or “Open Configuration Window” to do this as the below figure.



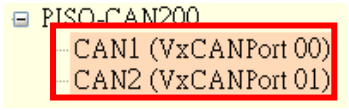


◆ **Operation Field:**

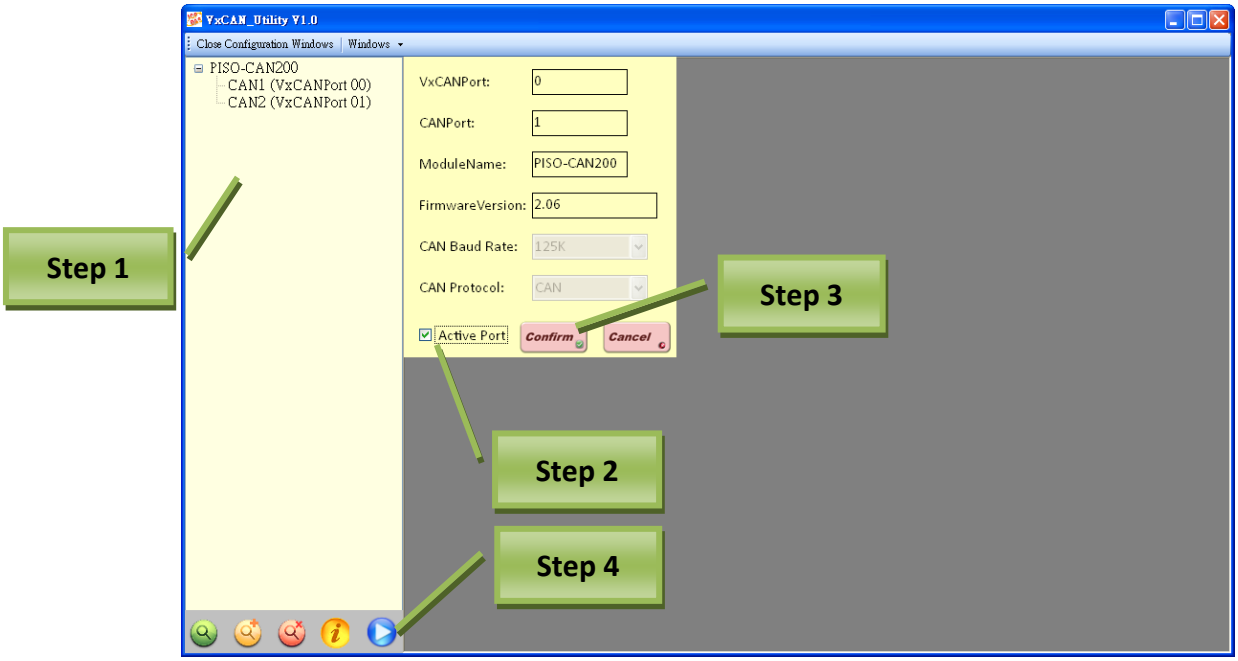


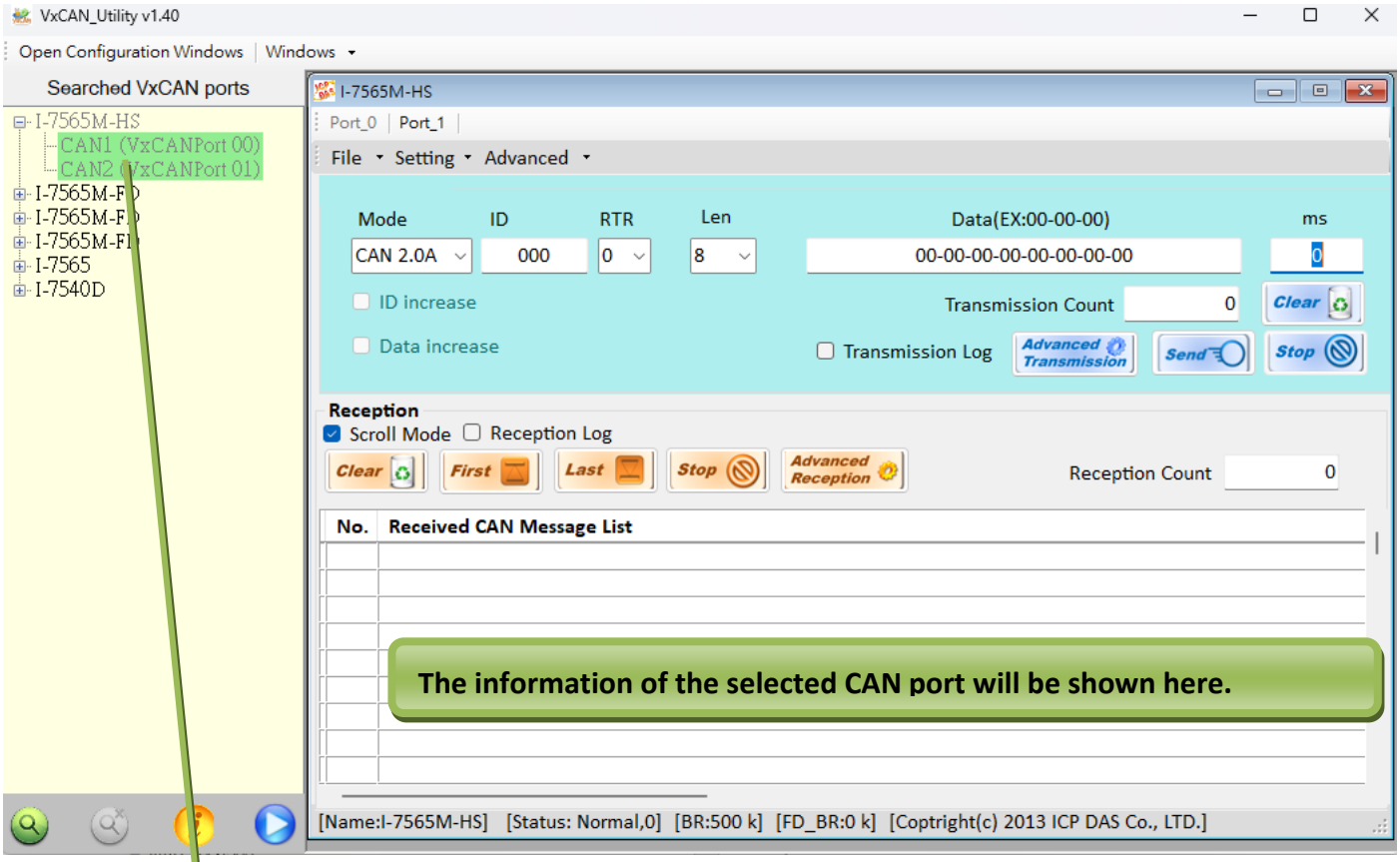
This field is used to operate each CAN device, the following steps describes how to use the functions to access the CAN network.

- **Step 1:** Select the CAN port of the CAN device used to access the CAN network.
- **Step 2:** Set the CAN baud rate and protocol. Then, check the checkbox of the Active Port to activate the CAN port, and click Confirm button to continue.
- **Step 3:** Afterwards, the activated module will be highlight on the device list.



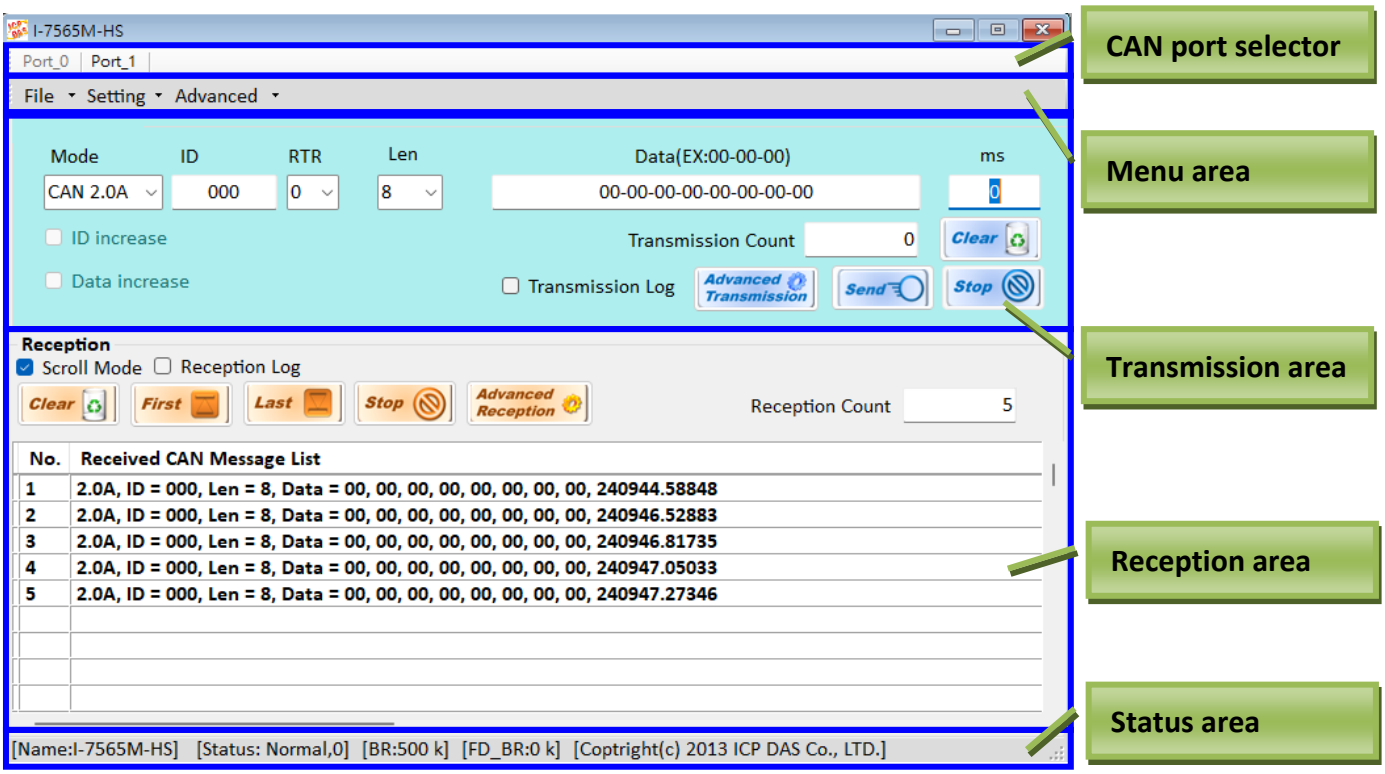
- **Step 4:** Click the button  , then the operation page is popped up in the operation field.





While one of the CAN port of the device has been activated and the operation page is popped up, the CAN ports in the list will be displayed in green. If activation fails, they will be shown in orange-red.

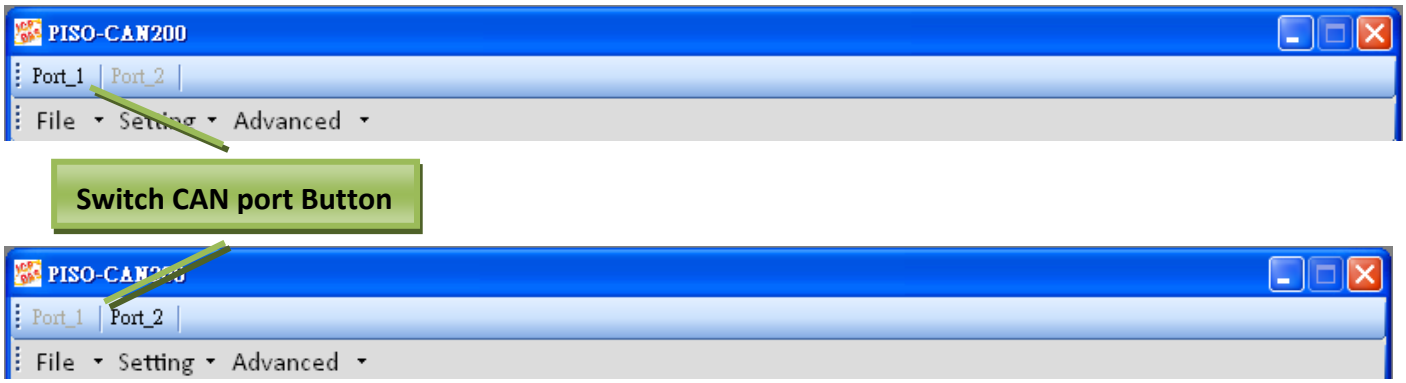
3.2.2. Operation Page



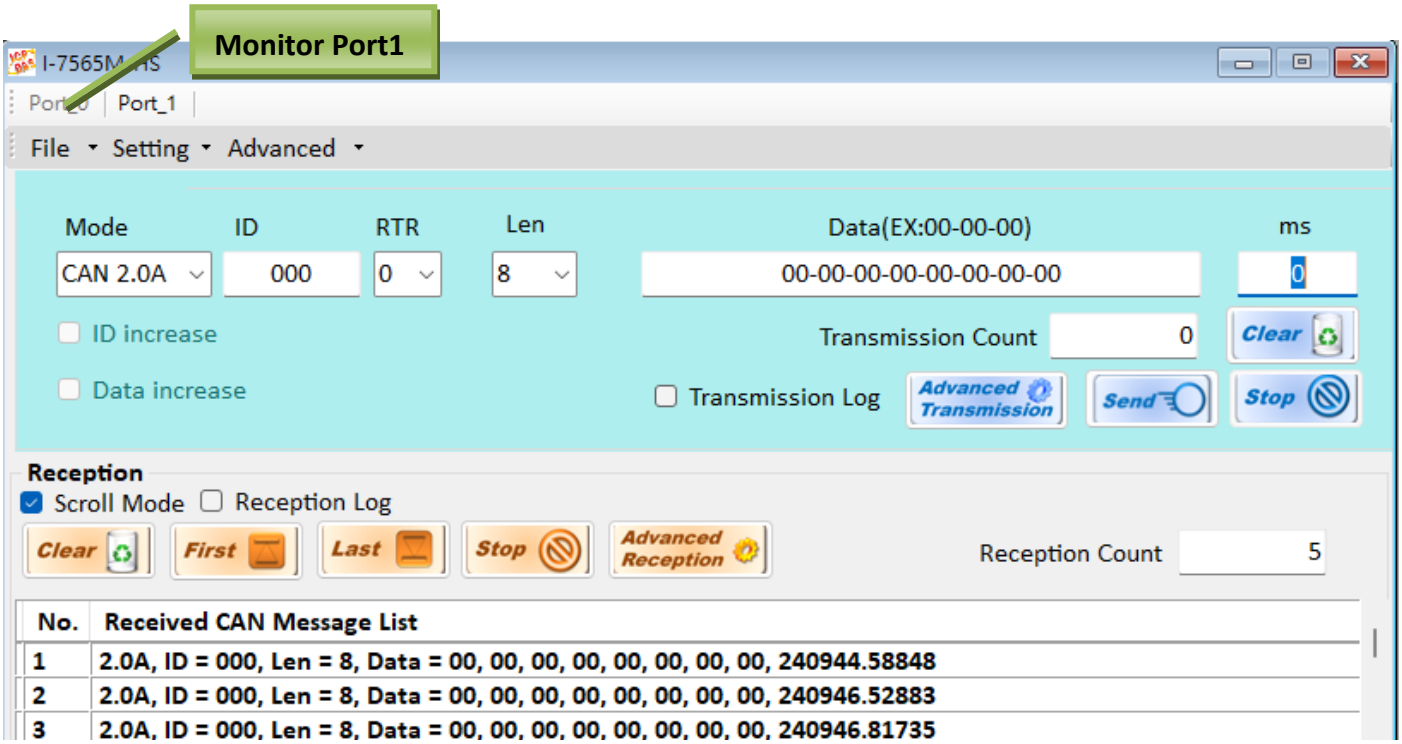
The functions in the operation page are described below:

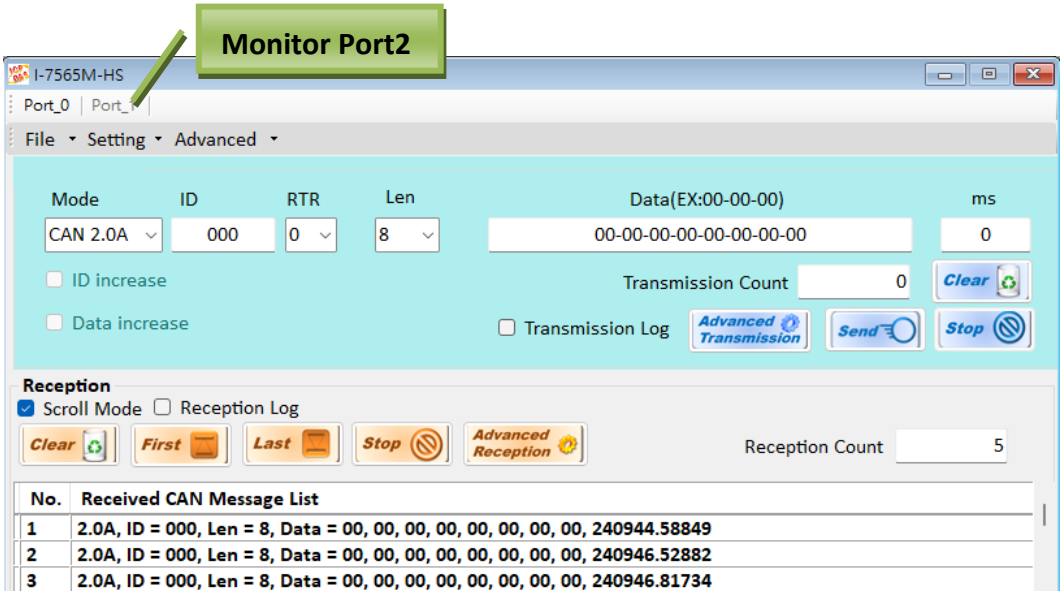
- **CAN port selector:** Users can use the CAN port selector to switch the No. of the CAN port which will be operated in the operation page.
- **Menu area:** This field is composed of three items which are including “File”, “Setting”, and “Advanced”. The functionality of each item is described in detail in the later section.
- **Transmission area:** Users can use this to configure the messages which will be sent to the CAN Bus. It includes the functions of the normal transmission and special transmission which allows the VxCAN_Utility to send a group of relative CAN messages.
- **Reception area:** The CAN messages got by the VxCAN_Utility will be shown in this field. User can set trigger condition, such as stopping the message reception, starting the data log or sending some specific messages, for the reception mechanism.
- **Status area:** The CAN device information will be shown in this field. It includes Module Name, Module Status, and Current CAN baud rate.

CAN port selector:



Users can switch the tag to change the activated virtual CAN port of the selected CAN devices for accessing the CAN network.

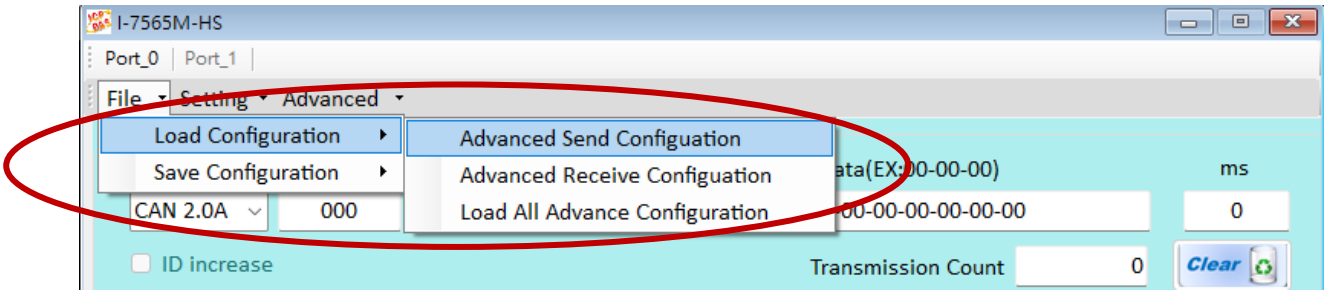




Menu area:

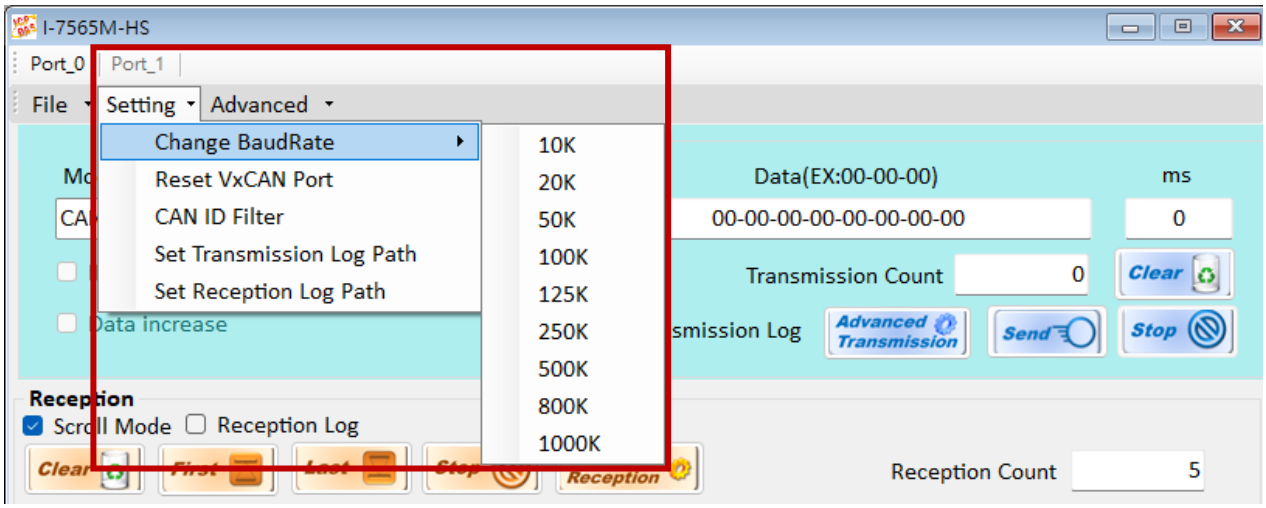
- **File:**

  Function of “Save Configuration” and “Load Configuration”. Users can save “Advanced Send/Receive Configuration” after setting done. Or, users can load “Advanced Send/Receive Configuration”.



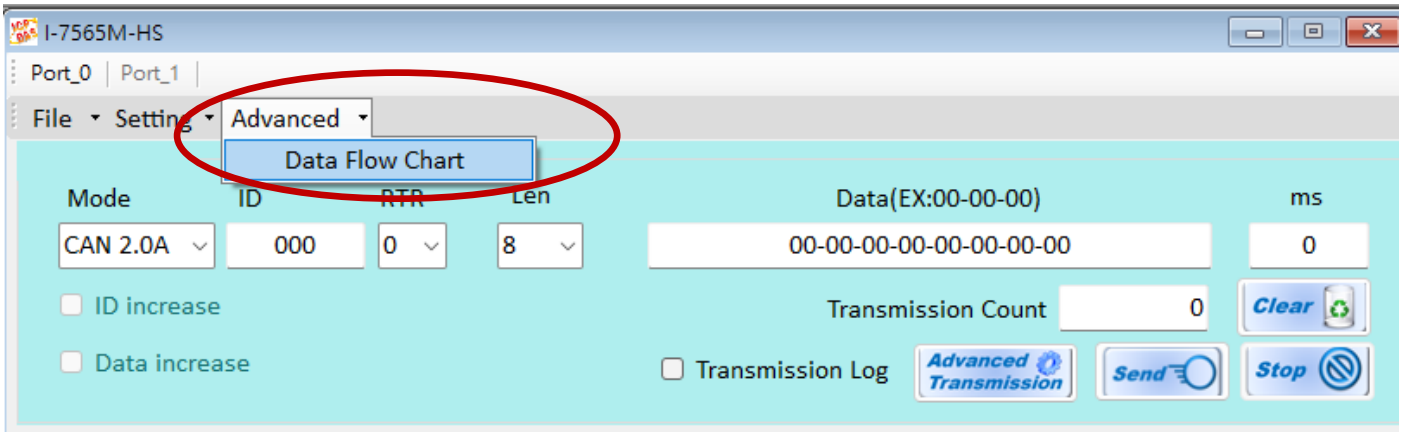
- **Setting:**

  Function of “Change Baud rate”, “Reset CAN port”, “Software CAN ID filter”, and “Set Transmission/Reception Log Path”.

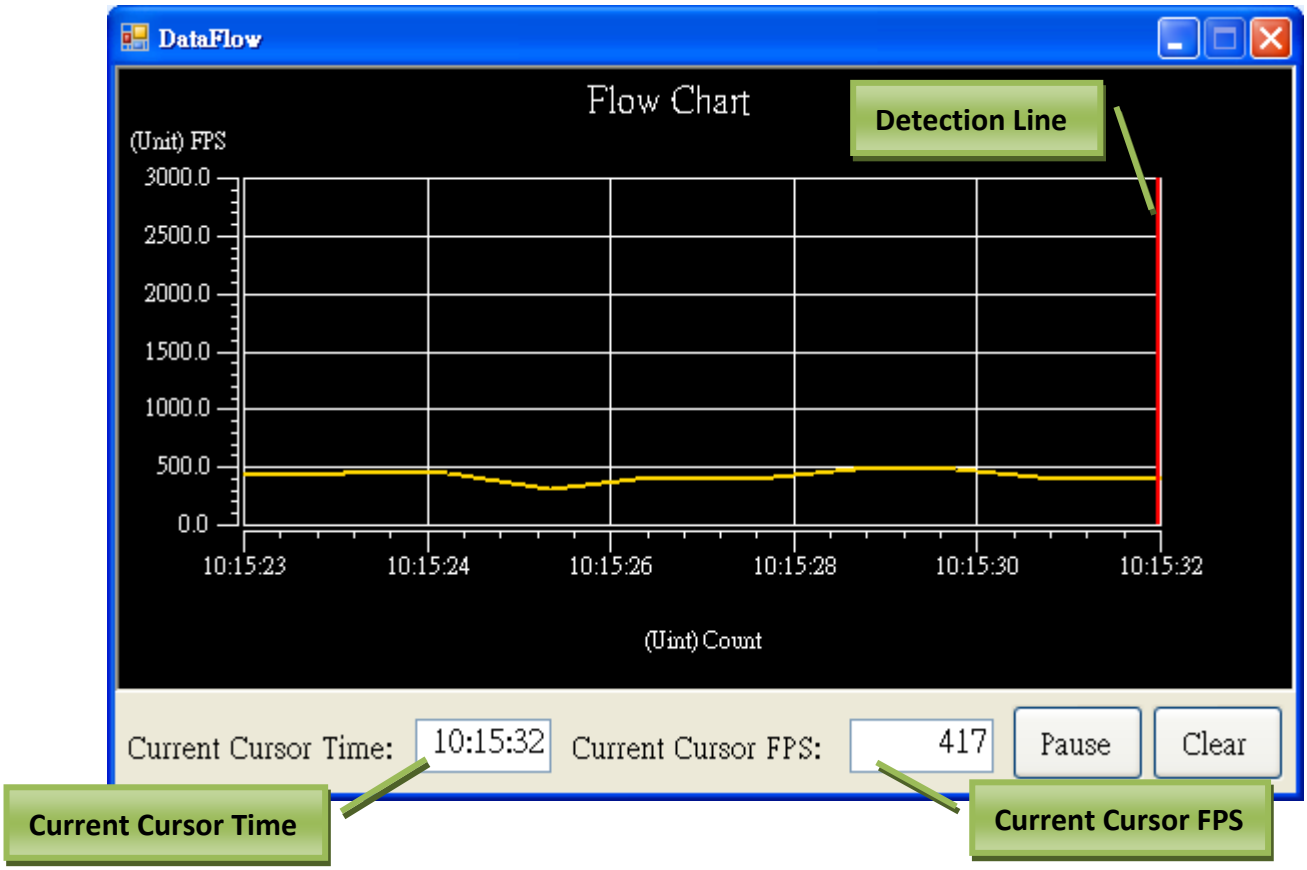


- **Advance:**

  Function of “Data flow chart”. Users can use it to detect Bus loading of each CAN port by the trend.



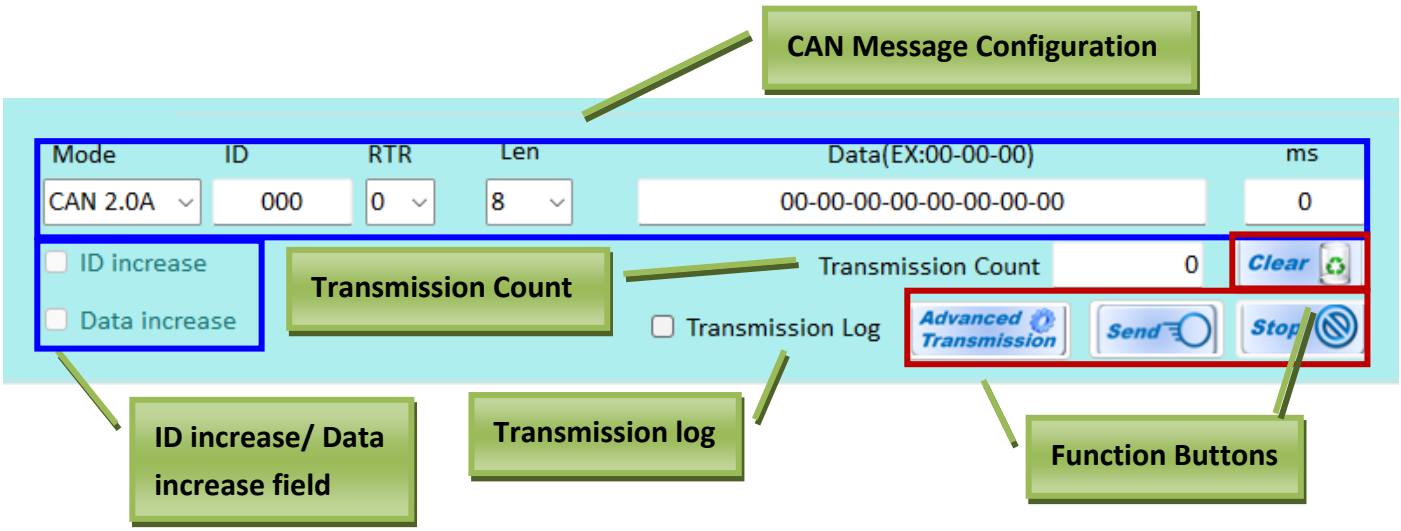
When users open the Data Flow Chart, the instantaneous CAN Bus loading is shown. It is very useful to diagnosis the CAN network. If the Bus loading is over the capability of processing CAN messages of the device, some CAN messages may be lost.



1. **Current Cursor Time:** This field indicates the current cursor X-axis position which is shown as time format value.
2. **Current Cursor FPS:** This field indicates the current cursor Y-axis position which is shown as FPS (Frame Per Second). It also means Bus loading.
3. **Detection line:** Users can drag this line to show the X-axis and Y-axis position.
4. **Pause button:** Pause to detect Bus loading
5. **Clear button:** Clear records and reset detection.

3.2.2.1. Transmission Functions

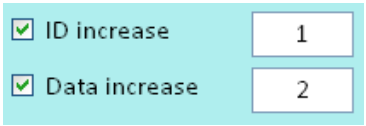
This section illustrates how to use the transmission functions. The following picture is the screen of the transmission functions.



- **CAN Message Configuration:** This field is used for filling the data of CAN message which will be transmitted to the CAN network, such as CAN ID, Mode, RTR, Data Length, data, and ms. The "ms" field is a period value (unit of millisecond), and is useful if the CAN messages will be sent cyclically. If the CAN message just need to be sent once, keep the ms field to zero.

  ※ We recommend that the minimum value of ms field is 10 ms.

- **ID increase/ Data increase field:** Decide whether the transmitted CAN message will increase the value of ID or Data. This field is only used when the CAN message is transmitted cyclically (i.e. the value of the ms field in the CAN message configuration can't be zero). When ID increase function or Data increase function is enabled, the users can set increase the value of ID or Data to 1 or more, such as following graph.

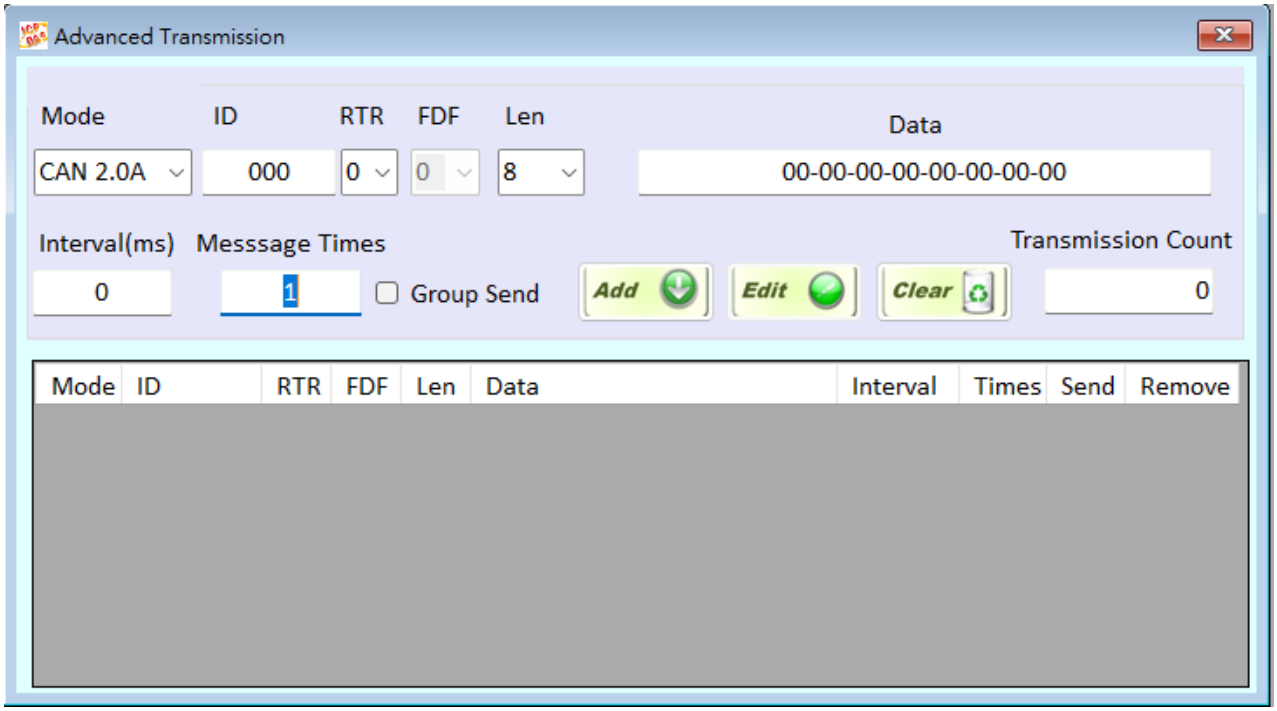


- **Transmission Count:** When the VxCAN_Utility sends a CAN message, the value of the transmitted number will be increased to 1. Click the "Clear Button" will reset this value.
- **Transmission log:** Checking this checkbox to record the transmission history. Each transmitted CAN messages will be saved into a CSV file. The path of the file can be modified in the "set transmission log path" in the menu.
- **Function Buttons:** Including "Send", "Stop", "Clear", and "Advanced Transmission" buttons.
  - **Send:** Click this button for sending CAN messages described in the CAN Message Configuration.
  - **Stop:** When VxCAN_Utility sends CAN messages cyclically, user can click this button to stop sending.
  - **Clear:** Reset the value of "Transmission Count" to zero.

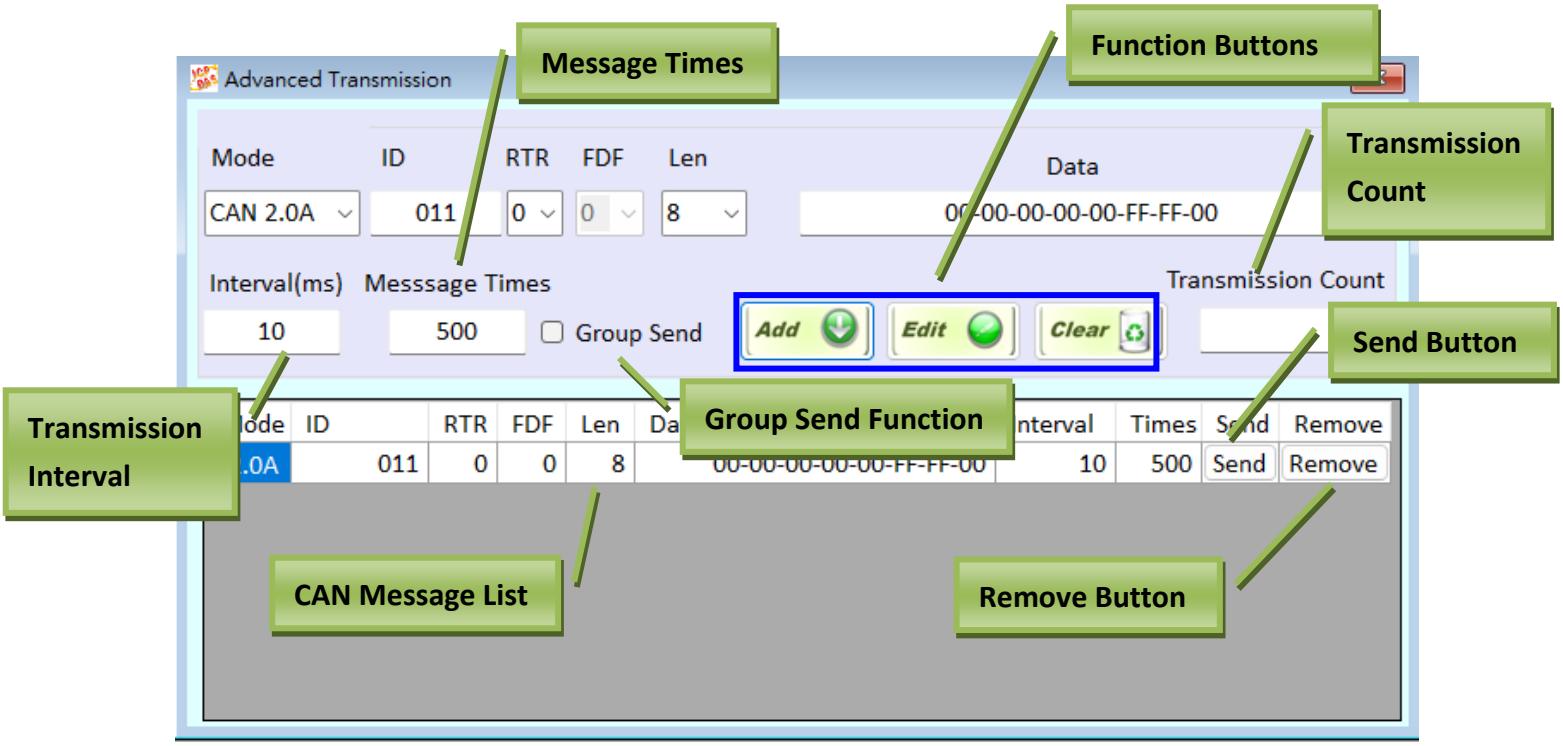
- **Advanced Transmission:** Use advance functions to send CAN messages. When users click this button, the “Advanced Transmission” dialog will be popped up. At this time, the functions described in this section are useless.

3.2.2.2. Advanced Transmission

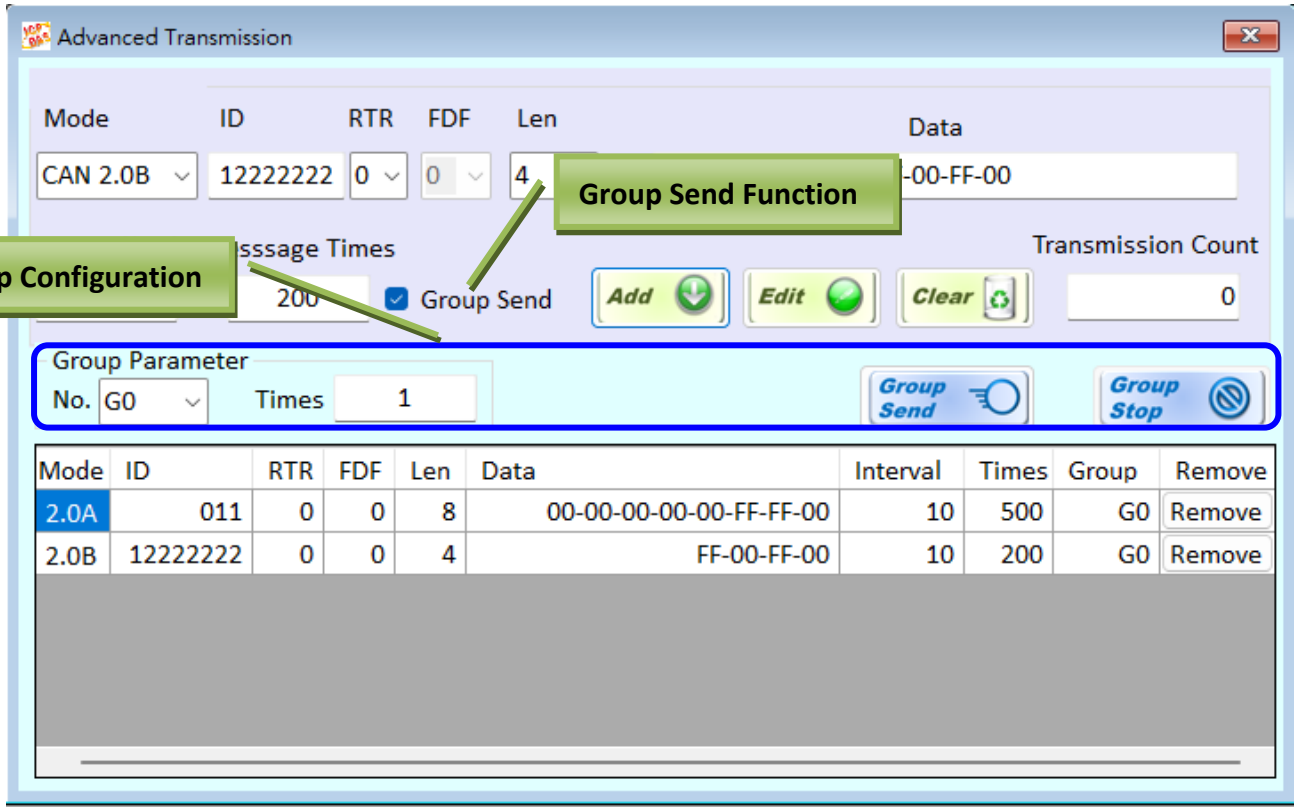
After clicking the Advanced Send button, the dialog as below is popped up.



The “Advanced Send” allows users to send CAN messages directly or organize some CAN messages as a group for sending. Here is the detail about how to send CAN messages directly.



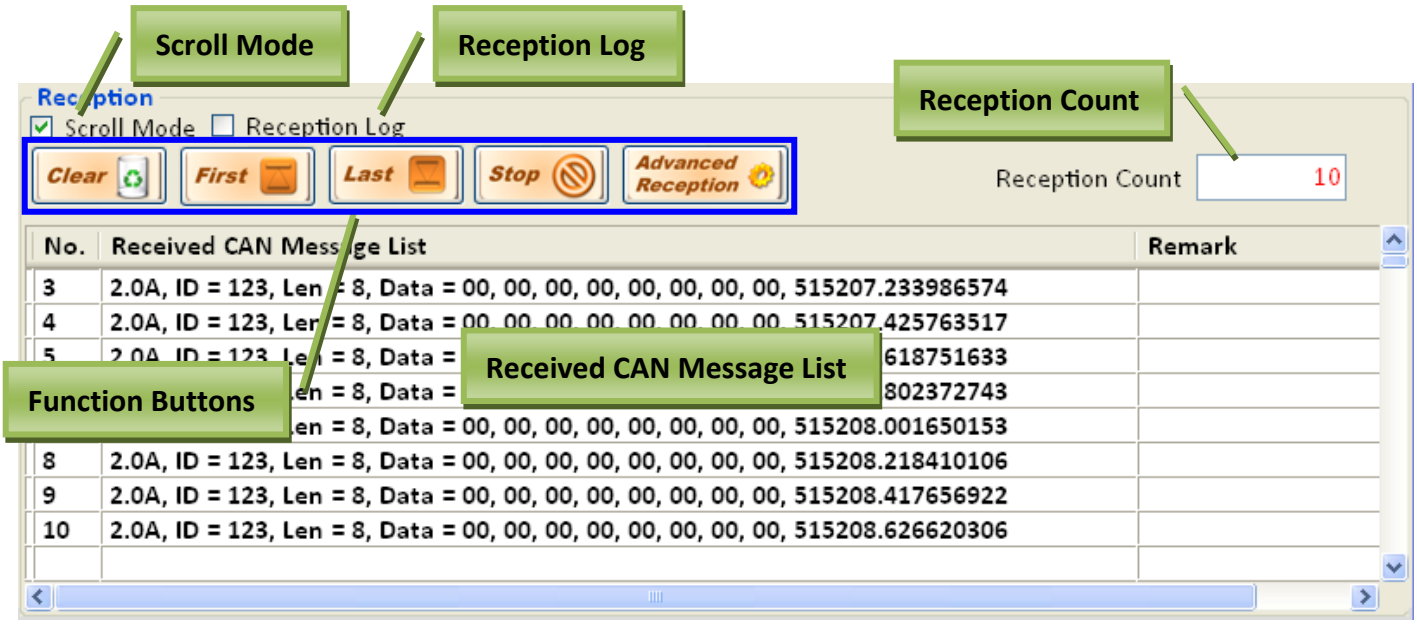
- **Interval (ms):** Time interval between two sending CAN messages. It also indicates the waiting time after sending this CAN message. It is only useful when the value of the Message Times is more than 1, or when the CAN message is applied by Group Send function.
  ※ We recommend that the minimum value of ms field is 10 ms.
- **Message Times:** Number of total CAN messages which will be sent repeatedly.
- **Function Buttons:** Including “Add”, “Edit”, and “Clear” buttons.
  - **Add button:** Add the CAN message into the CAN Message List. The maximum limitation of list is 65535.
  - **Edit button:** Edit the selected CAN message in the message list. After selecting one message from “CAN Message List”, this message will be shown on the Transmission Frame. Users can confirm the modification by clicking the “Edit” button.
  - **Clear button:** Reset the value of the “Transmission Count” to zero.
- **Transmission Count:** The number of CAN messages sent by the VxCAN_Utility.
- **Send Button:** Send a CAN message listed in the CAN Message List.
- **Remove Button:** Remove a CAN message from the “CAN message List”.
- **Group Send Function:** If users would like to organize some CAN messages as a group for sending, check this checkbox. Then, the Group Configuration will be shown under the Transmission Frame as below. Users can configure the parameters of the group here. The “Group Send” is a special function used for sending multiple CAN messages with the predefine schedule. Maximum 255 groups are supported.



- **Group No:** The group identification for the CAN message. Each CAN message must have a group ID. The messages which have the same group ID will be regarded as the same group.
- **Group Times:** The re-transmission times of the group. All of the CAN messages in the same group will share this value.
- **Group Send:** Start to send the messages in the group by predefine schedule. The Group Send does not support transmission log function.
- **Group Stop:** Stop sending all messages of the group.

3.2.2.3. Receive Function

The following figure is a reception screen of the VxCAN_Utility. The “Reception” frame is used for display the CAN messages that Utility received.

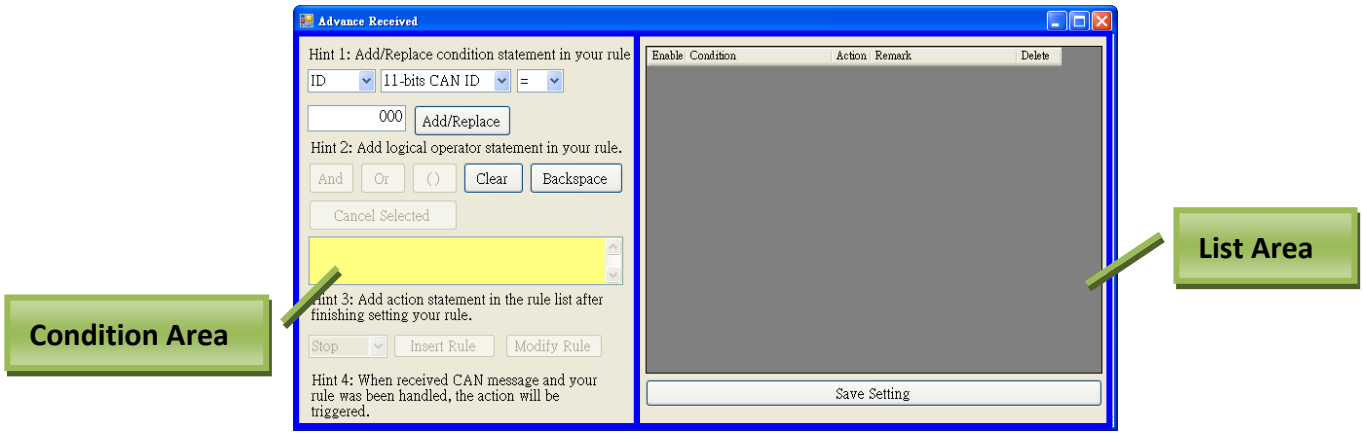


- **Scroll Mode:** Enable/disable the scrolling mode of the Received CAN Message List. If users need to stop scroll message list, don’t check this checkbox.
- **Reception Log:** Enable/disable the function to log the received CAN messages.
- **Reception Count:** The number of received CAN messages is shown here. Clicking Clear button can reset this value to zero.
- **Function Buttons:** Including Clear, First, Last, Stop, and Advance Receive buttons.
  - **Clear**  : Reset the “Reception Count” to zero, and clear the Received Message List.
  - **First**  : Jump to the first record of the Received CAN Message List. Before using this button, please disable the scroll mode.
  - **Last**  : Jump to the last record of the Received Message List. Before using this button, please disable the scroll mode.
  - **Stop**  : Stop to receive CAN messages. After clicking this button, the label of this button is toggled to Start. If users want to start receiving CAN messages, they click this button again.
  - **Advanced Receive**  : Click it to open the advanced reception interface. The details will be illustrated at the next section.
- **Received CAN Message List:** All of received CAN messages will be show here. This is a ring buffer, if the number of received message is over than 20000, the VxCAN_Utility will keep the last 20000 records.

3.2.2.4. Advanced Reception

After clicking the Advanced Reception button 

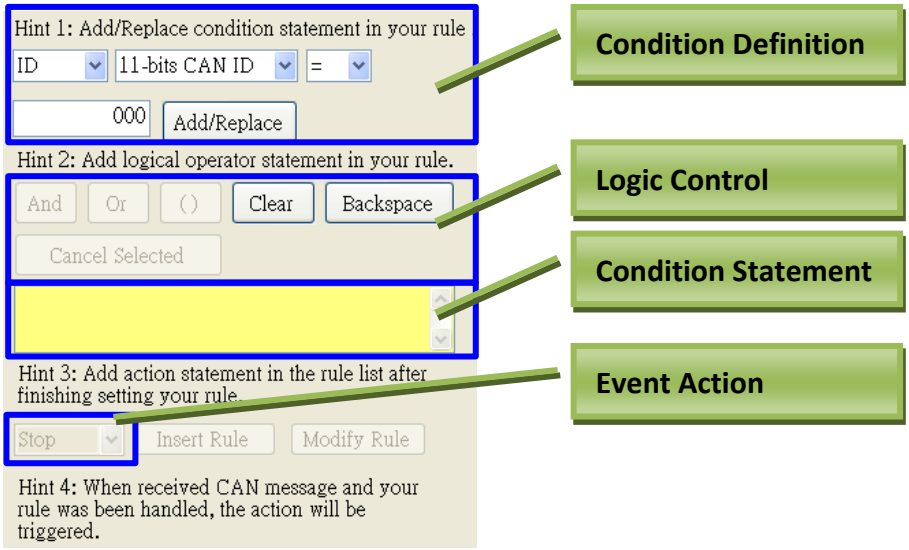
on the reception frame, the Advanced Receive interface will be popped up. The Advanced Reception interface provides the configuration the function of the trigger event. Users can define the conditions and corresponding actions when the VxCAN_Utility receives specific CAN messages.



The “Advanced Reception” interface is divided into two areas. One is conditional Area, and the other is List Area.

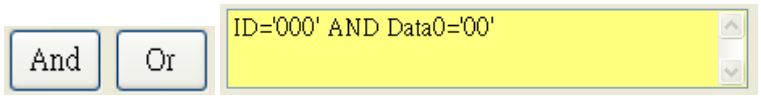
- **Condition Area:**

  Users can use this area to configure the conditions to trigger an event, and determine the corresponding action of the event. The details are described below.

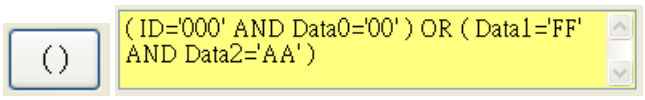


  - **Condition Definition:** An event is composed of several conditions. Here, users can define a condition by filling the value of these elements. For example, the screen displayed above indicates the condition is “11-bit CAN ID = 0x000”.
    - **Condition element:** Includes ID, Mode, Data, RTR, FDF, Data Length, and data.
    - **Comparison operator:** Includes “>”, “<”, “=”, “>=”, and “<=”.
    - **“Add/Replace” button:** After finishing the condition definitions, click this button to add the condition into the condition statement area. Users can also double click the condition list in the condition statement and click this button to replace the selected condition.

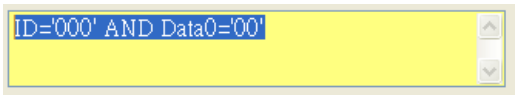
- **Logic Control:** This area is used to associate with two or more conditions.
  - **"And" and "Or" button:** It indicates the "And" and "Or" operator between two conditions.



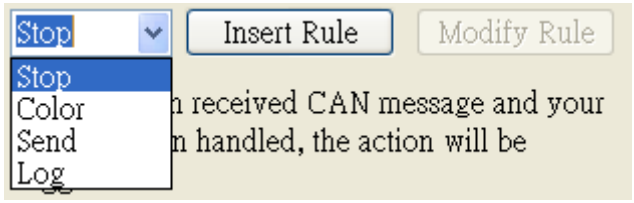
  - **"Parentheses" button:** In order to handle complex condition statements, users can use Parentheses for more than one condition. Select a fragment of the conditions in the condition statement first, and click Parentheses button to add the parentheses in the selected condition sections.



  - **"Clear" button:** Reset the condition statement to empty.
  - **"Backspace" button:** This function of this button is similar with the one in the keyboard. Double click the item or select some items in the condition statement first. Click the Backspace button will delete the selected items.



- **Condition Statement:** The result of the compound conditions will be shown in this field.
- **Event Action:** The VxCAN_Utility provides 4 kinds of action after the event is triggered. They are "stop receiving", "show color", "send CAN message", and "log received messages". Select one of them when finishing the configuration of the conditions.



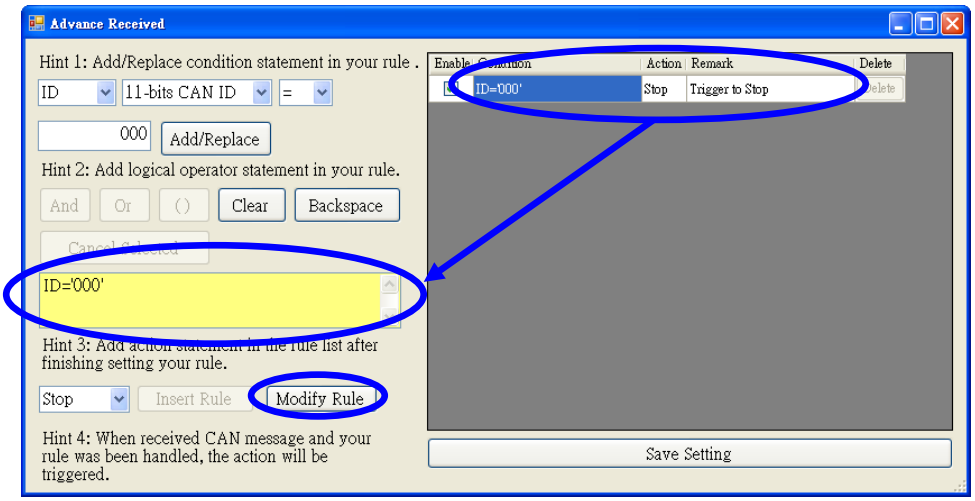
  - **"Insert Rule" button:** It is used to add rules into the list after finishing your condition.
  - **"Modify Rule" button:** It is used to modify the rules selected from the list after finishing your condition.

➢ **List area:**

This area lists the trigger events defined by users.

- **Enable:** Checking the checkbox will enable the predefined trigger event.
- **Condition:** Shows the condition statement defined by users.
- **Action:** Shows the corresponding action of the trigger event.

- **Delete:** Click this button to delete the selected condition statement.
- **How to Modify Condition Statement:** If the user wants to modify condition statement, first he needs to click the condition statement which he wants to modify from the list area. Afterwards, the condition statement will be shown on the condition statement area and the button “Modify Rule” will be enabled. After finishing the modification, please click the button “Modify Rule” to save setting.

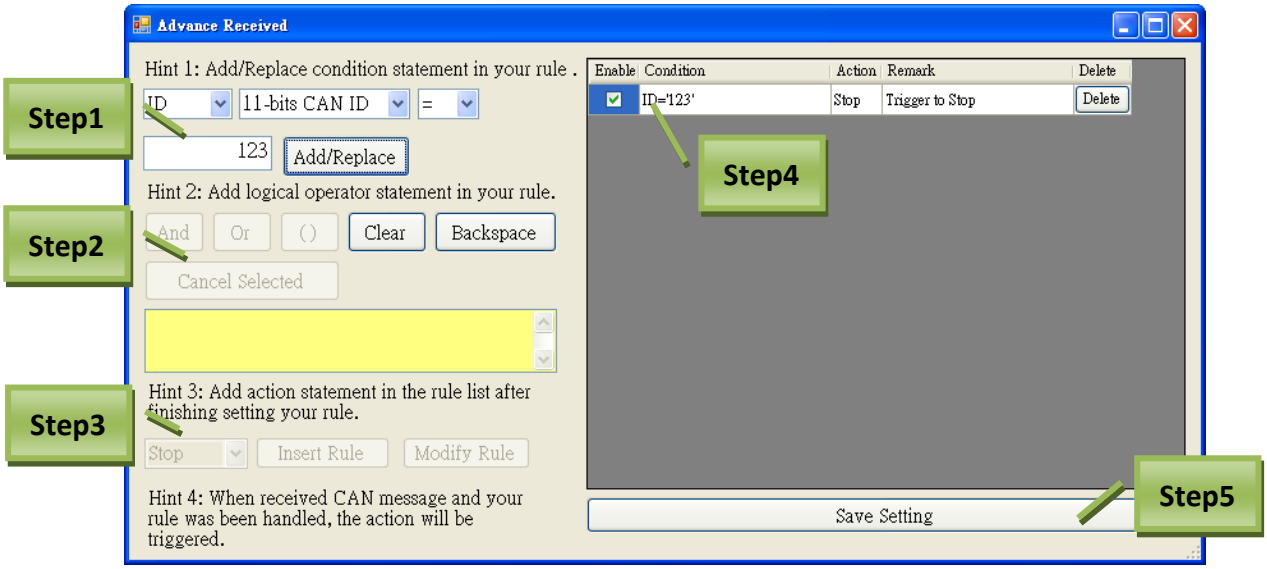


- **“Save Setting” button:** Saves all received trigger event setting.

There are examples demonstrated how to configure the conditions.

“Stop Receiving” event

Example 1: Define a condition that when receiving a CAN message which CAN ID is 0x123, the VxCAN_Utility will stop receiving the CAN messages.



Step 1: Set condition elements to “ID = 0x123” and click “Add/Replace” button to add a condition element.

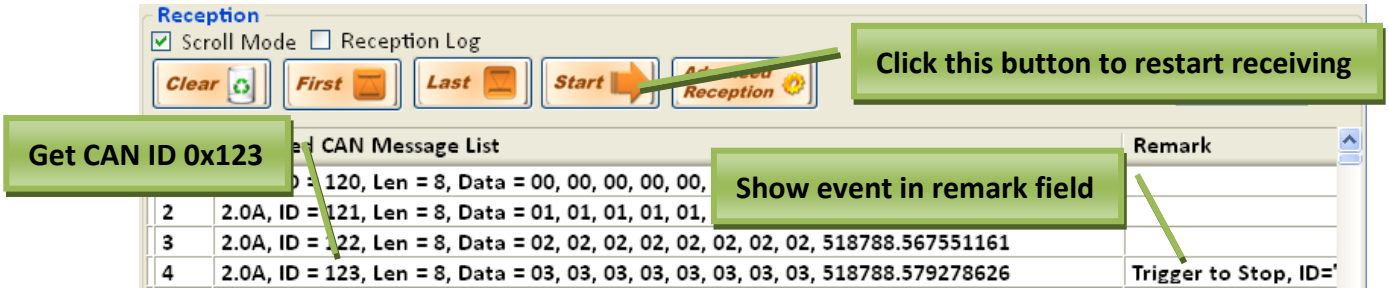
Step 2: Select proper logic to associate with two or more condition elements.

Step 3: After finishing the condition statement, select action “Stop” and click “Insert Rule” to add the condition into the list.

Step 4: Enable the condition statement by checking the checkbox.

Step 5: Click “Save Setting” button to save the configuration result.

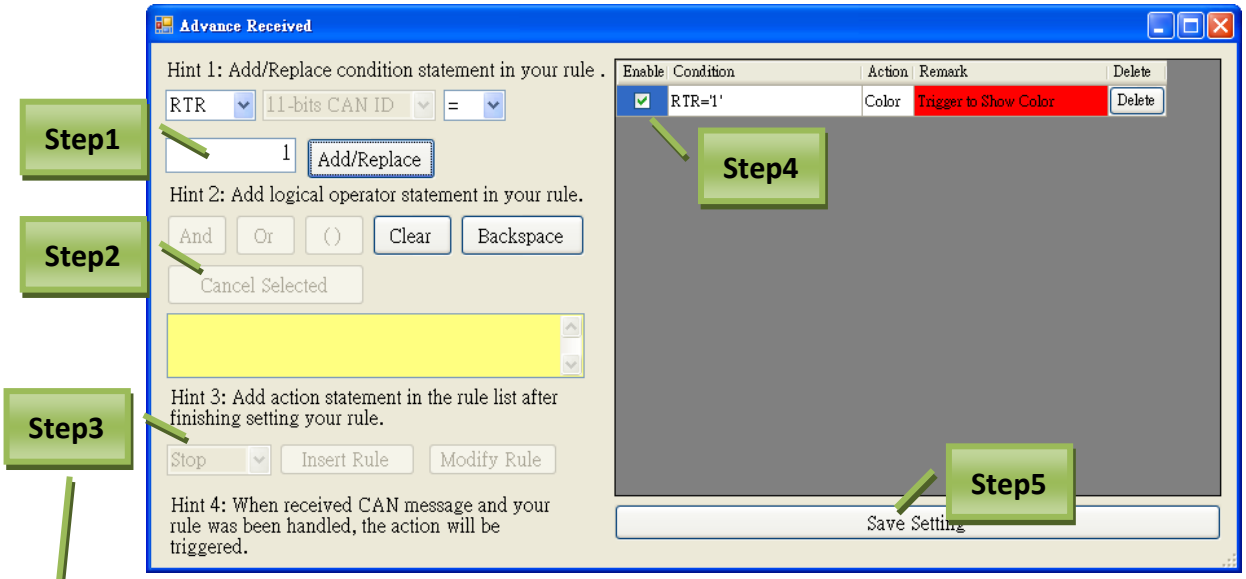
Afterwards, the VxCAN_Utility will stop receiving CAN message and show “Trigger to Stop” information in the remark field while the condition statement is meeting.



After clicking the “Start button”  , the Utility will start to receive CAN messages again.

“Show Color” event

Example 2: Define a condition that when receiving a CAN RTR message, the VxCAN_Utility will show this CAN message by predefined color.



Step 1: Set the condition elements to “RTR = 0x1” and click “Add/Replace” button to add a condition elements.

Step 2: Select proper logic to associate with two or more condition elements.

Step 3: After finishing the condition statement, select the action to “Color” and then a color panel will be popped up. Choose a color shown on the specific CAN messages, for example red color, and click “Insert Rule” to add the list.

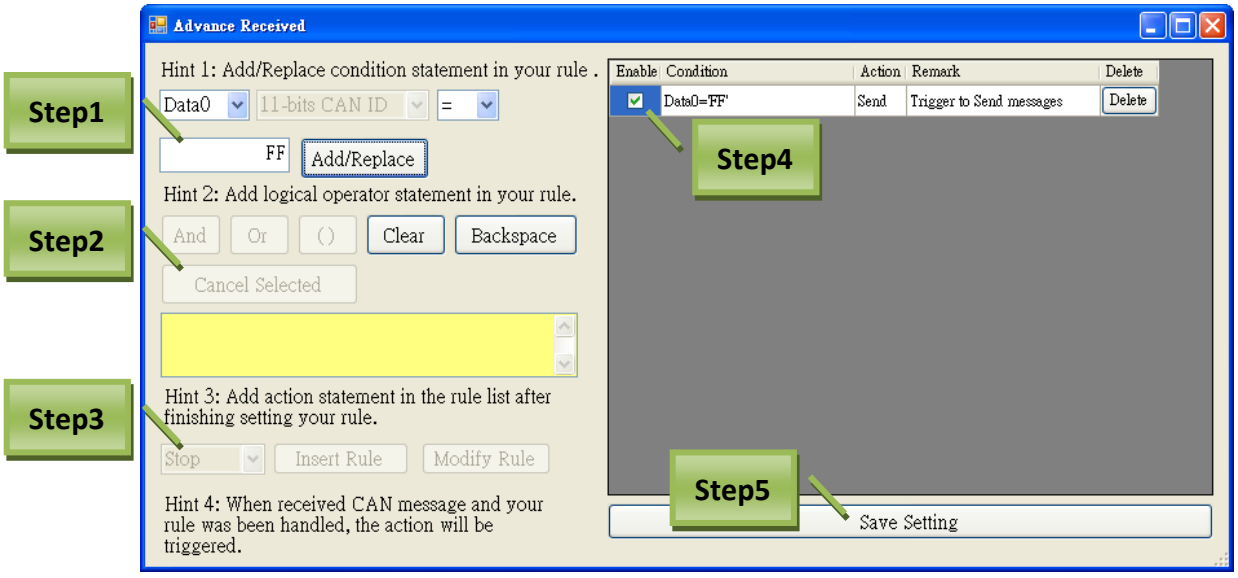
Step 4: Enable the condition statement by checking the checkbox.

Step 5: Click “Save Setting” button to save the configuration result.

When the VxCAN_Utility get the CAN RTR messages, the red color will be the background color on these messages, and hints will be shown in the remark field.

"Send CAN message" event

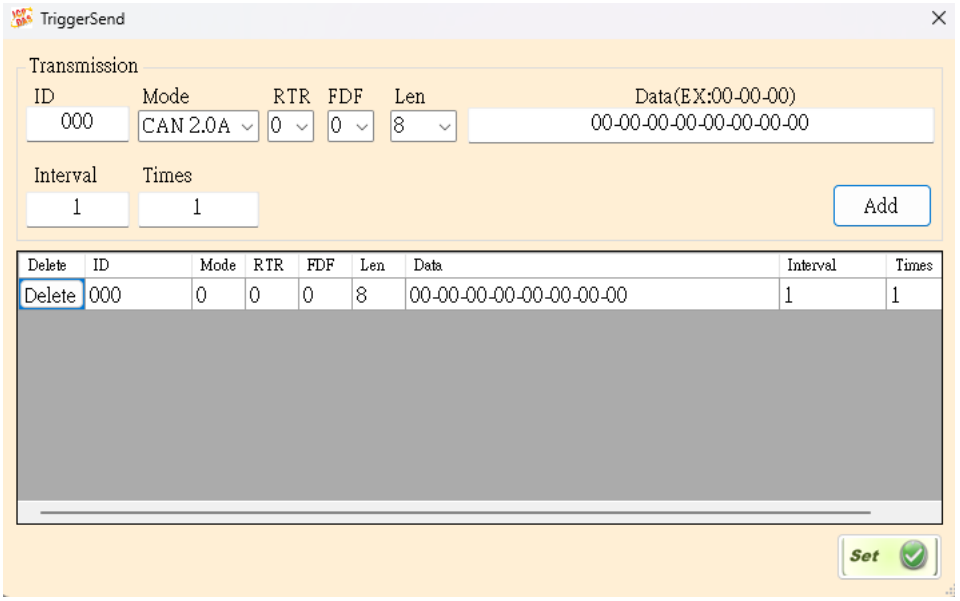
Example 3: Define a condition that when receiving a CAN message which Data 0 is 0xFF, the VxCAN_Utility will send a predefined CAN messages to the CAN network.



Step 1: Set the condition elements to "Data0 = 0xFF" and click "Add/Replace" button to add a condition elements.

Step 2: Select proper logic to associate with two or more condition elements.

Step 3: After finishing the condition statement, select the action "Send", and then a "TriggerSend" dialog will be popped up.

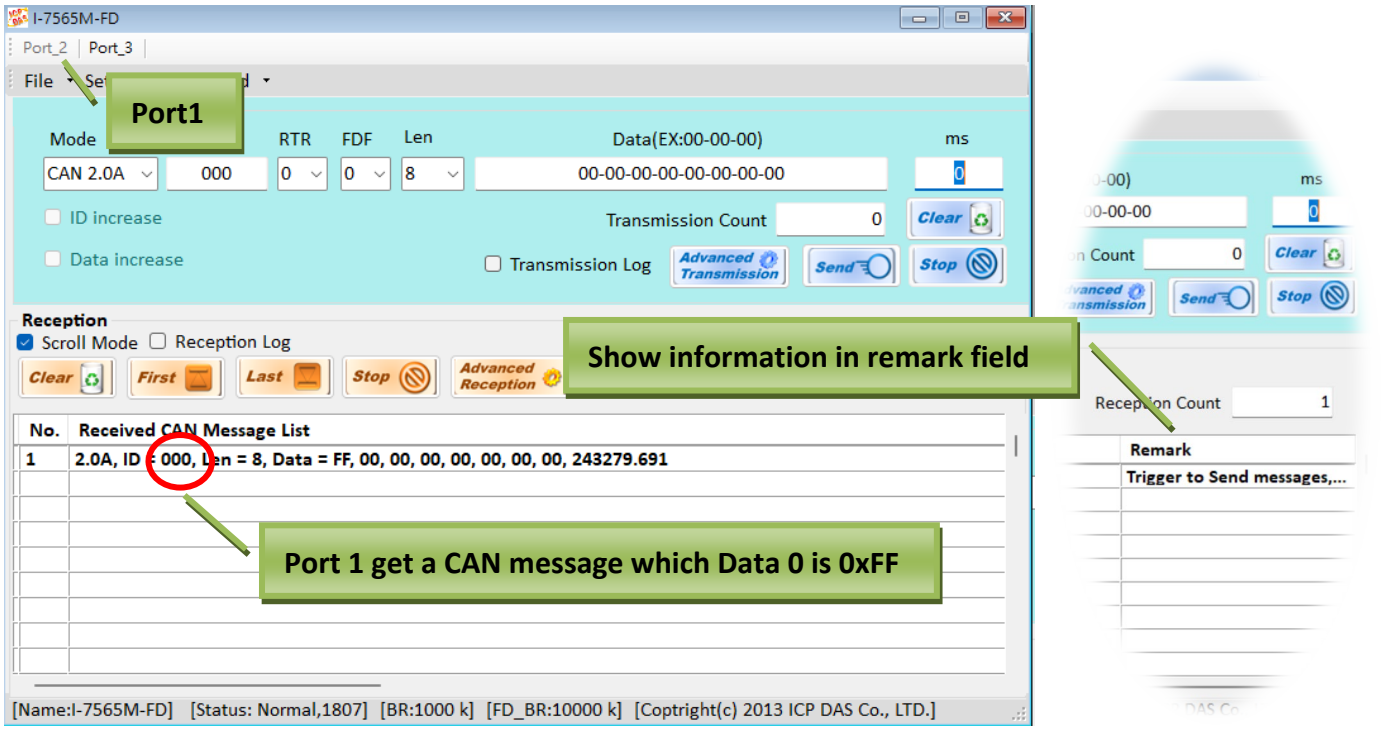


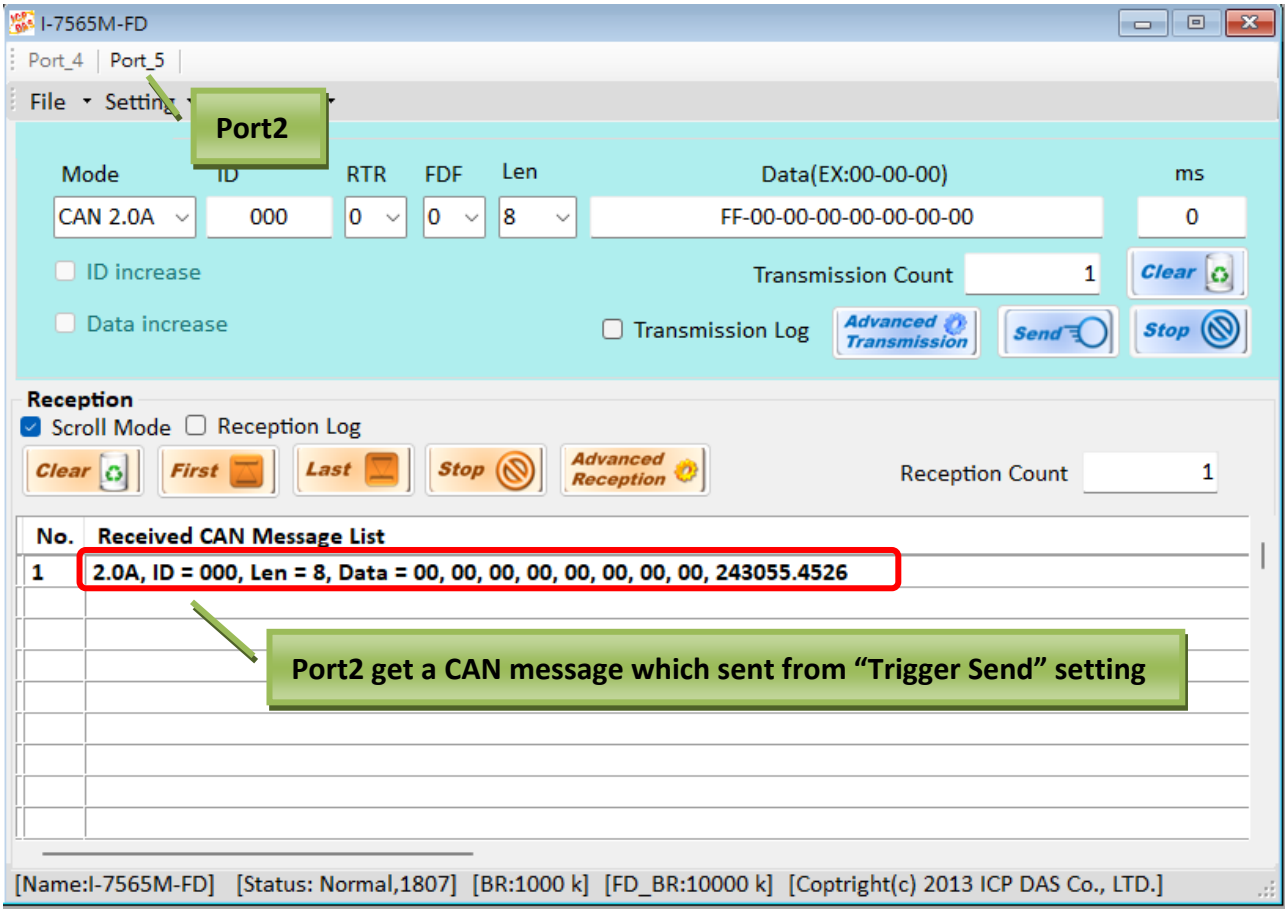
Users can configure the CAN messages sent to the CAN network while the condition statement is meeting. After finishing the configuration, click “Add” button to add a CAN message into the list. If users want to cancel the configuration of the specific CAN messages, use “Delete” button to do that. When everything is done, click “Set” button to save the setting. After finishing the “Trigger Send” configuration, click “Insert Rule” to add a condition statement into the list.

Step 4: Enable the condition statement by checking the checkbox.

Step 5: Click “Save Setting” button to save the configuration result.

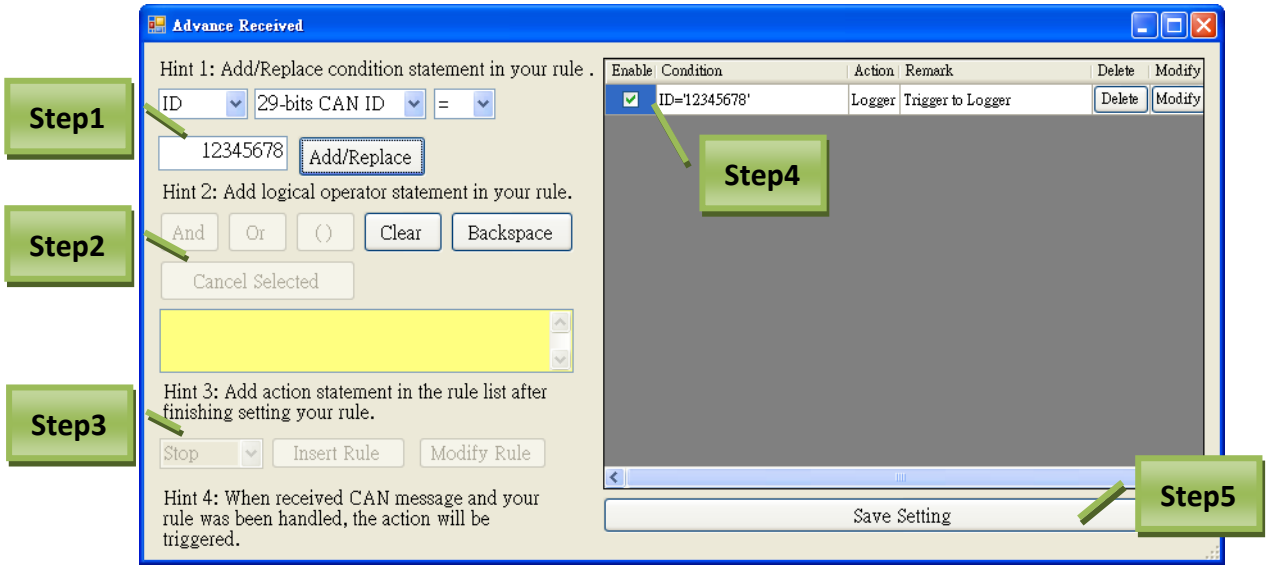
The VxCAN_Utility will send out the CAN messages which have been set before, and show the hint in the remark field when receiving a CAN message which Data 0 is 0xFF. Here, use two CAN ports to verify the results. All the results are shown as the screen below.





"Log received messages" event

Example 4: Define a condition that when receiving a CAN message which the CAN ID is 0x12345678, the VxCAN_Utility will start to log received CAN messages.



Step 1: Set the condition elements to "ID = 0x12345678" and click "Add/Replace" button to add a condition elements.

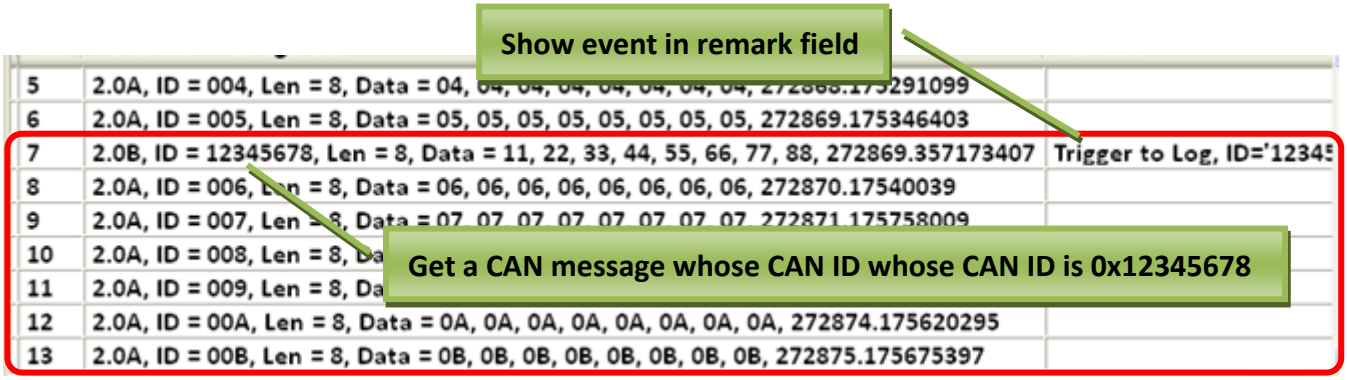
Step 2: Select proper logic to associate with two or more condition elements.

Step 3: After finishing the condition statement, select the action "Log" and click "Insert Rule" to add the condition statement to the list.

Step 4: Enable the condition statement by checking the checkbox.

Step 5: Click “Save Setting” button to save the configuration result.

The VxCAN_Utility will start to log received CAN messages and show the hint in remark field when receiving a CAN message whose CAN ID is 0x12345678.



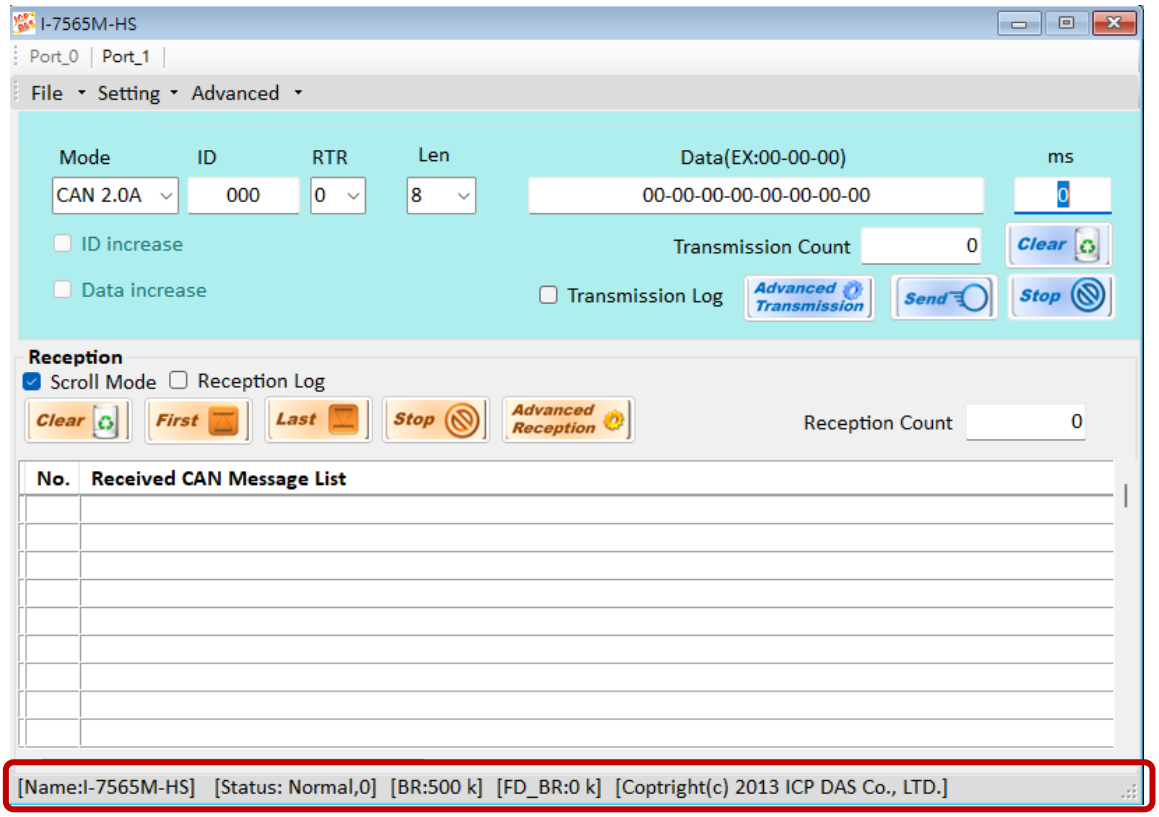
The default path of the log file is “C:\VxCAN_Logger\ RxLog.csv”.


RxLog.csv

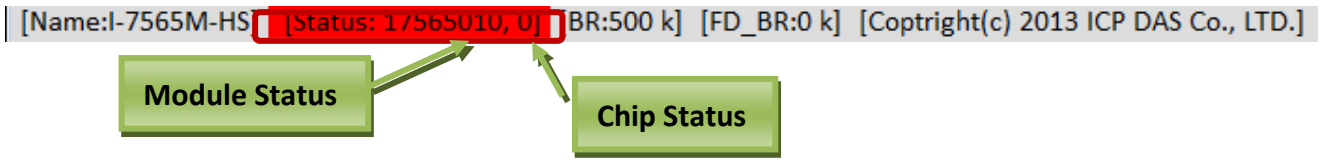
11/07 13:23:47:772 ---> (Rx) VxCANPort(0)	ID	Mode	RTR	Len	Data_00	Data_01	Data_02	Data_03	Data_04	Data_05	Data_06	Data_07
11/07 13:23:47:772 ---> (Rx) VxCANPort(0)	ID(0x12345678)	Mode(1)	RTR(0)	Len(8)	Data = 0x11	0x22	0x33	0x44	0x55	0x66	0x77	0x88
11/07 13:23:48.600 ---> (Rx) VxCANPort(0)	ID(0x6)	Mode(0)	RTR(0)	Len(8)	Data = 0x 6	0x 6	0x 6	0x 6	0x 6	0x 6	0x 6	0x 6
11/07 13:23:49.584 ---> (Rx) VxCANPort(0)	ID(0x7)	Mode(0)	RTR(0)	Len(8)	Data = 0x 7	0x 7	0x 7	0x 7	0x 7	0x 7	0x 7	0x 7
11/07 13:23:50.600 ---> (Rx) VxCANPort(0)	ID(0x8)	Mode(0)	RTR(0)	Len(8)	Data = 0x 8	0x 8	0x 8	0x 8	0x 8	0x 8	0x 8	0x 8
11/07 13:23:51.600 ---> (Rx) VxCANPort(0)	ID(0x9)	Mode(0)	RTR(0)	Len(8)	Data = 0x 9	0x 9	0x 9	0x 9	0x 9	0x 9	0x 9	0x 9
11/07 13:23:52.584 ---> (Rx) VxCANPort(0)	ID(0xA)	Mode(0)	RTR(0)	Len(8)	Data = 0x A	0x A	0x A	0x A	0x A	0x A	0x A	0x A
11/07 13:23:53.584 ---> (Rx) VxCANPort(0)	ID(0xB)	Mode(0)	RTR(0)	Len(8)	Data = 0x B	0x B	0x B	0x B	0x B	0x B	0x B	0x B

3.2.2.5. Status

The CAN interface status of the selected CAN device will be shown on the bottom of the screen. They are name, status, and baud rate or FD baud rate.



- **Name:** The module name of the selected CAN module.
- **Status:** There are Module Status and Chip Status. When the VxCAN_Utility detects some errors on the CAN module, the status area shows the error code with red background color. The meanings of the error codes are defined in the appendix I~IV.



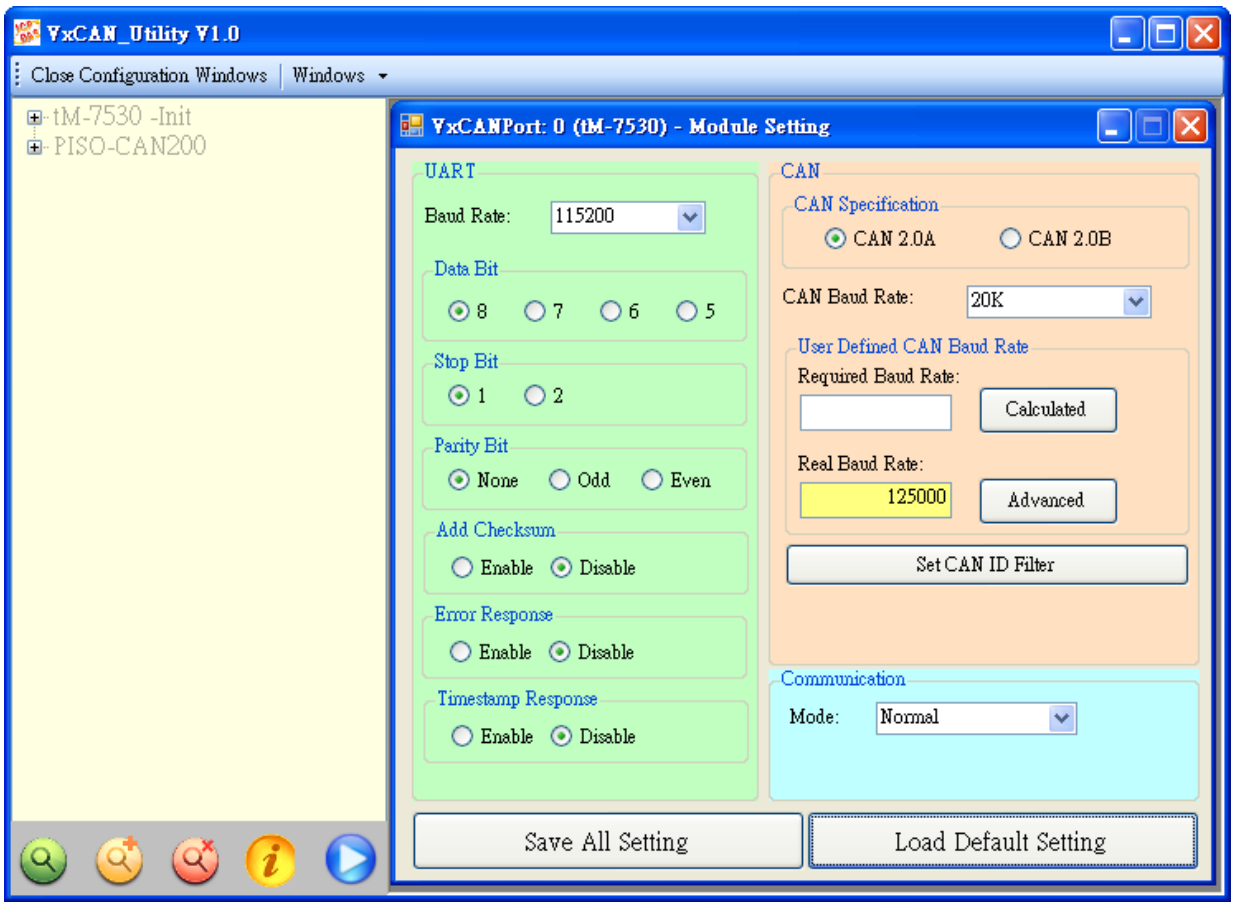
- **BaudRate:** The implemented baud rate of the port of the CAN module.
- **FD BaudRate:** Baud rate of the CAN FD module port. If it is a CAN-only device, this field is displayed as 0.

3.3. Module configuration

Some CAN converters, such as I-7530 series modules, must be configured by the utility tool first before using then on the CAN network. The VxCAN_Utility has integrated the utility of the CAN converters, and provides the functions to configure the CAN converters.

3.3.1. Configure I-7530 series

The configuration function supports the module I-7530(T), I-7530-FT, I-7530A, I-7565, I-7530A-MR, and tM-7530. The configuration of the I-7530 series modules is divided to three parts as the following screen. They are UART communication, CAN communication, and communication mode.



- **UART Configuration:**
  - **Baud Rate:** Set the UART baud rate. Different CAN converters have different max. UART baud rates. Users can select the proper UART baud for the CAN converters.
  - **Data Bit:** Set the UART data bit. The CAN converters support 4 kinds of data bit configuration.

    Note: VxCAN only supports 8 Data Bits.
  - **Stop Bit:** Set the UART stop bit. The CAN converters support 2 kinds of stop bit configuration.

    Note: VxCAN only supports 1 Stop Bits.
  - **Parity Bit:** Set the UART parity bit. The CAN converters support 3 kinds of parity bit configuration.

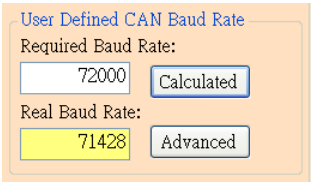
    Note: VxCAN only supports "None" for Parity Bit.
  - **Add Checksum:** Set the CAN converters to enable or disable the checksum mechanism in the UART communication. If enable the checksum function, all UART strings from / to the tM-7530 must append two bytes of the checksum information.

    *** Note: VxCAN does not support enabling Checksum.**
  - **Error Response:** Set the CAN converters to enable or disable the error response mechanism in the UART communication. If enable the error response, the error code will be replied when users use the wrong UART command string to communicate with the CAN converters.

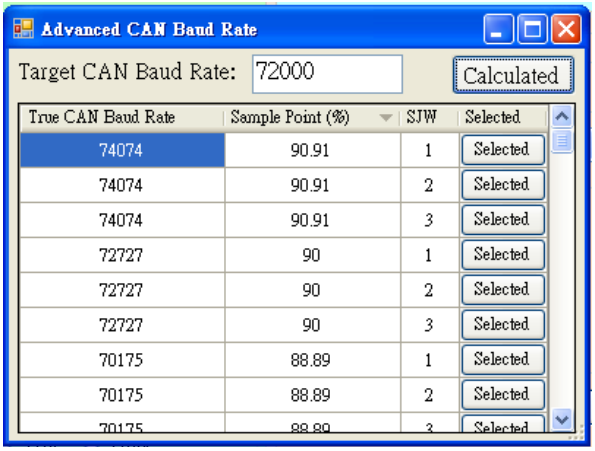
- **Timestamp Response:** Set the CAN converters to enable or disable the timestamp response mechanism. If enable the timestamp response, the timestamp of each CAN message will be appended in the UART string when the CAN message are output from the UART interface of the CAN converter. This function may be invisible because it is not supported by the selected CAN converters.

■ **CAN Configuration:**

- **CAN specification:** Decide which CAN specification, CAN 2.0A (11-bits CAN ID) or CAN 2.0B (29-bits CAN ID), will be implemented.
- **CAN baud rate:** Set the CAN baud rate. The CAN converters support several kinds of standard baud, such as 10K, 20K, 50K, 125K, 250K, 500K, 800K, 1000K, and 83.3K bps. If these baud rates don't fit the users' application, use user-defined CAN baud rate to configure the special CAN bauds.
- **User-defined CAN baud rate:** If the user-defined CAN baud rate is used, users need to fill the value of the baud rate in the "Required Baud Rate" field and click the "Calculated" button to calculate the real value. If the real baud rate is not the same as the value in the "Required Baud Rate" filed, it means that the required baud rate can't be reached because of the hardware limitation. The closest baud value will be implemented for instead.

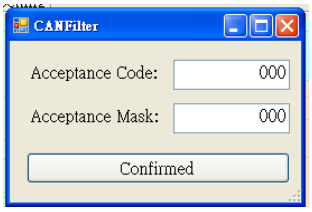


If users want to get more information of the real baud rate, click the "Advanced" button. Users can select the proper parameters of the real baud rate for the applications.

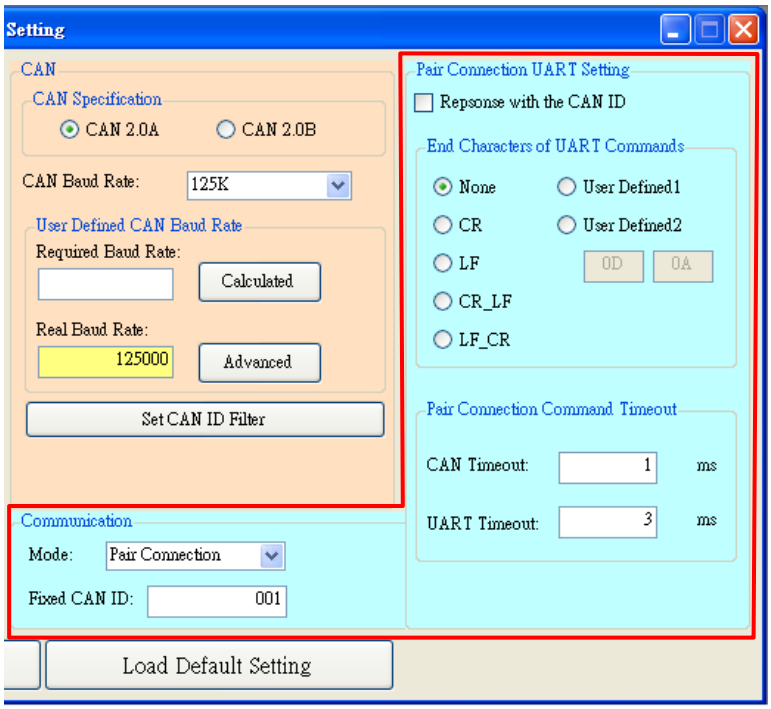


The field "Sample Point" means the percent of the sampling position of one bit data of the CAN message. The field "SJW" is the short for synchronization jump width. It is used to solve the problem of the phase shift between clock oscillators of different CAN devices. Generally, the sample point is set to close 87.5%, and the SJW is set to 1.

- **Set CAN ID Filter:** Click the button to pop up the configuration dialog of the CAN ID filter. Users can use the acceptance code and acceptance mask to determine the accepted CAN messages by CAN message IDs.

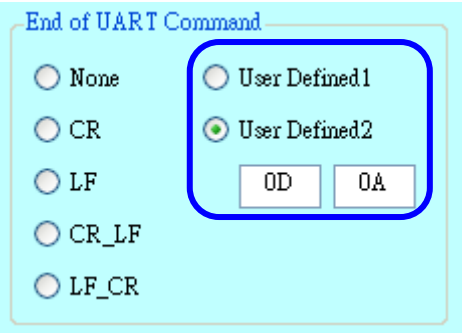


- **Communication Configuration:** It is used to set the communication mode of the CAN converters. In normal mode, the UART interface of the CAN converters only accept the command strings defined in the CAN converter's manual section 4. Any UART messages which don't follow the command strings will be regard as wrong messages. When the CAN converters transfer the CAN messages to UART interface, the CAN messages are presented by the command strings. In pair connection mode, the CAN converters will transfer any UART message to the data field of the CAN messages whose message ID are fixed and predefined by the utility tool. It is useful for transparent applications or pair connection applications. The tM-7530 has an additional communication mode, "Listen Only" mode. In this mode, the tM-7530 can only receive the CAN messages and can't send any CAN signal (include CAN Error Frame and the change of the ACK field) to the CAN network.
- **Pair Connection Mode:** When uses choose the Pair Connection in the Communication Mode, the pair connection configuration field will be presented.



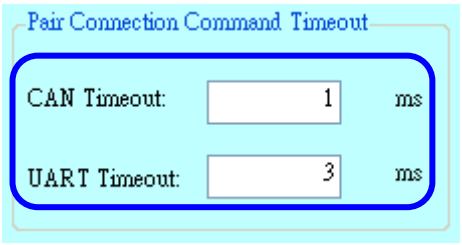
- **Fixed CAN ID:** Set the fixed CAN ID for the transmitted UART data to the CAN network. Each CAN converter must have different configuration of the fixed CAN ID. If there are more than one CAN converter in the pair connection mode, the same fixed CAN ID will make the CAN ID conflict error while these CAN converter are transferring the UART data to the CAN message at the same time.

- **Response with CAN ID:** Users can decide if the CAN ID needs to be transmitted with the UART data to the UART interface. In the pair connection application (see the picture of the section 1), the CAN ID always doesn't need to be send to the UART interface.
- **End Characters of UART Commands:** This function is used to set the end characters of the UART command received by the CAN converter. The UART data transferred from the CAN messages will not append the specific end characters. If the CAN converter gets the specific end characters from the UART interface, it is regarded as the end of the UART command, and the CAN converter will start to transfer the UART command to the CAN messages immediately.
  - **None:** None end character is used. When the pair connection command timeout is reached or the UART buffer is full, the CAN converter will start to transfer the UART data to the CAN network.
  - **CR:** Set the end character of the UART command to CR. The hexadecimal value of the ASCII code is '0x0D'.
  - **LF:** Set the end character of the UART command to CR. The hexadecimal value of the ASCII code is '0x0A'.
  - **CR_LF:** Set the end characters of the UART command to two characters, CR and LF. The hexadecimal value of the ASCII code is '0x0D' and '0x0A'.
  - **LF_CR:** Set the end characters of the UART command to two characters, LF and CR. The hexadecimal value of the ASCII code is '0x0A' and '0x0D'.
  - **User-defined:** This function allows users to define the special end characters, and is only supported by the tM-7530. The UserDefined1 or UserDefined2 are used to configure one or two end characters. Take followings figure for example, select the item "User Defined2" and set the hexadecimal value of the end characters to be the 0x0D and 0x0A. When the CAN converter get the UART messages with end characters '0x0D' and '0x0A', it is regarded as the ending of the UART message. The CAN converter will start to transfer the UART message to the CAN Bus.

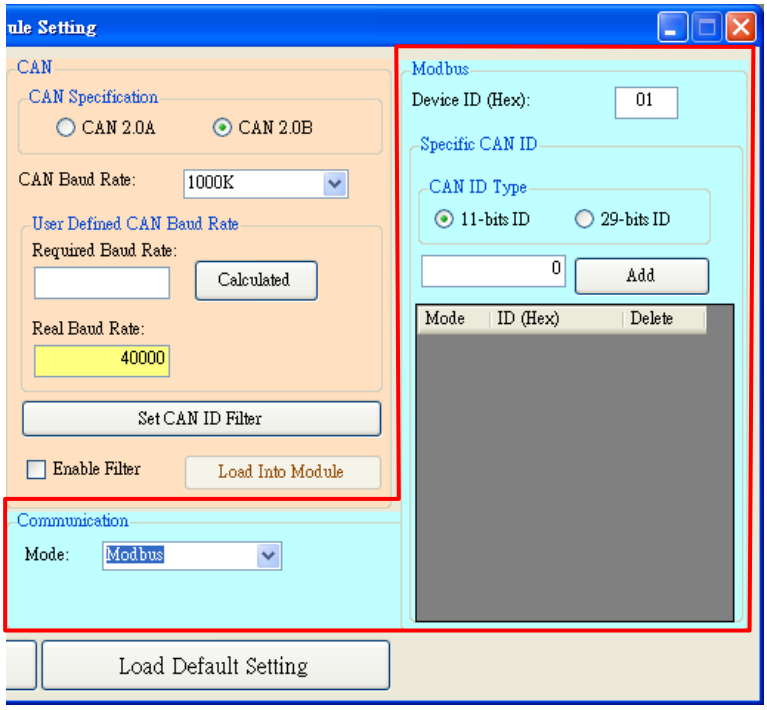


- **Pair Connection Command Timeout:** Only the tM-7530 and I-7530A-MR supports this function. The CAN timeout is used to decide the transformation timing of the CAN messages to the UART messages. After the CAN converter receives the CAN message, it will not transfer the CAN message to UART message until the time of the CAN timeout passes. If users would like to transfer the CAN message to UART messages immediately, set this value to 0. The function of the UART timeout is similar with the CAN timeout. It is decide the

transformation timing of the UART messages to CAN messages. Only the configuration of the end characters of the UART message is none, the UART timeout is useful. After receiving one character from the UART interface, the CAN converter will not transfer the data until the time of the UART timeout passes.



- **Modbus Mode:** The I-7530A-MR provides the Modbus communication functions. For this module, the Modbus item is present in the Mode list. When selecting the Modbus communication mode, the extension configuration page is displayed as below.



  - **Device ID(Hex):** The I-7530A-MR plays a role of the Modbus slave. Therefore, set the Modbus slave ID of the I-7530A-MR.
  - **Specific CAN ID:** The I-7530A-MR allows users to arrange a Modbus registers for the CAN messages with specific CAN ID. Set the specific CAN ID here for the purpose. Users can select the CAN specification, CAN 2.0A or CAN 2.0B, and set the CAN ID value into the ID field. Afterwards. Click “Add” button to add the setting into the list. When the I-7530A-MR gets these CAN messages with specific CAN ID, the CAN messages will be put in the specific Modbus registers. Users can read these Modbus registers to get the CAN messages directly, not from ring buffer. It is useful if these CAN messages are very critical. About the details, please refer to the user’s manual of the I-7530A-MR.

- **Save and default value button:**

After finishing the configuration, users can click the button “Save All Setting” to save the configuration in to the EEPROM of the CAN converter. If users would like to recover the parameters to the factory default, click “Load Default Setting” button to recover all of the configuration parameters to be default value. Afterwards, users can use the button “Save All Settings” to save the default parameters in to the EEPROM of the CAN converter.

The default values of the parameters of the CAN converter are shown below.

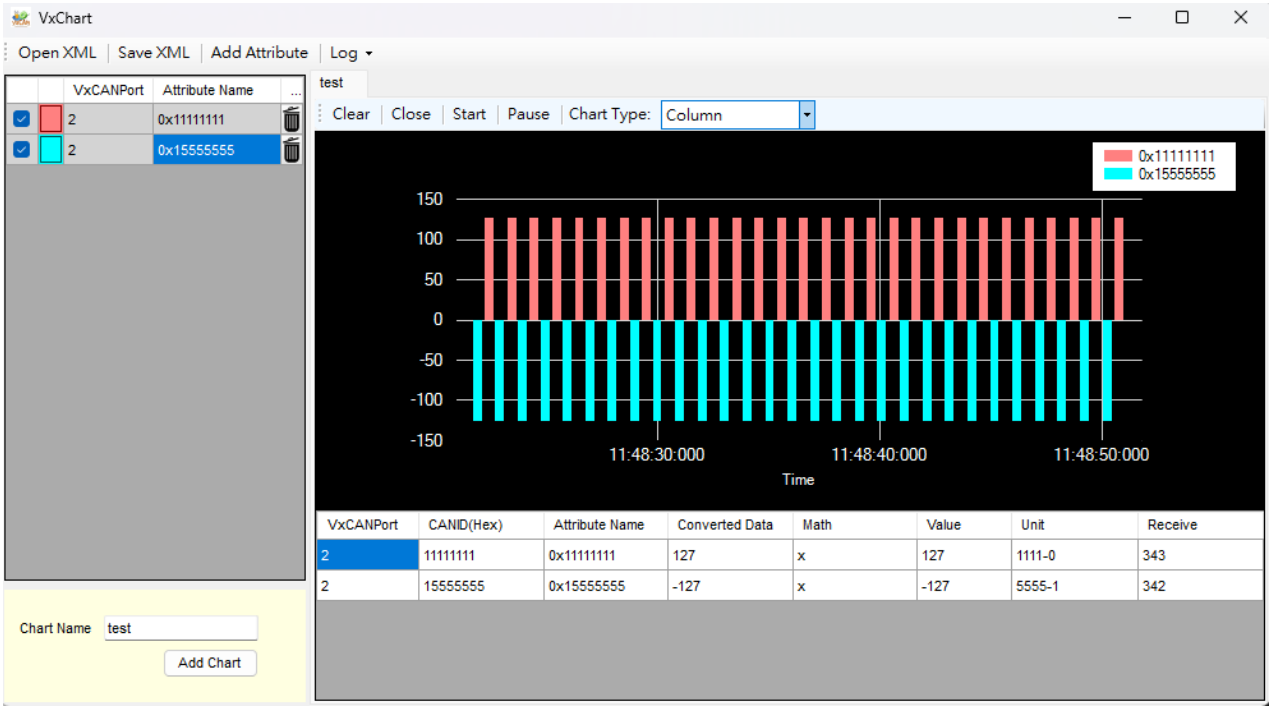
RS-232: RS-232 Baud rate = 115200/921600(for I-7565)
Data Bit = 8
Stop Bit = 1
Parity = None
Add Checksum = No
Error Response = No
TimeStamp Response = No

CAN: CAN Specification = 2.0A
CAN Bus Baud rate = 125K
Acceptance Code = 000
Acceptance Mask = 000

Communication:
Mode: Normal

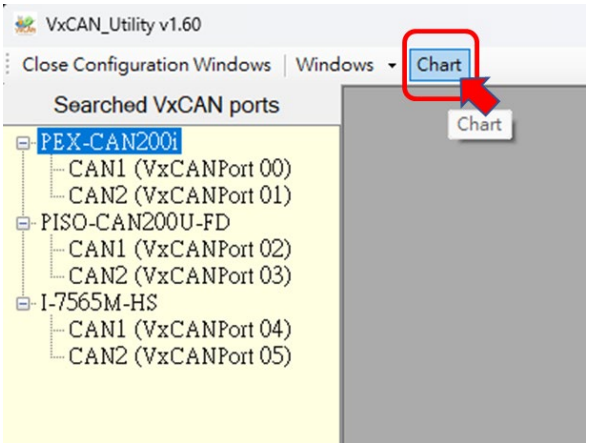
3.4. Chart

In the VxChart page, you can convert received CAN/FD messages into physical quantities and display them as charts. By importing an XML file or customizing attributes, users can provide the parameters required for data analysis. VxChart can help you analyze the received CAN messages, allowing you to more intuitively observe the meaning of the received CAN messages.



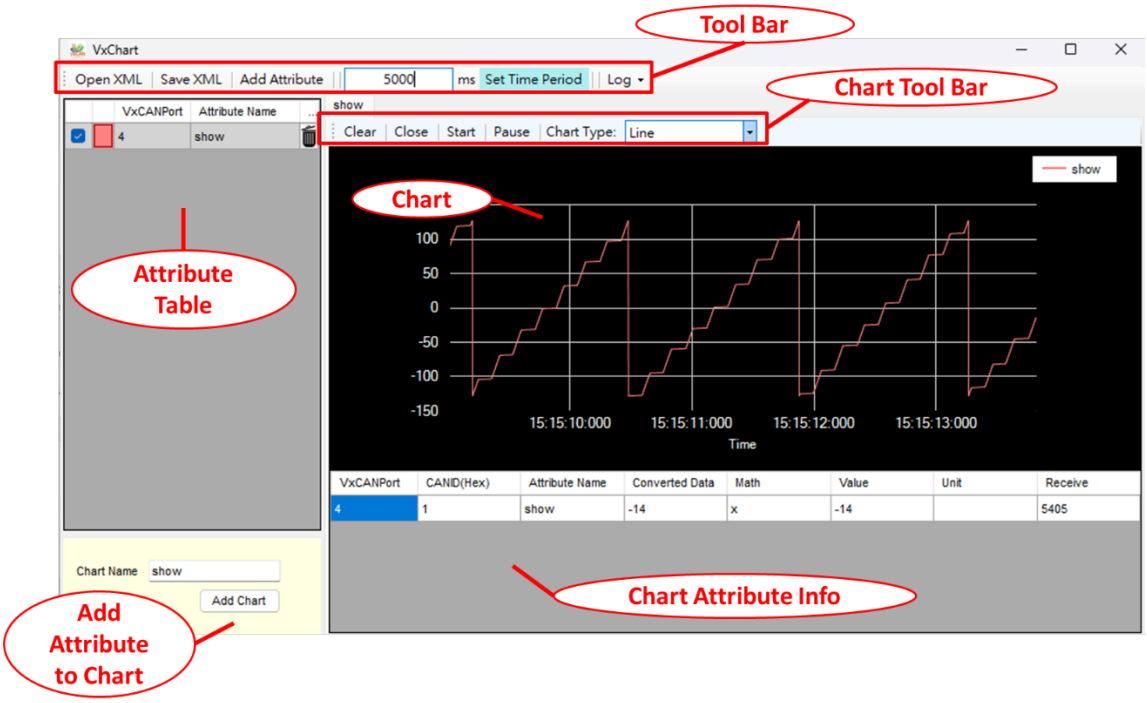
3.4.1. Open VxChart

Click the "Chart" option at the top of the VxCAN Utility window.



3.4.2. VxChart Interface Overview

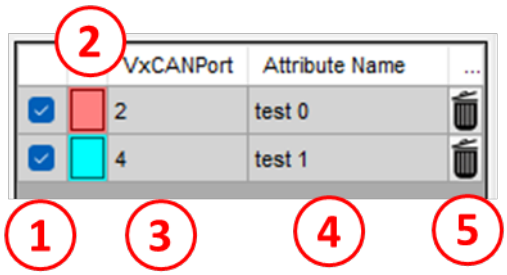
The VxChart page is divided into six main sections, as shown in the figure below:



1. Tool Bar:

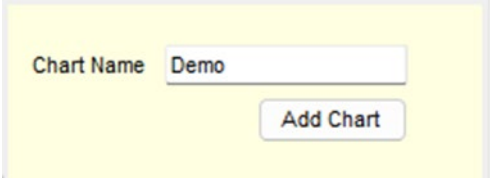
- Open XML: Open an XML file that conforms to the specified format from your computer and import all attributes within.
- Save XML: Save all attributes in the Attribute Table as an XML file. You can directly import the recorded attribute data the next time you use it by opening the XML file. (Note: Please avoid entering special characters as this may cause issues when re-importing. For a list of special characters, please refer to section 3.4.6.)
- Add Attribute: Opens a new window where you can directly edit and add new attributes. If added successfully, the new attribute will be displayed in the Attribute Table.
- Set Time Period: Set the duration for retaining data within the chart.
- Log: Provides a function to save all analyzed data as a CSV file. You can set the file save location and enable/disable the Log function here. (Note: The log file will be named with the system time, and a new file will be created and saved automatically when the file size exceeds 300MB.)

2. Attribute Table: Displays information of all attributes. When the corresponding VxCANPort module is enabled or not being used by a chart, the attribute is selectable (white), otherwise it is unselectable (gray).

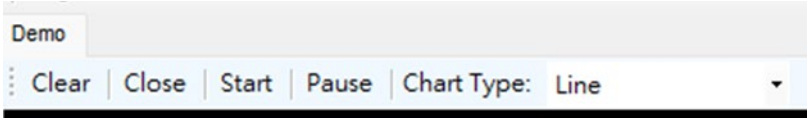


(1) Checkbox: Select the attributes you want to analyze before adding a chart.
(2) Color Block: Adjust the color representing the attribute in the chart.
(3) VxCANPort: Indicates that the attribute is analyzing data received from the specified VxCANPort.
(4) Attribute Name: Unique and cannot be duplicated. It represents the meaning of the attribute.
(5) Delete: Delete the attribute. (Note: If the attribute is being used by a chart, please close the chart before deleting.)
(6) Double-click the attribute field: Open the edit window for the selected attribute. You can adjust the selected attribute here.

3. Add Attribute to Chart: Add the selected attributes to the chart. New charts will be displayed in separate tabs on the right side.

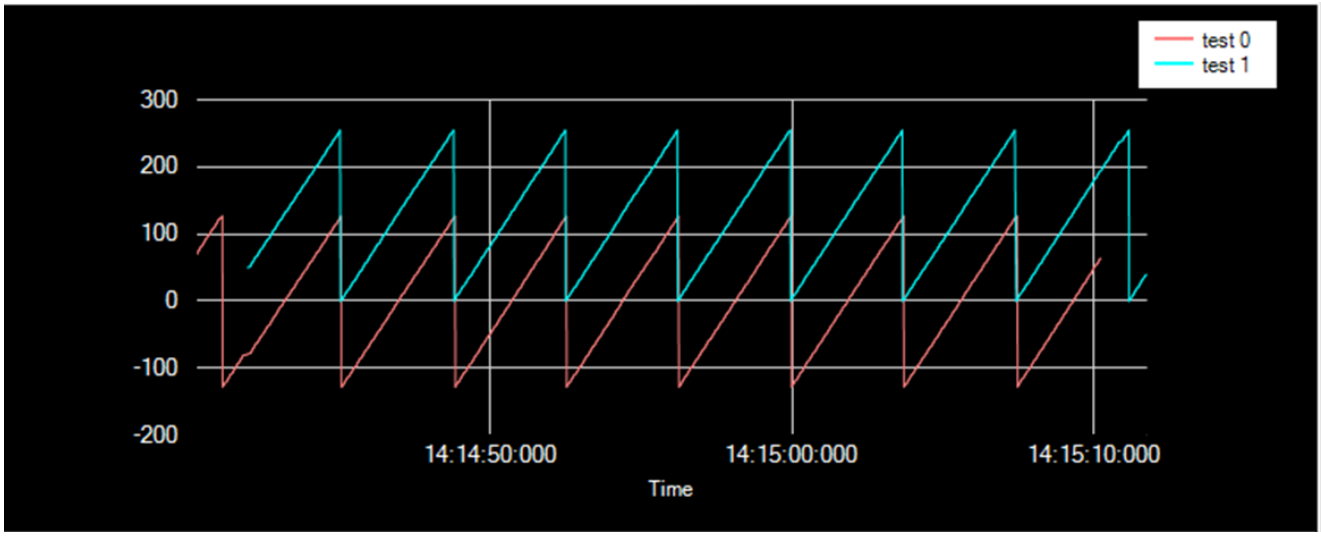


4. Chart Tool Bar: Chart controls on the current page.



- Clear: Clear all data in the current chart tab.
- Close: Close the current chart tab and stop using all attributes within.
- Start: Resume CAN data analysis for attributes in the current chart tab.
- Pause: Pause CAN data analysis for attributes in the current chart tab.
- Chart Type: Change the visualization type of data in the current chart tab.

5. Chart: Displays the physical values after converting CAN data. (Note: The chart only retains 30 seconds of data.)



6. Chart Attribute Info: Displays detailed information about the attributes being analyzed on the current chart page, including the amount of data received, analyzed data, and status.

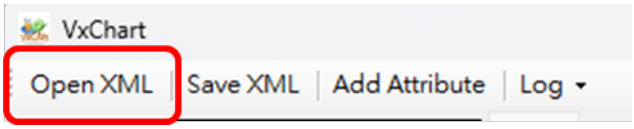
VxCANPort	CANID(Hex)	Attribute Name	Converted Data	Math	Value	Unit	Receive
2	0	test 0	64	x	64	test 0	3905
4	11	test 1	40	x	40	test 1	4137

- If the VxCANPort for the attribute is closed, the "Receive" column will display information indicating that the VxCANPort is closed. To resume analysis for this attribute, please use the "Add Chart" function again.

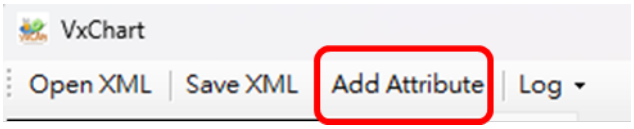
3.4.3. Add Attribute

There are two ways to add attributes: loading an external XML file and manually adding attribute data.

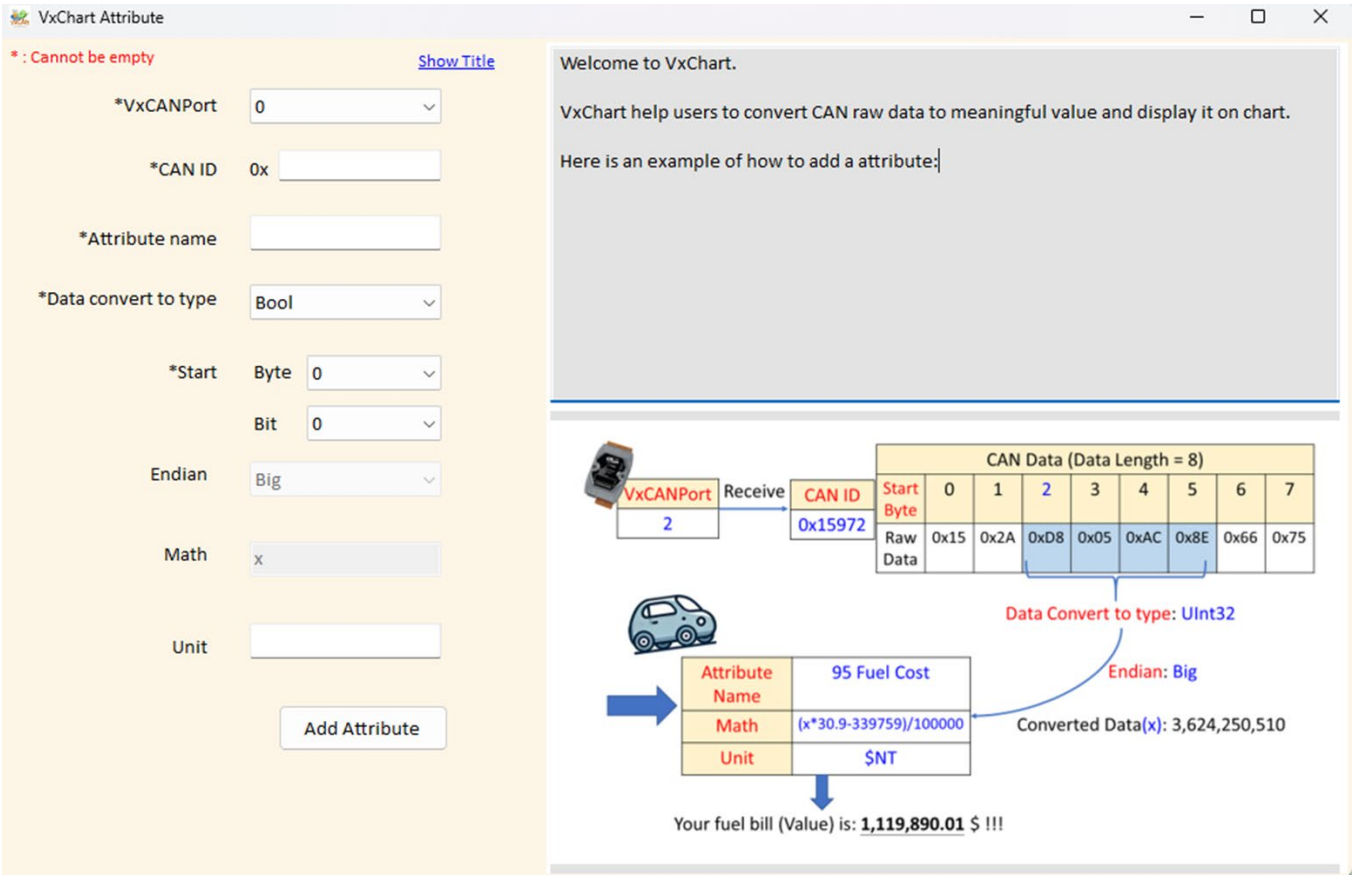
1. Loading an external XML file: Please select "Open XML" from the top toolbar and open the XML file. The system will check if the file format is correct during the import process. If the format is incorrect, the import will fail and an error message will be displayed. Once the import is successful, the imported attributes will be shown in the Attribute Table below. Detailed XML format will be introduced in section 3.4.6.



2. Manually adding attribute data: Please select "Add Attribute" from the top toolbar, which will open a new window. If you are unfamiliar with XML format, it is recommended to use the manual addition function. After adding attributes, you can save the attributes in the Attribute Table as an XML file. Subsequently, you can directly edit the XML content externally or import attributes directly from the XML file.



In this section, we will explain how to manually add attributes. After selecting "Add Attribute", a window for editing attribute data will open. As shown in the figure below, the left side of the window is for entering the attribute information to be analyzed, and the right side provides explanations for the attribute fields.

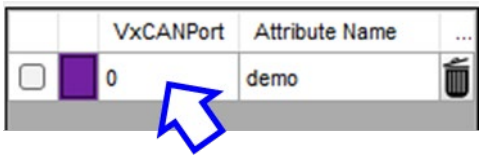


Field Descriptions:

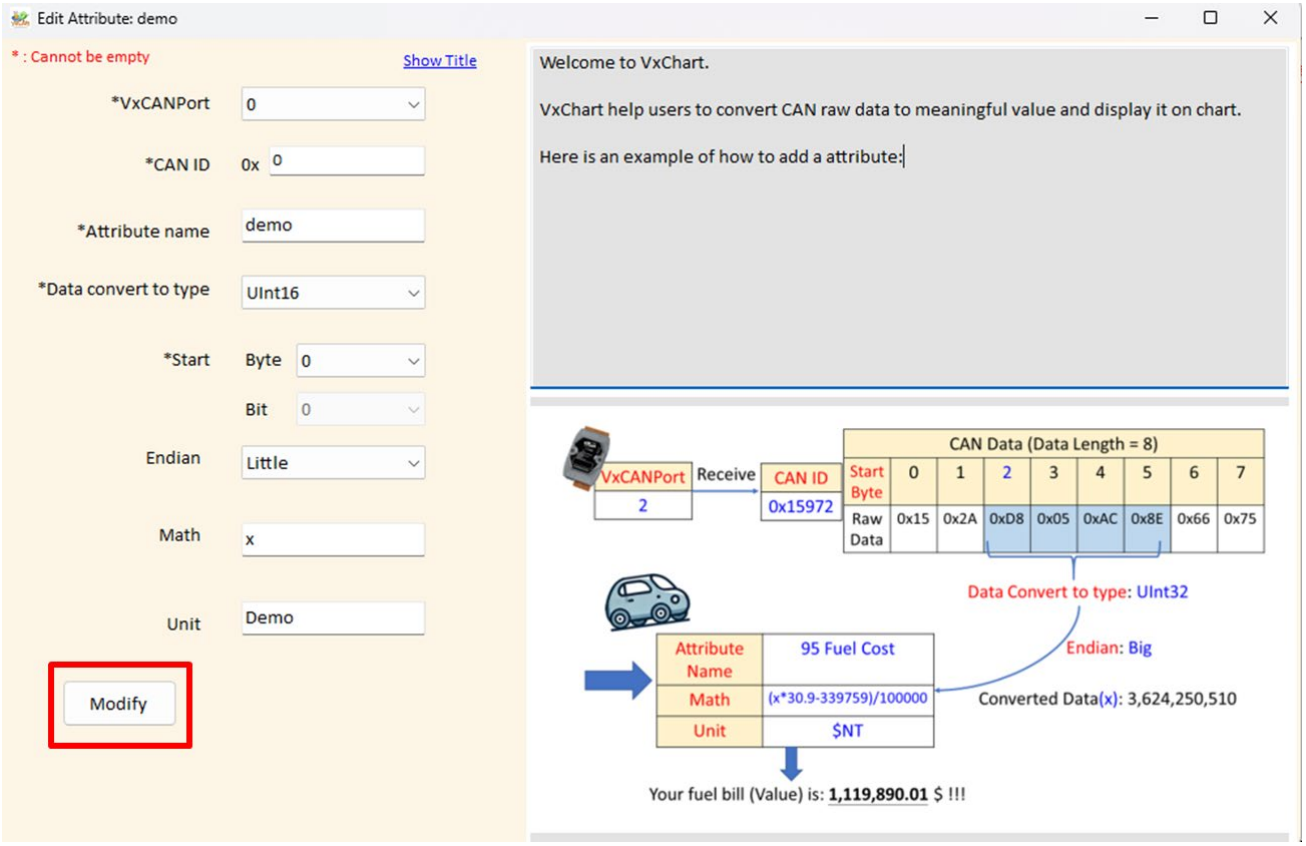
1. VxCANPort: Please select the corresponding VxCANPort number for the CAN port of the module you are using.
2. CAN ID: Please enter the CAN ID you want to analyze.
3. Attribute name: Please enter the name for the attribute you want to analyze. Note that this field cannot contain names that have already been used.
4. Data convert to type: Please select the data type to which you want to convert the CAN data.
5. Start Byte/Bit: Please select the starting byte of the CAN data. Start Bit can only be selected when the data type is bool.
6. Endian: When selecting a type other than bool, SignedByte, or UnsignedByte, you can choose between Big Endian and Little Endian. The default is Big Endian.
7. Math: The operation to be performed after converting the CAN data to a numerical value. Please use "x" in the mathematical formula to represent the variable that will be replaced with the CAN data value. Supported operators include: +(addition), -(subtraction), *(multiplication), /(division), ^(power), $\sqrt{}$ (square root). Math defaults to "x", which means the value after converting the CAN data to the specified data type.
8. Unit: Please enter the unit of the data after conversion to a physical quantity.
9. Add Attribute: After entering all the data, click "Add Attribute". After confirming that the data is correct, the newly added attribute will be displayed in the Attribute Table.

3.4.4. Edit Attribute

In the Attribute Table, double-click on the attribute you want to modify to open the editing window.Note: Attributes currently in use cannot be edited.

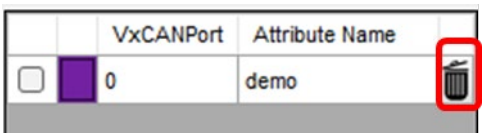


The attribute editing window that opens is similar to the Add Attribute operation (section 3.4.3). The fields in the window will display the settings of the selected attribute. After making adjustments, select "Modify" to complete the attribute editing.



3.4.5. Delete Attribute

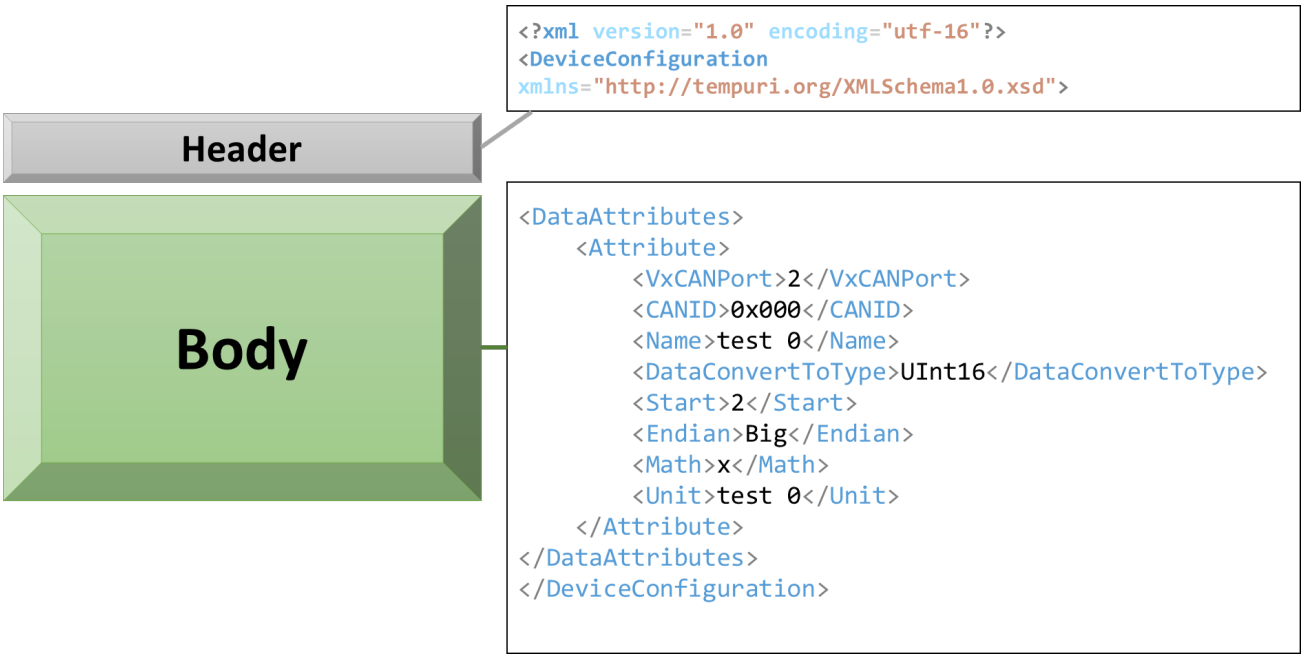
To delete an attribute, locate the trash can icon on the far right of the desired attribute's row in the Attribute Table. Clicking on this icon will remove the attribute data. Note: You cannot delete an attribute that is currently being used by a chart.



3.4.6. XML

In section 3.4.3, we introduced two methods to add attributes. In this section, we will explain how to edit the XML file externally.

The XML structure consists of two parts: header and body. The header part is the XML declaration, and the body contains the attribute data.



Header: The header section must be filled in strictly according to the specified format. Please do not make any adjustments.

```
<?xml version="1.0" encoding="utf-16"?>
<DeviceConfiguration xmlns="http://tempuri.org/XMLSchema1.0.xsd">
```

Body: For a detailed description of the attributes within the <Attribute> tag, please refer to section 3.4.3. This section will focus on the formatting guidelines. Please note that not all attribute data is required to be filled in, and the "Name" in XML refers to the "Attribute Name".

Attribute	Required Status
VxCANPort	Required
CAN ID	Required
Name	Required
DataConvertToType	Required
Start	Required
Endian	Optional
Math	Optional
Unit	Optional

1. VxCANPort: 0 ~ 255.
2. CAN ID: 0x000 ~ 0x1FFFFFFF.
   (Note: Please include the prefix “0x” when entering the value.)
3. Name: Any string.
   (Note: Each attribute should have a unique name.)
4. DataConvertToType: Bool, SignedByte, UnsignedByte, Int16, UInt16, Int32, UInt32, Float.
5. Start: Integer/float.
   Note: The decimal places after "Start" represent the Start Bit, and the integer part represents the Start Byte. The Start Bit (float) is only applicable when the DataConvertToType is set to "Bool"; otherwise, please enter an integer value. The following image demonstrates an example when the type is set to "Bool", with the Start value entered as 2.4.
6. Endian: Little, Big. (Default: Big)
7. Math: Valid mathematical expression. Please refer to Section 3.4.3 for the input format. (Default: x)
8. Unit: Any string.

Note: Due to the special meanings of certain symbols in XML, please avoid entering the following symbols: [<], [>], [‘], [“], [&].

Tip: The fastest way to become familiar with XML is to manually add attributes by referring to Section 3.4.3, then select "Save XML" from the toolbar. Finally, open the saved XML with an editor and cross-reference it to quickly understand the format of the XML content.

3.5. Editing Chart Attribute Example

In this section, an application scenario will be simulated, demonstrating two different methods for editing attribute information: editing attributes using XML and manually adding attributes.

3.5.1. Example 1

Suppose there is a boiler with two temperature sensors. The temperatures detected by the sensors are transmitted to your device via CAN. The CAN ID is 0x14158A. The first data is stored starting from the 0th byte, and the second data starts from the 4th byte. The format of each data point is a Float (floating point number) and is stored in Big Endian format. The temperature returned by the sensors is in Fahrenheit (F). Today, you want to analyze the returned data and observe the temperature curve of the boiler in Celsius (C) while it's operating. When using the method of manually adding attributes, you would input the following information:

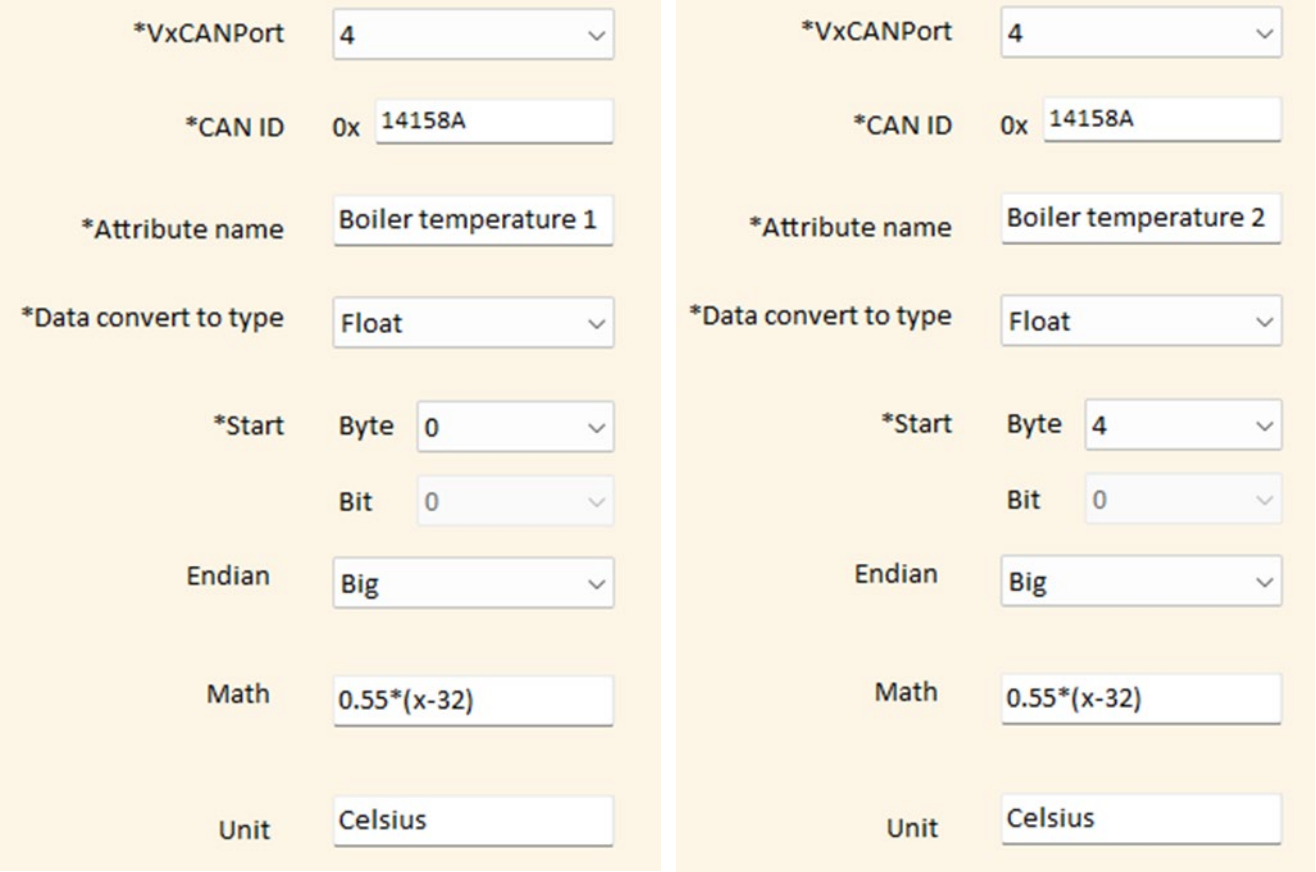


Figure: Manually Input Boiler Temperature Attributes

If you use the method of externally editing XML, you will get the following information after completion:

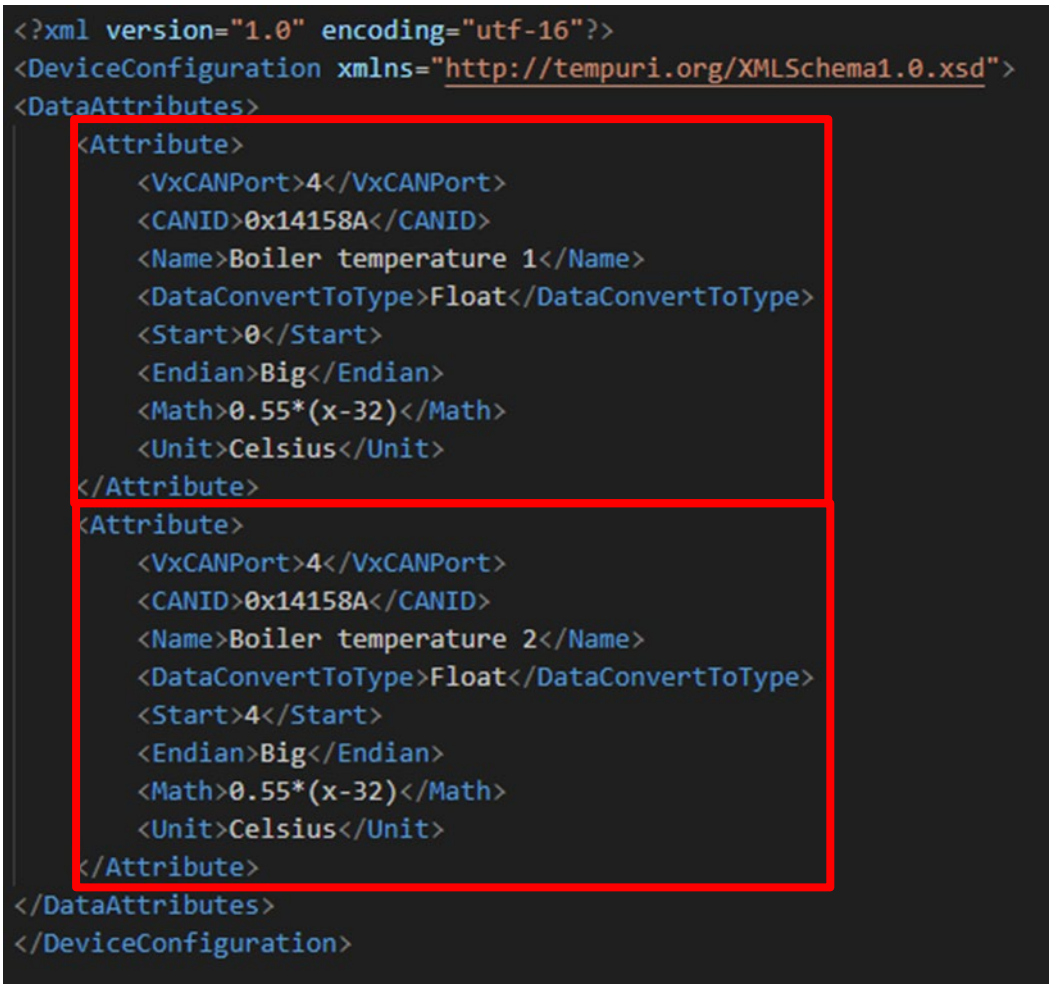


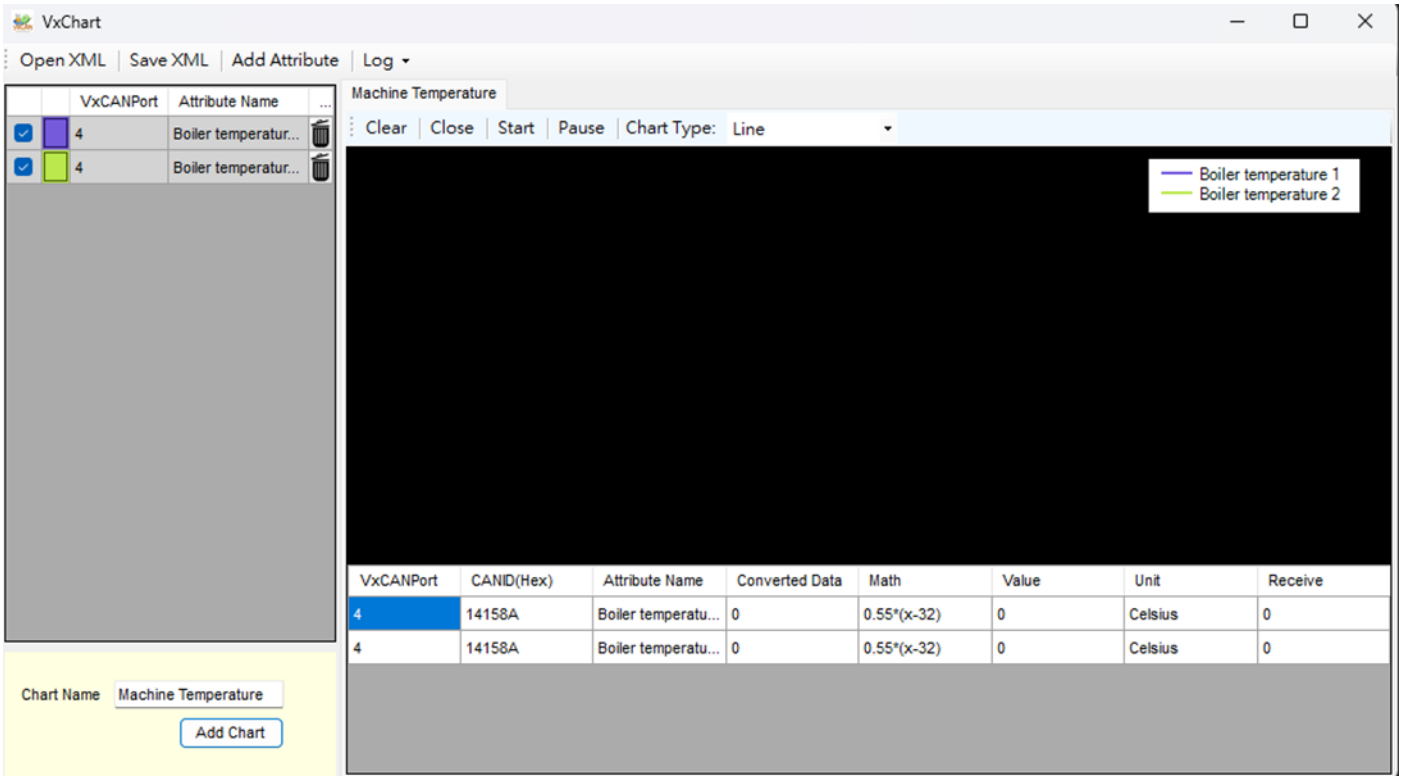
Figure: Editing XML to Input Boiler Temperature Attributes

Both methods of inputting attributes can successfully add the attributes to the Attribute Table in VxChart.

Please note: when editing XML, be sure to strictly follow the format and sequence.

1. VxCANPort: The CAN module used to receive the data returned by the boiler.
2. CANID: 0x14158A, the CAN ID used by the boiler to return temperature information in this example.
3. Name/Attribute Name: Represents the value detected by the corresponding temperature sensor.
4. Data Convert To Type: The data type to which the CAN Data should be converted. In this example, the data type is Float (floating point number).
5. Start: The starting position for reading the data from CAN Data.
6. Endian: In this example, the data returned by the boiler is in Big Endian format.
7. Math: 0.55(x-32). This formula is used to convert Fahrenheit (F) to Celsius (C). The "x" represents the value of the CAN Data after being converted to Float (Fahrenheit).
8. Unit: Represents the unit of the data. After conversion, the unit here is Celsius (C), so "Celsius" is entered.

Screen showing successful addition and use in VxChart:



3.5.2. Example 2

Now, suppose you need to monitor the switch status of two gas valves: one for hydrogen and the other for helium.

The CAN ID for the hydrogen gas valve is 0x1A1597, with the data stored at Start Byte 0 and Start Bit 2. The data type is Boolean (Bool).

The CAN ID for the helium gas valve is 0x1B1597, with the data stored at Start Byte 1 and Start Bit 2. The data type is Boolean (Bool).

Today, you need to observe whether when one gas valve is opened, the other is in a closed state. When manually adding the attributes, you would input the following information:

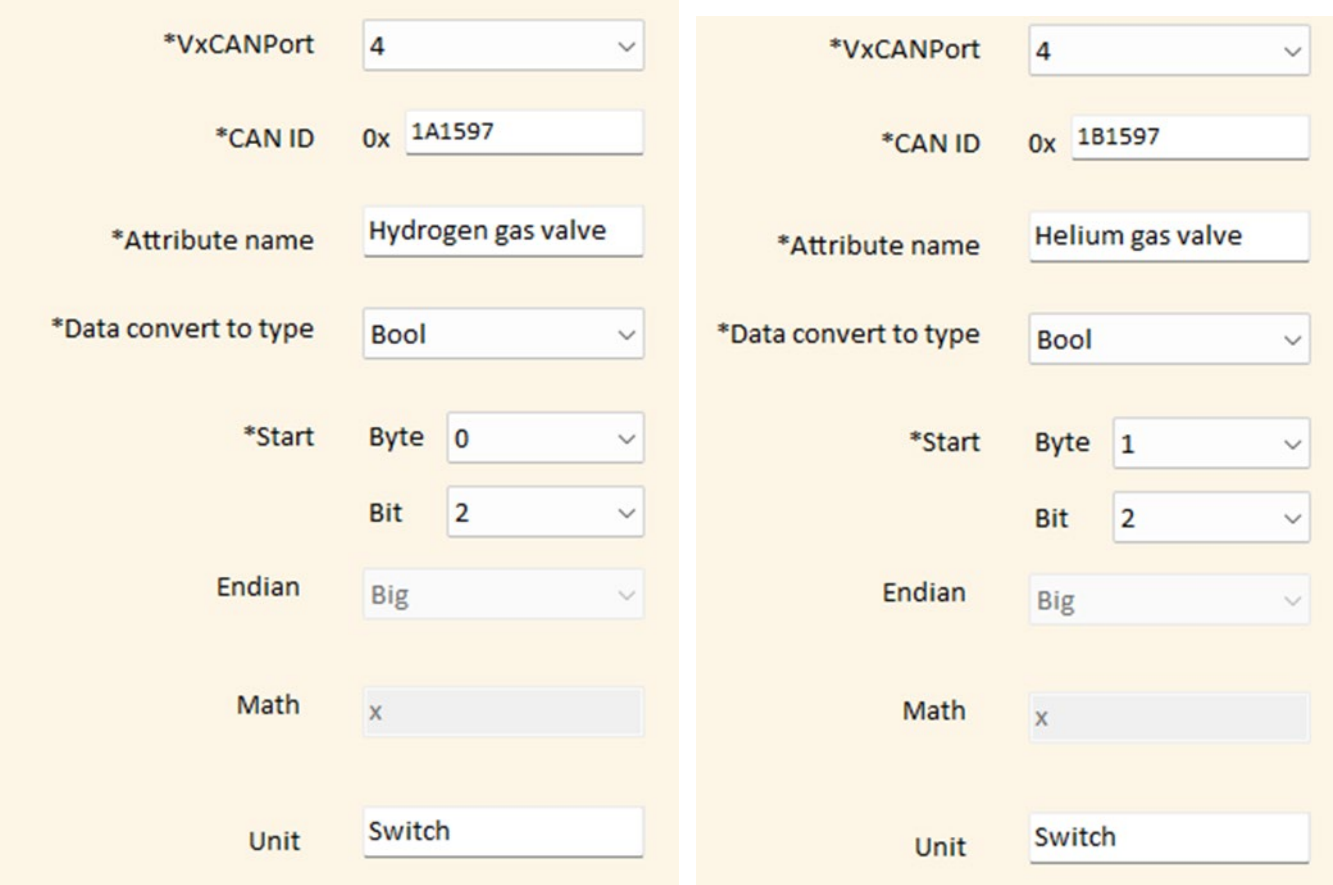


Figure: Manually Input Gas Valve Attributes

If you use the method of externally editing XML, you will get the following information after completion:

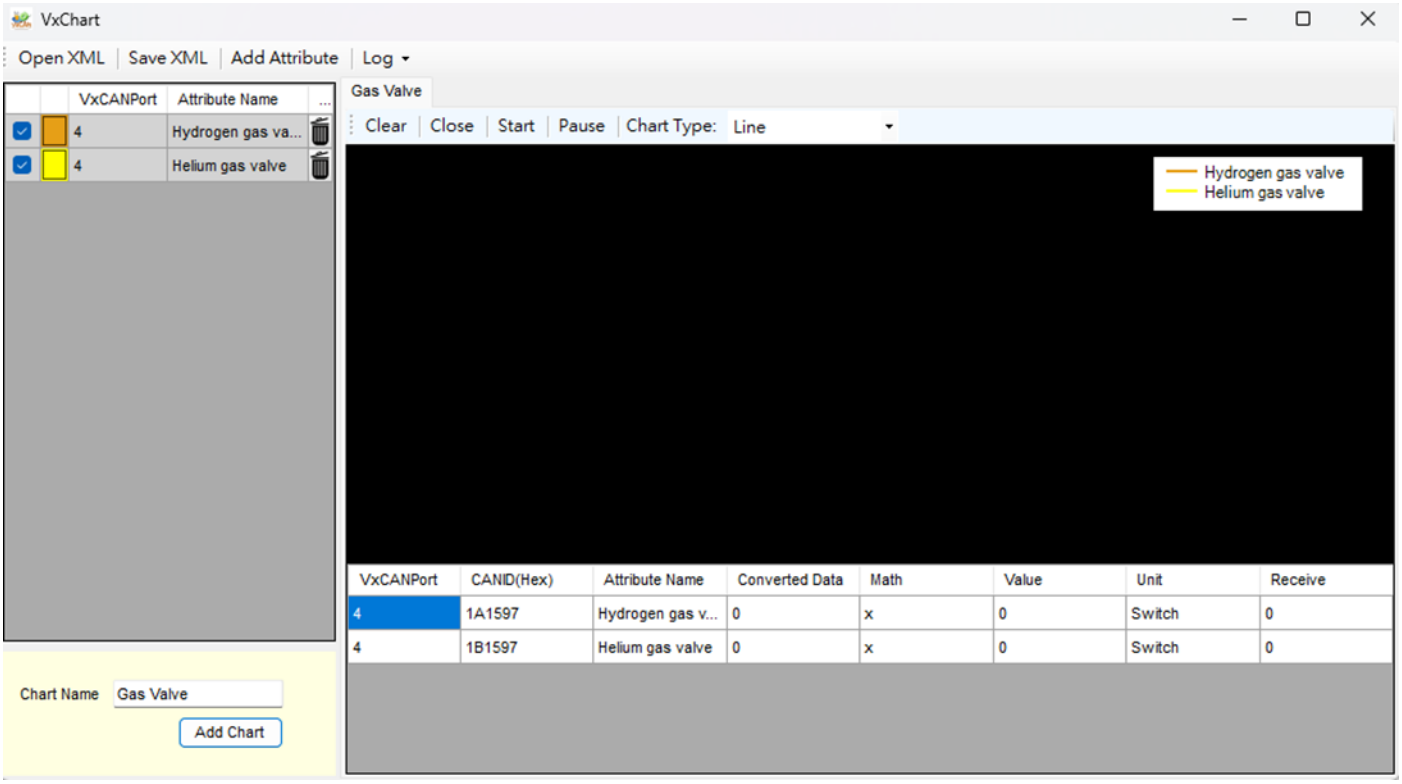
```
<?xml version="1.0" encoding="utf-16"?>
<DeviceConfiguration xmlns="http://tempuri.org/XMLSchema1.0.xsd">
<DataAttributes>
    <Attribute>
        <VxCANPort>4</VxCANPort>
        <CANID>0x1A1597</CANID>
        <Name>Hydrogen gas valve</Name>
        <DataConvertToType>Bool</DataConvertToType>
        <Start>0.2</Start>
        <Unit>Switch</Unit>
    </Attribute>
    <Attribute>
        <VxCANPort>4</VxCANPort>
        <CANID>0x1B1597</CANID>
        <Name>Helium gas valve</Name>
        <DataConvertToType>Bool</DataConvertToType>
        <Start>1.2</Start>
        <Unit>Switch</Unit>
    </Attribute>
</DataAttributes>
</DeviceConfiguration>
```

Figure: Editing XML to Input Gas Valve Attributes

Both methods of inputting attributes can successfully add the attributes to the Attribute Table in VxChart.
Please note: when editing XML, be sure to strictly follow the format and sequence.

1. VxCANPort: The CAN module used to receive the data returned by the two gas valves.
2. CANID: In this example, 0x1A1597 is the CAN ID for the hydrogen gas valve, and 0x1B1597 is the CAN ID for the helium gas valve.
3. Name/Attribute Name: Represents the value monitored by each gas valve.
4. Data Convert To Type: The data type to which the CAN Data should be converted. In this example, the data type is Boolean (Bool).
5. Start: The starting position for reading the data from CAN Data. Note that in the XML, the integer part represents the Start Byte, and the decimal part represents the Start Bit. For example, 1.2 for the helium gas valve means Start Byte 1, Start Bit 2.
6. Endian: In this example, when the data type is Boolean (Bool), this field does not need to be provided.
7. Math: In this example, when the data type is Boolean (Bool), this field does not need to be provided (because the output will only have two states: 0 and 1).
8. Unit: Represents the unit of the data. Here, it refers to the valve switch status, so "Switch" is used as the unit.

Screen showing successful addition and use in VxChart:



4. TroubleShooting

4.1. The Search issue

(1) After plugging a new module in the PC or removing an old module from the PC, please be sure to re-search the module again to update the search list information of the VxCAN_Utility.

(2) UART series modules: Please set UART to "Init" mode with the following configurations: 8 (Data Bits), None (Parity), 1 (Stop Bits), and disable Checksum to enable the module to be detected and controlled.

4.2. The Data Loss Issue

There are two issues which may cause the data lose.

(1) Software receive buffer is overflow:
The VxCAN_Utility is designed for the PC.
If the PC is running with heavy loading, there is not enough resource for the VxCAN_Utility to receive the CAN messages. This may cause the data loss of the VxCAN_Utility.
And then make data loss.

(2) Hardware receive buffer is overflow:
Each CAN module has its maximum reception limitation due to the hardware. If the quality of the CAN messages is over the limitation, the CAN message will be lost.

4.3. The Performance of Group Send Issue

The function of the Group Send is implemented by multi-thread technology. Thus, the performance of the Group Send will be affected due to the CPU loading of the PC.

Appendix A. Error Status Table of Supported Modules

I. I-7530 series and I-7565

Module Status Table of I-7530 series and I-7565:

Module Status	Name	Comment
7530000	HW_UARTCAN_WaitConfig	I-7530 series / I-7565 is waiting for baud rate configuration
7530001	HW_UARTCAN_COMPort_FunctionError	The function call of the command string is wrong.
7530002	HW_UARTCAN_COMPort_PortError	The COM port number is error.
7530003	HW_UARTCAN_COMPort_BaudRateError	The COM port baud rate is error.
7530004	HW_UARTCAN_COMPort_DataError	The data bit is error.
7530005	HW_UARTCAN_COMPort_StopError	The stop bit is error.
7530006	HW_UARTCAN_COMPort_ParityError	The parity bit is error.
7530007	HW_UARTCAN_COMPort_CheckSumError	The checksum of the command string or the response string is error.
7530008	HW_UARTCAN_COMPort_ComPortNotOpen	The COM port has not been opened.
7530009	HW_UARTCAN_COMPort_SendThreadCreateError	The COM port created transmission thread error.
7530010	HW_UARTCAN_COMPort_SendCmdError	The COM port is error when sending command.
7530011	HW_UARTCAN_COMPort_ReadComStatusError	The COM port status is error.
7530012	HW_UARTCAN_COMPort_ResultStrCheckError	The result string from the COM port checked error.
7530013	HW_UARTCAN_COMPort_CmdError	The COM port command is error.
7530015	HW_UARTCAN_COMPort_TimeOut	The COM port has no response. Check the wire connection or the power of the CAN converter.
7530025	HW_UARTCAN_COMPort_ComPortInUse	The COM port is using by other program. Check that any software is using the COM port.
7530026	HW_UARTCAN_COMPort_OpenComError	When opening the COM port, it has errors
7530027	HW_UARTCAN_COMPort_SendSizeError	The COM port reception size error.
7530030	HW_UARTCAN_ModuleNameError	The module is not supported.
7530031	HW_UARTCAN_SendCMDFail	There are errors when sending command to I-7530 series / I-7565.
7530032	HW_UARTCAN_ModuleNoResponse	The module did not reply. Check the wire connection or the power of the CAN converter.
7530033	HW_UARTCAN_ModuleReplyError	The I-7530 series or I-7565 replied the error message.
7530034	HW_UARTCAN_SetBaudRateError	When setting the CAN baud rate of the I-7530 series / I-7565,

		the module replies error.
7530035	HW_UARTCAN_ACKError	All nodes on the Bus that correctly receives a message are expected to send a dominant level in the so-called Acknowledgement Slot in the message. The transmitter will transmit a recessive level here. If the transmitter can't detect a dominant level in the ACK slot, an Acknowledgement Error is signaled.
7530036	HW_UARTCAN_FormError	The CAN message have a fixed format. Those parts are the CRC Delimiter, ACK Delimiter, End of Frame, and so on. If a CAN controller detects an invalid value in one of the CAN fixed fields, a “Form Error” is signaled.
7530037	HW_UARTCAN_CRCError	The CAN chip calculates a 15-bit CRC value. Any node that detects a different CRC in the message than what it has calculated itself will signal a CRC error.
7530038	HW_UARTCAN_StuffError	When the CAN chip detected more than five consecutive bits of the same level, the stuff error is signaled. Check that the terminal resister or add core to filter the noise signal.
7530039	HW_UARTCAN_DataOverrunError	The CAN chip detected the CAN data overrun error. The Overrun error occurs when another CAN data arrives even before the previous CAN data has not been read from the CAN's receive buffer. That means the previous CAN data would be lost.
7530040	HW_UARTCAN_ErrorPassiveMode	The CAN chip is in the error passive mode. When any one of the Tx or Rx error counters
7530041	HW_UARTCAN_CANBusOff	The CAN chip is in the Bus off state. This is the fatal error of the CAN Bus. Check the CAN Bus wire connection, baud rate of all CAN modules or terminal resister.
7530050	HW_UARTCAN_CANBusHasData	There are CAN data before module been activated.

Chip Status Table of I-7530 series and I-7565

AsciiToHex(FF)	Description
Bit 7	Bus Off Mode
Bit 6	Error Passive Mode
Bit 5	Reserved
Bit 4	Overrun Buffer
Bit 3	Stuff Error General
Bit 2	CRC Error General
Bit 1	Form Error General
Bit 0	Acknowledgment Error General

II. I-7540D

Module Status Table of I-7540D:

Module Status	Name	Comment
7540000	HW_7540_WaitConfig	I-7540D is waiting for baud rate configuration.
7540001	HW_7540_OpenSocketFail	It is fail when opening PC Socket.
7540002	HW_7540_ConnectFail	It is fail when connecting with the I-7540D.
7540003	HW_7540_SendCMDFail	There are errors when sending command to I-7540D.
7540004	HW_7540_ModuleNoResponse	There is no response from the I-7540D.
7540005	HW_7540_ModuleReplyError	The module replied error or wrong message.
7540006	HW_7540_SetBaudRateError	When setting the CAN baud rate of the I-7540D, the module replied error.
7540007	HW_7540_TransmitBufferLocked	The CAN transmission buffer is locked. The CPU cannot access the transmit buffer. The message is waiting for transmission or is already in process.
7540008	HW_7540_TransmissionIncomplete	The CAN transmission is incomplete. The transmission complete status bit will remain be signaled (incomplete) until a message is transmitted successfully.
7540009	HW_7540_CANBusOff	The CAN chip is in the Bus off state. This is the fatal error of the CAN Bus. Check the CAN Bus wire connection, baud rate of all CAN modules or terminal resister.

Chip Status Table of I-7540D:

AsciiToHex(FF)	Name	Value	Function
Bit 7 (MSB)	Bus Status	1	Bus-off; the SJA100 is not involved in Bus activities.
		0	Bus-on; the SJA1000 is involved in Bus activities.
Bit 6	Error Status	1	Error; at least one of the error counter has reached or exceeded the CPU warning limit.
		0	Ok; both error counters are below the warning limit.
Bit 5	Transmit Status	1	Transmit; the SJA1000 is transmitting a message.
		0	Idle; no transmit message is in progress.
Bit 4	Receive Status	1	Receive; the SJA1000 is receiving a message.
		0	Idle; no receive message is in progress.
Bit 3	Transmission Complete	1	Complete; last requested transmission has been successfully completed.
		0	Incomplete; the previously requested transmission is not yet completed.

	Status		
Bit 2	Transmit Buffer Status	1	Released; the CPU may write a message into the transmit buffer.
		0	Locked; a message is waiting for transmission or is already in process.
Bit 1	Data Overrun Status	1	Overrun; a message was lost
		0	Absent; no data overrun has occurred
Bit 0 (LSB)	Receive Buffer Status	1	Full; one or more messages are available in the RXFIFO
		0	Empty; no message is available

III. I-7565-H1/H2

Module Status Table of I-7565-H1/H2:

Module Status	Name	Comment
7565000	HW_I7565Hx_WaitConfig	I-7565H1 or I-7565-H2 is waiting for baud rate configuration.
7565001	HW_I7565Hx_ModuleNameError	The I-7565Hx module's name is error.
7565002	HW_I7565Hx_ModuleNotExist	The I-7565Hx module doesn't exist in this COM port.
7565003	HW_I7565Hx_COMPortNotExist	The COM port doesn't exist.
7565004	HW_I7565Hx_COMPortInUse	The COM port is in used.
7565005	HW_I7565Hx_COMPortNotOpen	The COM port has not been opened.
7565006	HW_I7565Hx_CANConfigFail	The CAN hardware in the module initialized fail.
7565007	HW_I7565Hx_CANHARDWAREError	The CAN hardware in the module initialized fail.
7565008	HW_I7565Hx_CANPortNoError	The module doesn't support this CAN port.
7565009	HW_I7565Hx_CANFIDLengthError	The CAN Filter-ID number exceed Max number.
7565010	HW_I7565Hx_CANDevDisconnect	The connection between PC and I-7565Hx is broken.
7565011	HW_I7565Hx_CANTimeOut	There is no response when sending configuration command to the I-7565Hx.
7565012	HW_I7565Hx_CANConfigCmdError	The Configuration command doesn't support.
7565013	HW_I7565Hx_CANConfigBusy	The Configuration command is busy.
7565014	HW_I7565Hx_CANRxBufEmpty	The CAN reception buffer is empty.
7565015	HW_I7565Hx_CANTxBufFull	The CAN transmission buffer is full.
7565016	HW_I7565Hx_CANUserDefISRNoError	The user-defined ISR No. is error (0~7).
7565017	HW_I7565Hx_CANHWSendTimerNoError	The timer of the hardware send number is error(0~4).
7565030	HW_I7565Hx_ACKError	All nodes on the Bus that correctly receives a message are expected to send a dominant level in the so-called Acknowledgement Slot in the message. The transmitter will transmit a recessive level here. If the transmitter can't detect a dominant level in the ACK slot, an Acknowledgement Error is signaled.

7565031	HW_I7565Hx_FormError	The CAN message have a fixed format. Those parts are the CRC Delimiter, ACK Delimiter, End of Frame, and so on. If a CAN controller detects an invalid value in one of the CAN fixed fields, a "Form Error" is signaled.
7565032	HW_I7565Hx_CRCError	The CAN chip calculates a 15-bit CRC value. Any node that detects a different CRC in the message than what it has calculated itself will signal a CRC error.
7565033	HW_I7565Hx_StuffError	When the CAN chip detected more than five consecutive bits of the same level, the stuff error is signaled. Check that the terminal resister or add core to filter the noise signal.
7565034	HW_I7565Hx_DataOverrunError	The CAN chip detected the CAN data overrun error. The Overrun error occurs when another CAN data arrives even before the previous CAN data has not been read from the CAN's receive buffer. That means the previous CAN data would be lost.
7565035	HW_I7565Hx_ErrorPassiveMode	The CAN chip is in the error passive mode. When any one of the Tx or Rx error counters raises above 127, the node will enter "Error Passive" state. The CAN converter still works fine.
7565036	HW_I7565Hx_CANBusOff	The CAN chip is in the Bus off state. This is the fatal error of the CAN Bus. Check the CAN Bus wire connection, baud rate of all CAN modules or terminal resister.
7565040	HW_I7565Hx_LoadDLLError	Load VCI_CAN.DLL Error

Chip Status Table of I-7565-H1/H2:

AsciiToHex(FF)	Description
Bit 7	Bus Off Mode
Bit 6	Error Passive Mode
Bit 5	Arbitration Lost
Bit 4	Overrun Buffer
Bit 3	Stuff Error General
Bit 2	CRC Error General
Bit 1	Form Error General
Bit 0	Acknowledgment Error General

IV. I-7565M-HS

Module Status Table of I-7565M-HS:

模組 狀態	名稱	解釋
17565000	HW_I7565MHS_WaitConfig	Wait for CAN Baud rate configuration.
17565001	HW_I7565MHS_DEV_OPError	OP field of the configuration command error.
17565002	HW_I7565MHS_DEV_FCError	FC field of the configuration command error.
17565003	HW_I7565MHS_DEV_DLError	DL field of the configuration command error.
17565004	HW_I7565MHS_DEV_WriteDataError	Fail to write data into device.
17565010	HW_I7565MHS_COMM_TIMEOUT	No response when sending configuration commands to I-7565M-HS.
17565011	HW_I7565MHS_INVALID_Port	Invalid Port.
17565012	HW_I7565MHS_CANRxNoData	The CAN receive buffer is empty.
17565013	HW_I7565MHS_CANTxBufOverflow	The CAN transmit buffer is full.
17565014	HW_I7565MHS_WriteListCIDNumErr	寫入列表 CID 號碼錯誤。
17565021	HW_I7565MHS_INVALID_DEVICE	Invalid device.
17565022	HW_I7565MHS_DEVICE_IN_USED	Device already in used.
17565023	HW_I7565MHS_DEVICE_NOT_EXIST	Device not exist.
17565024	HW_I7565MHS_DEVICE_INFO_ERROR	Get device information error.
17565025	HW_I7565MHS_INVALID_USB_PACKAGE_SIZE	Invalid USB package size
17565026	HW_I7565MHS_WRITE_FILE_FAIL	Write file fail.
17565027	HW_I7565MHS_USB_TX_OVERFLOW	USB Tx buffer overflow.
17565030	HW_I7565MHS_EXCEED_DEVICE_NUM	Exceed maximum supported USB device
17565031	HW_I7565MHS_DEVICE_NOT_OPEN	USB device not open
17565040	HW_I7565MHS_LoadDLLError	Failed to load CAN_HS.dll.
17565042	HW_I7565MHS_Connecting_01_Start	Still attempting to open I-7565M-HS.

V. I-7565M-FD

Module Status Table of I-7565M-FD:

模組 狀態	名稱	解釋
27565000	HW_I7565MFD_WaitConfig	Wait for CAN Baud rate configuration.
27565001	HW_I7565MFD_DEV_OPError	OP field of the configuration command error.
27565002	HW_I7565MFD_DEV_FCError	FC field of the configuration command error.
27565003	HW_I7565MFD_DEV_DLError	DL field of the configuration command error.
27565004	HW_I7565MFD_DEV_WriteDataError	Fail to write data into device.
27565010	HW_I7565MFD_COMM_TIMEOUT	There is no response when sending configuration

		command to the I-7565M-FD.
27565011	HW_I7565MFD_INVALID_Port	Invalid Port.
27565012	HW_I7565MFD_CANRxNoData	The CAN reception buffer is empty.
27565013	HW_I7565MFD_CANTxBufOverflow	The CAN transmission buffer is full.
27565014	HW_I7565MFD_BitRateNotSupport	Invalid arbitration baud rate or invalid data baud rate.
27565015	HW_I7565MFD_ExceedCANFilterID	CAN Filter ID list size parameter not support."
27565021	HW_I7565MFD_INVALID_DEVICE	Invalid I-7565M-FD device.
27565022	HW_I7565MFD_DEVICE_IN_USED	The I-7565M-FD module has already in used.
27565023	HW_I7565MFD_DEVICE_NOT_EXIST	The I-7565M-FD module doesn't exist.
27565024	HW_I7565MFD_DEVICE_INFO_ERROR	Get I-7565M-FD information error.
27565025	HW_I7565MFD_INVALID_USB_PACKAGE_SIZE	Invalid USB package size.
27565026	HW_I7565MFD_WRITE_FILE_FAIL	Write file fail.
27565027	HW_I7565MFD_USB_TX_OVERFLOW	USB Tx buffer overflow.
27565030	HW_I7565MFD_EXCEED_DEVICE_NUM	Exceed maximum supported USB device.
27565031	HW_I7565MFD_DEVICE_NOT_OPEN	The I-7565M-FD module not open yet.
27565040	HW_I7565MFD_LoadDLLError	Loading CAN_FD.dll failed.
27565042	HW_I7565MFD_Connecting_01_Start	Still trying to open I-7565M-FD.

VI. PISO-CAN series board

Module Status Table of PISO-CAN series board:

Module Status	Name	Comment
9030000	HW_PISOCAN_WaitConfig	Wait for CAN Baud rate configuration.
9030001	HW_PISOCAN_DriverError	The windows driver of the CAN board is error.
9030002	HW_PISOCAN_ActiveBoardError	This CAN board can't be activated.
9030003	HW_PISOCAN_BoardNumberError	The CAN board number exceeds the maximum board number (7).
9030004	HW_PISOCAN_PortNumberError	The CAN port number exceeds the maximum port number.
9030005	HW_PISOCAN_ResetError	CAN chip hardware reset error.
9030006	HW_PISOCAN_SoftResetError	CAN chip software reset error.
9030007	HW_PISOCAN_InitError	CAN chip initiation error.
9030008	HW_PISOCAN_ConfigError	CAN chip configure error.
9030009	HW_PISOCAN_SetACRError	Set to Acceptance Code Register error.

9030010	HW_PISOCAN_SetAMRError	Set to Acceptance Mask Register error.
9030011	HW_PISOCAN_SetBaudRateError	Set CAN baud rate error.
9030012	HW_PISOCAN_EnableRxIrqFailure	Enable CAN chip receive interrupt failure.
9030013	HW_PISOCAN_DisableRxIrqFailure	Disable CAN chip receive interrupt failure.
9030014	HW_PISOCAN_InstallIrqFailure	Installing PCI board IRQ failure.
9030015	HW_PISOCAN_RemoveIrqFailure	Removing PCI board IRQ failure.
9030016	HW_PISOCAN_TransmitBufferLocked	The CAN transmission buffer is locked. The CPU cannot access the transmit buffer. The message is waiting for transmission or is already in process.
9030017	HW_PISOCAN_TransmitIncomplete	The CAN transmission is incomplete. The transmission complete status bit will remain be signaled (incomplete) until a message is transmitted successfully.
9030018	HW_PISOCAN_ReceiveBufferEmpty	CAN chip RXFIFO is empty
9030019	HW_PISOCAN_DataOverrun	The CAN chip detected the CAN data overrun error. The Overrun error occurs when another CAN data arrives even before the previous CAN data has not been read from the CAN's received buffer. That means the previous CAN data would be lost.
9030020	HW_PISOCAN_ReceiveError	Receive data is not completed.
9030021	HW_PISOCAN_SoftBufferIsEmpty	Software buffer in driver is empty.
9030022	HW_PISOCAN_SoftBufferIsFull	Software buffer in driver is full.
9030023	HW_PISOCAN_TimeOut	Function no response and timeout.
9030024	HW_PISOCAN_InstallIsrError	Installing user ISR failure.
9030030	HW_PISOCAN_CANBusOff	The CAN chip is in the Bus off state. This is the fatal error of the CAN Bus. Check the CAN Bus wire connection, baud rate of all CAN modules or terminal resister.
9030031	HW_PISOCAN_CANError	CAN Bus have some error.

Chip Status Table of PISO-CAN series board:

AsciiToHex(FF)	Name	Value	Function
Bit 7 (MSB)	Bus Status	1	Bus-off; the SJA100 is not involved in Bus activities
		0	Bus-on; the SJA1000 is involved in Bus activities
Bit 6	Error Status	1	Error; at least one of the error counter has reached or exceeded the CPU warning limit
		0	Ok; both error counters are below the warning limit
Bit 5	Transmit Status	1	Transmit; the SJA1000 is transmitting a message
		0	Idle; no transmit message is in progress
Bit 4	Receive Status	1	Receive; the SJA1000 is receiving a message
		0	Idle; no receive message is in progress
Bit 3	Transmission	1	Complete; the previously requested transmission is not yet completed

	Complete Status	0	Incomplete; the previously requested transmission is not yet complement
Bit 2	Transmit Buffer Status	1	Released; the CPU may write a message into the transmit buffer
		0	Locked; a message is waiting for transmission or is already in process
Bit 1	Data Overrun Status	1	Overrun; a message was lost
		0	Absent; no data overrun has occurred
Bit 0 (LSB)	Receive Buffer Status	1	Full; one or more messages are available in the RXFIFO
		0	Empty; no message is available

VII. PISO-CAN-FD series board

Module Status Table of PISO-CAN-FD series board:

Module Status	Name	Comment
19030000	HW_PISOCANFD_WaitConfig	Wait for CAN Baud rate configuration.
19030001	HW_PISOCANFD_INIT_DRIVER_ERROR	An error occurred while initializing the driver.
19030002	HW_PISOCANFD_COMM_DRIVER_ERROR	An error occurred while communicating with the driver.
19030003	HW_PISOCANFD_DRIVER_UNINIT_ERROR	The driver is not initialized.
19030004	HW_PISOCANFD_ACTIVE_BOARD_ERROR	An error occurred while activating the board.
19030005	HW_PISOCANFD_BOARD_ALREADY_ACTIVE_ERROR	The board has already been activated.
19030006	HW_PISOCANFD_BOARD_UNACTIVE_ERROR	The board is not activated.
19030007	HW_PISOCANFD_BOARD_NUMBER_ERROR	The board value exceeds the valid range.
19030008	HW_PISOCANFD_PORT_NUMBER_ERROR	The CAN port value exceeds the valid range.
19030009	HW_PISOCANFD_PORT_UNINIT_ERROR	The CAN port is not initialized.
19030032	HW_PISOCANFD_FLT_FRAME_FORMAT_ERROR	CAN filter format setting must be standard identifier(0) or extended identifier(1).
19030033	HW_PISOCANFD_FLT_ACC_ACM_INCONSISTENCY_ERROR	CAN filter setting of inMask and inAcceptance arguments should be “inAcceptance & inMask == inAcceptance".
19030034	HW_PISOCANFD_FLT_STD_ACC_TOOLARGE_ERROR	The standard identifier of CAN acceptance filter setting large than 0x7FF.
19030035	HW_PISOCANFD_FLT_EXT_ACC_TOOLARGE_ERROR	The extended identifier of CAN acceptance filter setting large than 0x1FFFFFFF.
19030036	HW_PISOCANFD_FLT_STD_ACM_TOOLARGE_ERROR	The standard identifier of CAN mask filter setting large than 0x7FF.

19030037	HW_PISOCANFD_FLT_EXT_ACM_TOOLARGE_ERROR	The extended identifier of CAN mask filter setting large than 0x1FFFFFFF.
19030048	HW_PISOCANFD_INVALID_OPMODE_ERROR	CAN operation mode setting is not support.
19030049	HW_PISOCANFD_SOFTBUFF_IS_EMPTY	CAN receive buffer is empty.
19030050	HW_PISOCANFD_NORM_BITRATE_NOT_SUPP_ERROR	CAN normal bit rate setting is not support.
19030051	HW_PISOCANFD_DATA_BITRATE_NOT_SUPP_ERROR	CAN data bit rate setting is not support.
19030052	HW_PISOCANFD_DATABR_LESS_THAN_NORMBR_ERROR	CAN data bit rate setting is less than normal bit rate setting.
19030053	HW_PISOCANFD_INVALID_NORMAL_SAMPLEPOINT	CAN normal baud rate sample point setting is not support.
19030054	HW_PISOCANFD_INVALID_DATA_SAMPLEPOINT	CAN data baud rate sample point setting is not support.
19030257	HW_PISOCANFD_SYS_BOARD_ALREADY_ACTIVE	The board has already been activated.
19030258	HW_PISOCANFD_SYS_PORT_NUMBER_ERROR	The Port value is invalid.
19030259	HW_PISOCANFD_SYS_INSTALL_IRQ_ERROR	An error occurred while installing the system interrupt.
19030272	HW_PISOCANFD_SYS_CAN_RESET_ERROR	Reset CAN chip error.
19030273	HW_PISOCANFD_SYS_CAN_OPMODE_ERROR	CAN chip operation mode error.
19030274	HW_PISOCANFD_SYS_REQ_CONFIG_MODE_TIMEOUT	Change CAN chip to configuration mode timeout.
19030275	HW_PISOCANFD_SYS_CHANGE_MODE_TIMEOUT	Change CAN chip operation mode timeout.
19030276	HW_PISOCANFD_SYS_SPI_FULLSPEED_RAMTEST_ERROR	Test CAN chip RAM data error.
19030277	HW_PISOCANFD_SYS_CAN_FilterIdx_OutOfRange	CAN filter index setting is out of range.
19030288	HW_PISOCANFD_SYS_CAN_TRANSMIT_ERR_DETECTED	Detect an error when transmitting a CAN message.
19030289	HW_PISOCANFD_SYS_SOFTBUFF_IS_EMPTY	CAN receive buffer of driver is empty.
19030290	HW_PISOCANFD_SYS_SOFTBUFF_IS_FULL	CAN receive buffer of driver is full.
19030291	HW_PISOCANFD_SYS_HARDWAREBUFF_IS_EMPTY	CAN receive buffer of CAN chip is empty.
19030292	HW_PISOCANFD_SYS_HARDWAREBUFF_IS_FULL	CAN transmit buffer of CAN chip is full.
19030300	HW_PISOCANFD_CANBusOff	The CAN chip is in the bus off state.This is the fatal error of the CAN bus. Check the CAN bus wire connection, baud rate of all CAN modules or terminal resister.
19030301	HW_PISOCANFD_CANError	CAN bus have some errors.

19030400	HW_PISOCANFD_BoardNotMatched	The expected communication board model does not match the actual detected board model.
19031000	HW_PISOCANFD_GroupSendIsStart	Group Send start,plz close or wait for finishing.
19031001	HW_PISOCANFD_CoreThreadDontExit	Don't Create Core Thread
19031002	HW_PISOCANFD_GroupThreadDontExit	Don't Create Group Thread