NexAIoT Co., Ltd

nDAS Serial

User Manual (1.0)

Document Release History

Date	Software	Manual	Log
2023/05/31	1.0.0.1n	Beta 1.7	Cloud, Network(Bridge), wifi
2023/06/09	1.0.0.1n	Beta 1.8	Analog Input, OLED
2023/06/13	1.0.0.1n	Beta1.9	OPCUA
2023/07/24	1.0.0.1	Beta1.10	RestFul, LED, DIP function, Safe Mode
2023/08/25	1.0.0.4	1.01	User Manual MP Version
2023/09/20	1.0.0.5beta	1.02	Change Modbus rule 64->32, Modbus channel 64->128

Catalog

Chapter 1. nDAS Introduction

The nDAS software specification is shown in the Table 1-1. Note：To ensure system security, please logout after using the system.

Table 1-1 nDAS software specification

nDAS	
System	
Browser	Google Chrome
Interface Language	English, Traditional Chinese, Simplified Chinese, Japanese, Korean
System Log	Maximum of 500 lines allowed.
Login Timeout	The system will automatically log you out if there is no activity for 30 minutes.
General mode allows a maximum number of connections	16
The database page allows a maximum number of connections	1
I/O Module	
ns050	8 Digital Input Channel, 8 Digital Output
ns051	16 Digital Input Channel
ns056	16 Digital Output Channel
ns017	8 Analog Input Channel
Script specification	
CALC	Up to 100. (00 ~ 99)。
MODBUSSTR	The addresses can range from 1 to 65535, with data stored in Input Register (3x)
MODBUSINT	The addresses can range from 1 to 65535, with data stored in Input Register (3x)
MODBUSUINT	The addresses can range from 1 to 65535, with data stored in Input Register (3x)
MODBUSBIT	The addresses can range from 1 to 65535, with data stored in Input Status (1x)
MODBUSHSTR	The addresses can range from 1 to 65535, with data

	stored in Holding Register (4x)
MODBUSHINT	The addresses can range from 1 to 65535, with data stored in Holding Register (4x)
MODBUSHUINT	The addresses can range from 1 to 65535, with data stored in Holding Register (4x)
MODBUSHBIT	The addresses can range from 1 to 65535, with data stored in Coil Status (0x)
Communication protocol	
Modbus	Data is stored in Coil Status (0x) and Holding Register (4x). System status is stored in Input Status (2x), where each status is represented by 1 bit.

Chapter 2. nDAS Startup and Usage

2.1 Launching nDAS

Before using this software, it is essential to connect the power supply. For detailed instructions on the startup procedure, please refer to the electrical schematic diagram.

2.2 DIP Function Definition.

The DIP switch is used for initializing LAN port settings, Wi-Fi settings, and safe mode in the nDAS software. To enable the functionality of the DIP switch, it is necessary to power off and restart the nDAS device. The definitions of the DIP switch functions are provided in Table 2-1.

Table 2-1 DIP function definition

DIP Status	Content
DIP SW1 ON	LAN1 initial IP: 10.0.0.1 LAN2 initial IP: 11.0.0.1
DIP SW1 OFF & DIP SW2 ON	Wi-Fi is AP Mode
DIP SW1 ON & DIP SW2 ON	Safe Mode, please refer to section 2.2.1

Please note that in Safe Mode, although an IP address is still configured for the LAN, if an external network interface is connected, there will be two IP addresses present.

2.2.1 Safe Mode

The nDAS software provides users with a safe mode option. In case the software experiences abnormalities that prevent the web page from opening correctly, you can enter safe mode and choose how to restart the nDAS software (as shown in Figure 2-1). Please refer to Table 2-2 for detailed instructions and explanations.

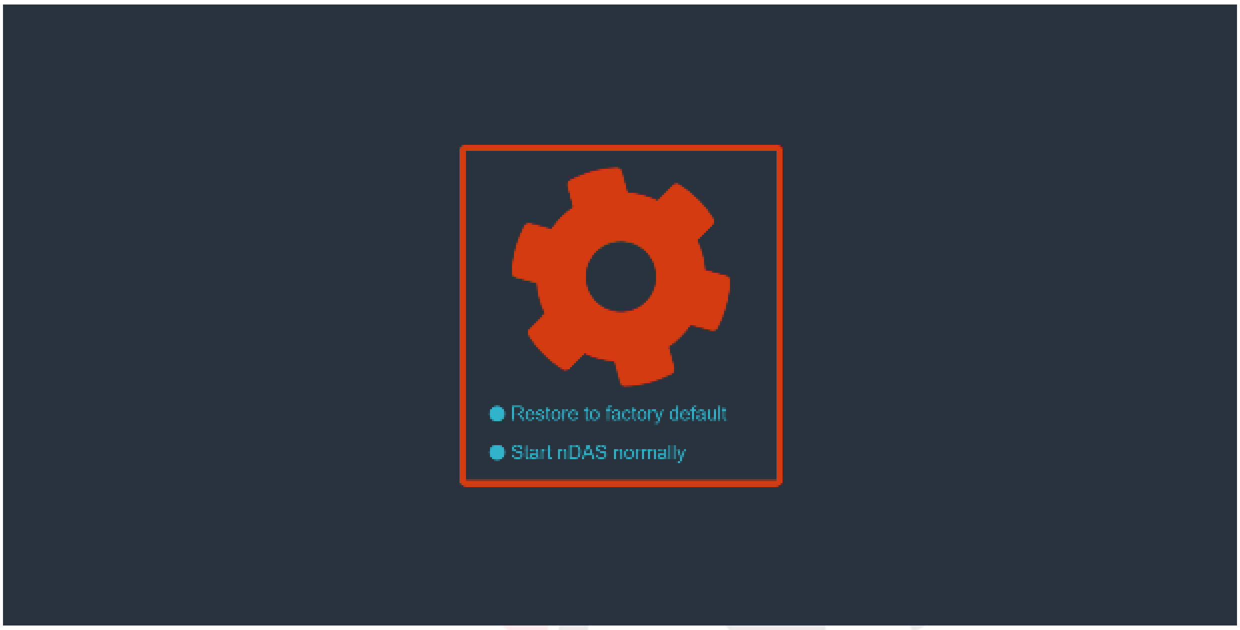


Figure 2-1 Safe Mode

Table 2-2 Restart the nDAS software in safe mode

	說明
Restore to factory default	Remove the preferences, OPCUA, database, project, user's operator, and .htpasswd, then restart.
Normal	The automatic loading of projects is not enabled (please refer to section 4.1.1.1 for information on how to enable this feature). You can access the nDAS software to modify the settings. This situation typically arises when there are scripts or other configurations within the project that cause the autoload feature to result in a crash of the nDAS software. However, it is essential to have knowledge of the login credentials (including the account password if Sign-In Password is enabled, please refer to section 4.1.1.1 for information on this feature).

2.3 LED Definition

The definitions of the LED indicator lights are depicted in Table 2-3.

Table 2-3 LED definition

	LED definition
LED1	System status: The green light signifies normal operation with network services enabled, while the red light indicates an error. When nDAS is powered on, the green light briefly turns off, it will turn on when nDAS system boot up.
LED2	Network status: In Wi-Fi mode, the red light signifies being in AP mode, while the green light indicates being in infrastructure mode. If nDAS doesn't have Wi-Fi, the green light signifies a wired network connection, while its absence indicates no wired network connection.
LED3	For Wi-Fi in infrastructure mode, the intensity of the Wi-Fi wireless signal is indicated by the strength of the green light.
LED4	For Wi-Fi in infrastructure mode, the intensity of the Wi-Fi wireless signal is indicated by the strength of the green light.
LED5	For Wi-Fi in infrastructure mode, the intensity of the Wi-Fi wireless signal is indicated by the strength of the green light.
LED6	For Wi-Fi in infrastructure mode, the intensity of the Wi-Fi wireless signal is indicated by the strength of the green light.

2.4 Software Mode

This software has two modes: the view mode and the general mode (with administrative privileges). The introduction of each mode will be explained in subsequent sections.

Note：Data captured by the software will only be written into the database and transmitted during operation in these two modes.

2.4.1 View Mode

For unauthorized users, only viewing software operations and chart presentation status is available. The chart presentation area displays the I/O status(as shown in Figure 2-2), and the Modbus Channel chart presentation displays the Modbus status (as shown in Figure 2-3)。

Note：If Modbus Channel is not enabled, the status of Modbus Channel in the status list will be hidden.

Note：If the drawing option of Modbus Channel is not selected, the Modbus chart display area will not show the chart.

Figure 2-2 View Mode

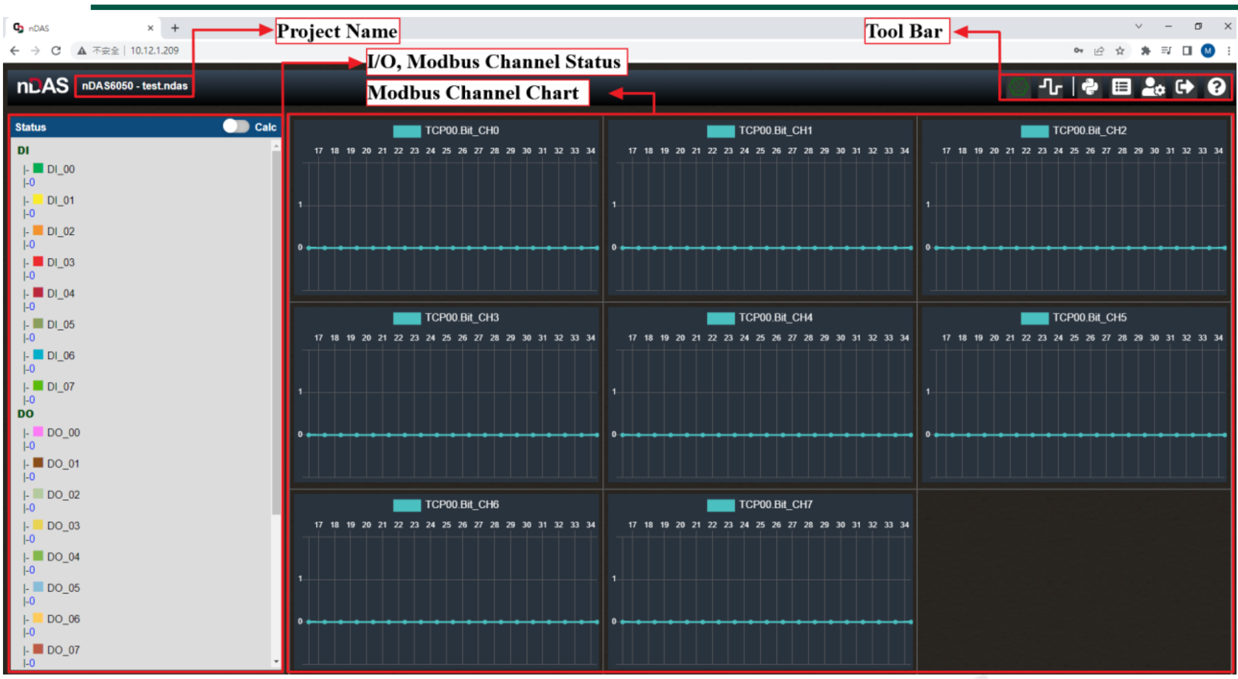


Figure 2-3 View Mode-Status of Chart Presentation

2.4.2 General mode (with administrative privileges)

A user with administrative privileges (as shown in Figure 2-4) has the ability to control the software's startup and shutdown, as well as load other projects. By clicking on the toggle view icon in the toolbar, it is possible to switch between normal and management modes (as shown in Figure 2-5)。

Note：If Modbus Channel is not enabled, the status of Modbus Channel in the status list will be hidden.

Note：If the drawing option of Modbus Channel is not selected, the Modbus chart display area will not show the chart.

Figure 2-4 General mode (with administrative privileges)

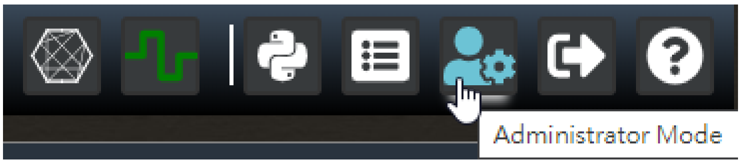


Figure 2-5 General mode (with administrative privileges) - Click on the "Switch Administrator Mode" icon.

2.5 OLED Display

nDAS OLED provides essential functionality settings, allowing users to view nDAS parameters or execute nDAS configurations through the OLED screen.

2.5.1 Introduction to nDAS OLED Icons.

nDAS OLED is designed with a 4-grid menu layout, allowing users to access various function settings through the menu, as show in Figure 2-6.

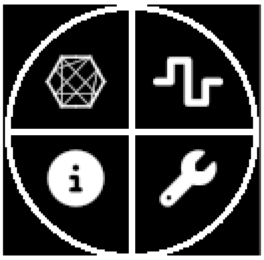
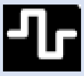


Figure 2-6 Main Menu Screen

2.5.1.1 Primary Function Menu

Icons	Introduction	Icons	Introduction
	Modbus		I/O
	System Information		System Setting

2.5.1.2 Modbus Function Menu

Icons	Introduction	Icons	Introduction
COM0	COM Port	TCP00	TCP/IP Port

2.5.1.3 I/O Function Menu

Icons	Introduction	Icons	Introduction
DI	Digital Input	DO	Digital Output
AI	Analog Input		

2.5.1.4 Information Function Menu

Icons	Introduction	Icons	Introduction
	System Information		Network Interface
	System Time/Date		Free Space

2.5.1.5 Setting Function Menu

Icons	Introduction	Icons	Introduction
	Wireless(WIFI)		Project
	Screen Rotation		System Logout/Login
	Reboot System		Simulator

2.5.2 nDAS OLED Function Description

2.5.2.1 Modbus Function Menu

The Modbus Function Menu on the OLED screen includes the COM Port or TCP Port for Modbus Master (as shown in Figure 2-7). If the Modbus channel is already activated, clicking on the Port will bring up the menu screen, as shown in Figure 2-8(using Modbus TCP Master as an example). The Modbus channel status will be displayed on the screen. If the R/W mode includes Write Mode, clicking on the status position or rotating the OLED screen allows for value input. For detailed functionality information, please refer to section 4.1.2.

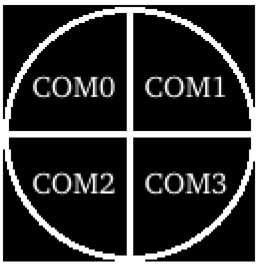


Figure 2-7 Modbus Master Port

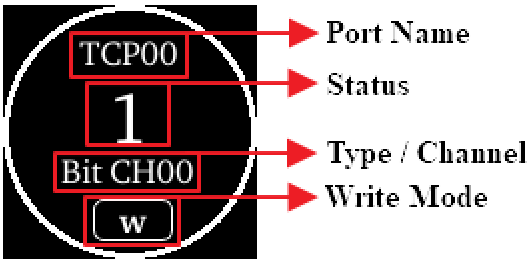


Figure 2-8 OLED Modbus TCP Master

2.5.2.2 I/O Function Menu

The I/O Function Menu on the OLED screen displays the current nDAS I/O modules, as shown in Figure 2-9(using Simulator as an example).

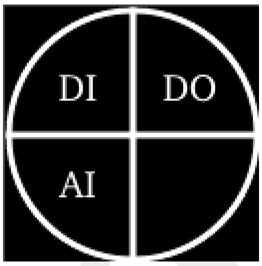


Figure 2-9 Simulator I/O Module

2.5.2.2.1 DI

Clicking on DI will bring up the menu screen (as shown in Figure 2-10), displaying the DI status on the screen. Rotating the OLED allows for channel switching. For detailed functionality information, please refer to section 4.1.3.1.

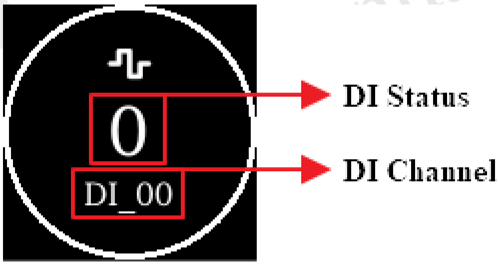


Figure 2-10 DI Module

2.5.2.2.2 DO

Clicking on DO will bring up the menu screen (as shown in Figure 2-11), displaying the DO status on the screen. Rotating the OLED allows for channel switching. If Write Mode is activated and the OLED is rotated, the output value can be changed. For detailed functionality information, please refer to section 4.1.3.2.

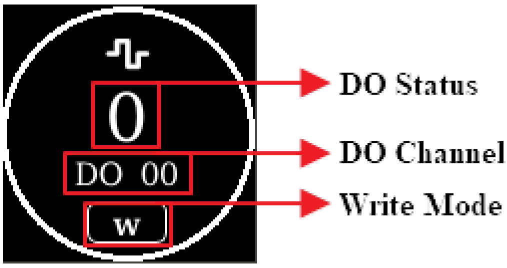


Figure 2-11 DO Module

2.5.2.2.3 AI

Clicking on AI will bring up the menu screen (as shown in Figure 2-12), displaying the AI status on the screen. Rotating the OLED allows for channel switching. For detailed functionality information, please refer to section 4.1.3.3.

Figure 2-12 AI Module

2.5.2.3 Information Function Menu

The Information Function Menu on the OLED screen includes system information, network interface, system time, and storage space, as shown in Figure 2-13.

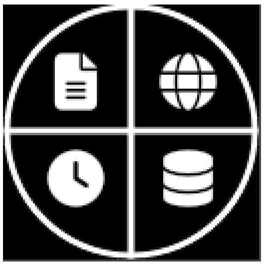


Figure 2-13 Information Function Menu

2.5.2.3.1 System Information

Icons	Introduction
	Show system version.
	Show firmware version.
	Show current project.
	Show I/O type.

2.5.2.3.2 Network Interface

Icons	Introduction
	Show the network interface name and IP address.
	Show the network interface name and MAC information.

2.5.2.3.3 System Time/Date

Icons	Introduction
	Show the system time. Touch the icon above will display the system time zone.
	Show the system time zone.

2.5.2.3.4 Free Space

Icons	Introduction
	Show free space.

2.5.2.4 Setting Function Menu

The Setting Function Menu on the OLED screen includes wireless network, load project, OLED screen rotation, system logout/login, system restart, and simulator, as shown in Figure 2-14.

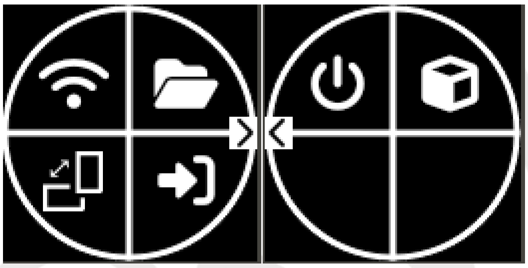


Figure 2-14 Setting Function Menu

2.5.2.4.1 Wireless (WIFI)

The wireless network supports both AP (Access Point) and Infrastructure modes, as shown in Table 2-4 and Table 2-5. For detailed functionality information, please refer to section 4.1.1.3.

Table 2-4 AP Mode

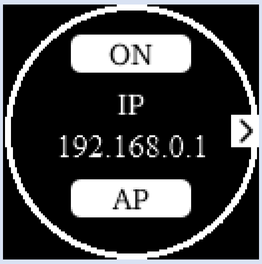
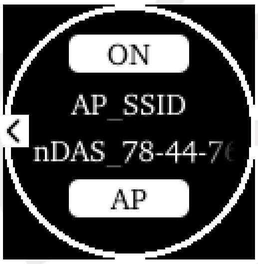
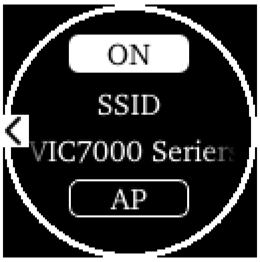
Icons	Introduction
	1、Show AP Mode IP。 2、Touch the button above can enable/disable the wireless network functionality, as shown in Figure 2-15. 3、Touch the button below can switch between AP and Infrastructure modes, as shown in Figure 2-16.
	1、Show AP Mode SSID。 2、Touch the upper button enables/disables the wireless network functionality, as depicted in Figure 2-15. 3、Touch the lower button allows you to switch between AP and Infrastructure Mode, as shown in Figure 2-16.

Table 2-5 Infrastructure Mode

Icons	Introduction
	1、Show Infrastructure Mode IP。 2、Touch the upper button enables or disables the wireless network functionality, as depicted in Figure 2-15. 3、Touch the lower button allows you to switch between AP and Infrastructure Mode, as shown

	in Figure 2-16.
	1、Show Infrastructure Mode SSID。 2、Touch the upper button, you can enable or disable the wireless network functionality, as depicted in Figure 2-15. 3、Touch the lower button, you can toggle between AP and Infrastructure Mode, as shown in Figure 2-16.

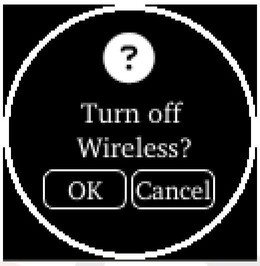


Figure 2-15 Activate/Deactivate wireless network.

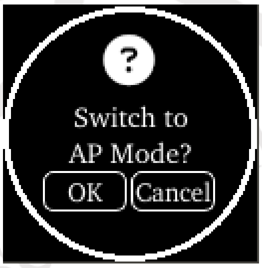


Figure 2-16 Switch to AP Mode.

2.5.2.4.2 Load Project

The Load Project screen allows you to rotate the OLED to switch between different projects, as shown inFigure 2-17. Clicking the button loads the selected project, as depicted in Figure 2-18. For detailed functionality information, please refer to section 3.2.

Figure 2-17 Project Selection

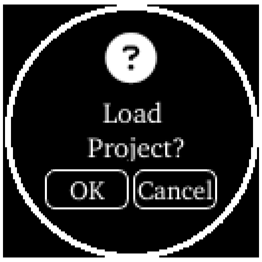


Figure 2-18 Load Project

Note: It is necessary to activate the PIN Password option and log in on the OLED screen. Please refer to section 2.5.2.4.4 for more details.

2.5.2.4.3 Screen Rotation

Touch the menu screen will bring up the interface shown in Figure 2-19. Clicking on "Angle" allows you to physically rotate the OLED screen.

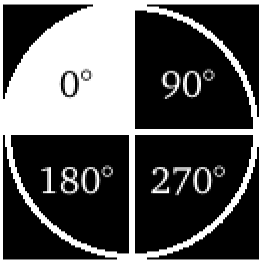


Figure 2-19 OLED screen rotation

2.5.2.4.4 System Logout/Login

It is necessary to activate the PIN Password option (please refer to section 4.1.1.5). Enter the password and click "OK".

2.5.2.4.5 System Reboot

Touch the system restart icon displays the screen as shown in Figure 2-20, asking the user if they want to restart the system.

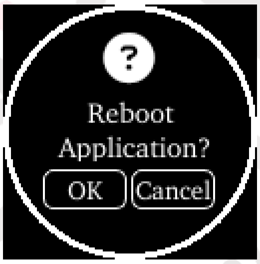


Figure 2-20 Restart the system.

2.5.2.4.6 Simulator

Touch the system restart icon displays the screen (as shown in Figure 2-21), and clicking the button allows you to enable or disable the simulator (as shown in Figure 2-22).

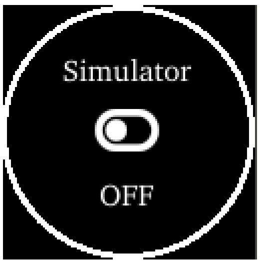


Figure 2-21 Simulator Mode

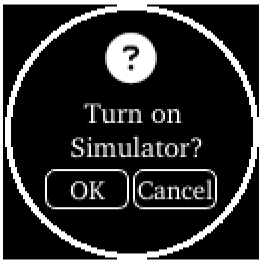


Figure 2-22 Whether to enable/disable the simulator.

2.6 Login

Upon confirming the activation of nDAS, open a web browser on your computer and input the IP address of nDAS in the URL bar. This will display the view mode (as shown in Figure 2-23). Clicking on the login icon in the toolbar (as shown in Figure 2-24) to display the login dialog (as shown in Figure 2-25). The default username and password are shown in Table 2-6, while the parameters for the login dialog are listed in Table 2-7. To change the login password, refer to section 4.1.1.1.2. Please refer to section 2.4 for the two available modes of this software.

Note： If there is no activity within 30 minutes of login, you will be forcefully logged out.

Figure 2-23 View Mode

Figure 2-24 Click login icon

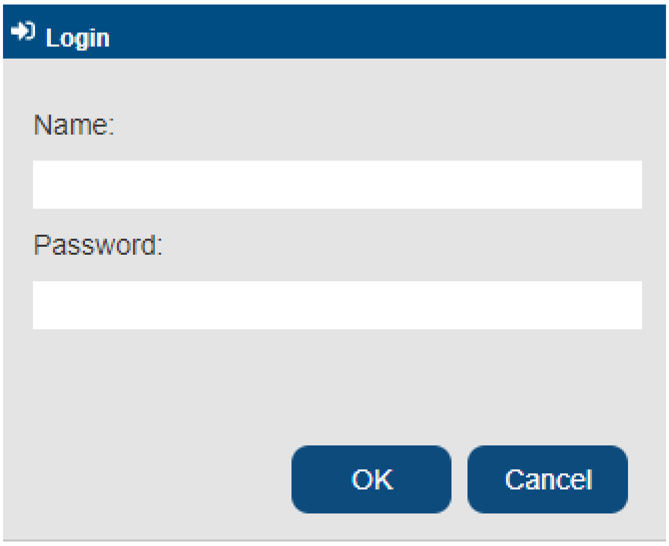


Figure 2-25 Login dialog

Table 2-6 Account name table

Account name	Default password	Premissions
admin	123456	Have administrative privileges.

Table 2-7 Parameter table for login dialog

Parameter name	Content
Name	Input username
Password	Input password

Note：When an administrator login, the software interface will display “logging in” (as shown in Figure 2-26). To force a login, click anywhere on the screen (as shown in Figure 2-27), and a login dialog will appear (as shown in Figure 2-28). If the login is successful, the perviously logged-in administrator will be forced to logout.

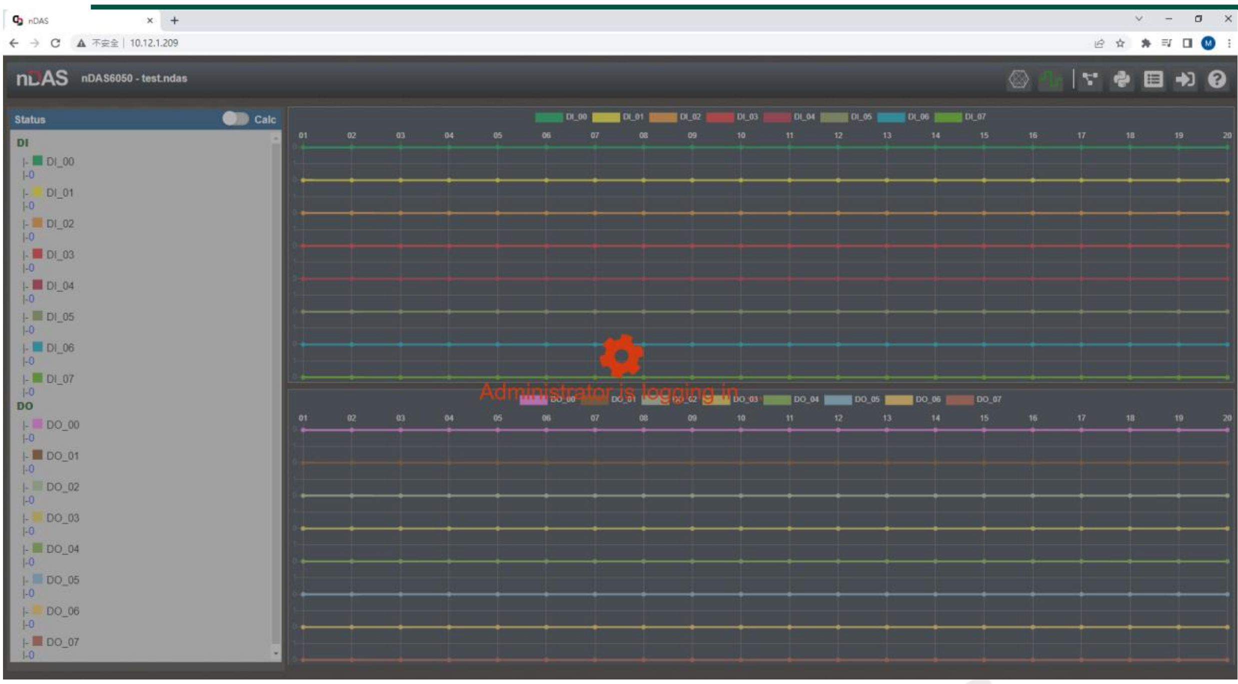


Figure 2-26 Display logging in status

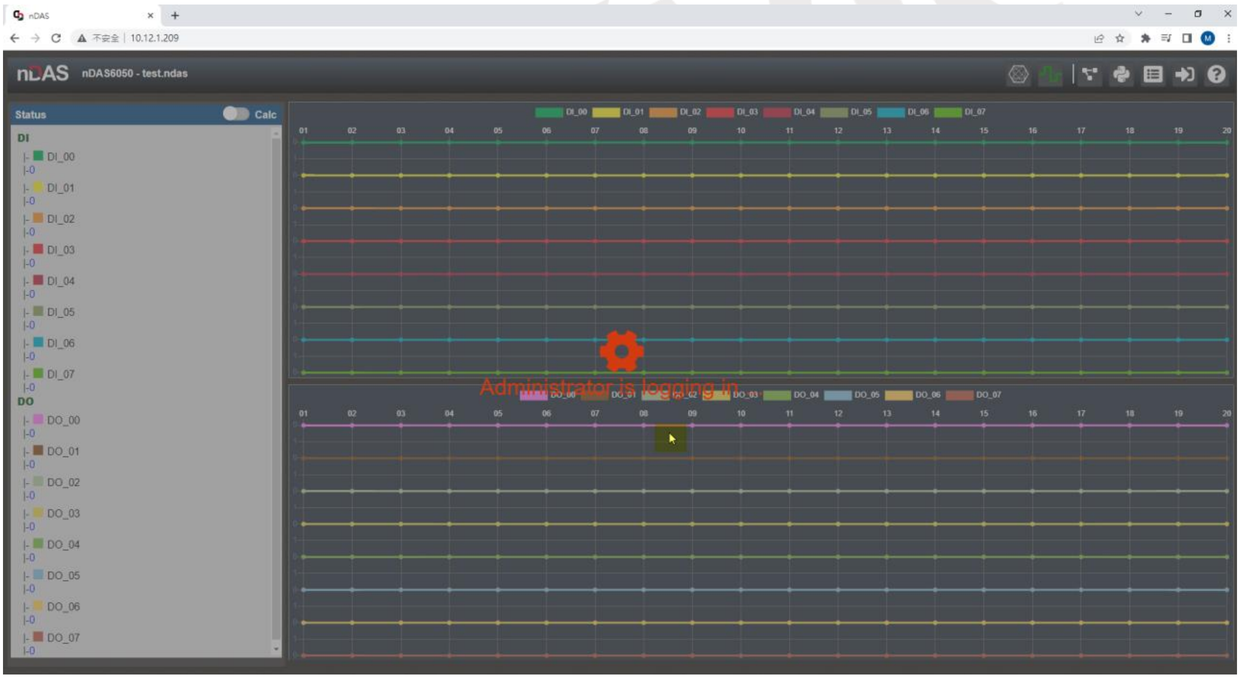


Figure 2-27 Click anywhere on the screen

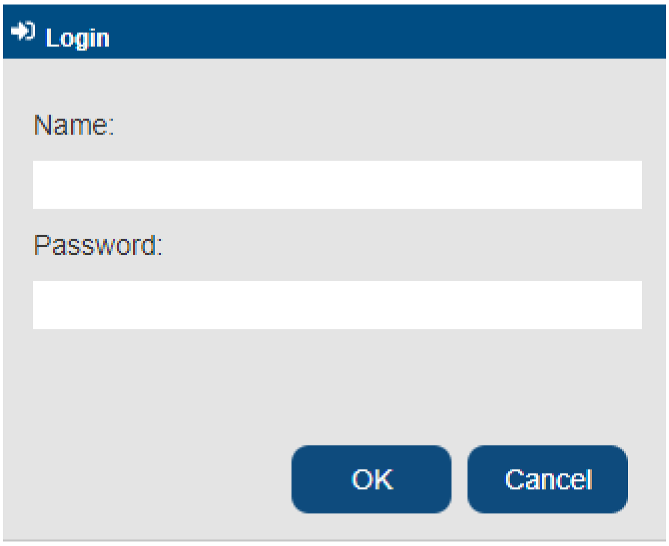


Figure 2-28 Login dialog

Chapter 3. Project Management

3.1 New Project

Click the "New Project" icon (as shown in Figure 3-1) to display the "New Project" settings dialog(as shown in Figure 3-2). After completing the settings, click "OK" to complete the project creation. Refer to Table 3-1 for the parameters of the "New Project" dialog.

Note：The project name is required and must not be duplicated with existing project names.

Figure 3-1 Click New Project icon

Figure 3-2 New project settings dialog

Table 3-1 New project dialog parameter table

Parameter name	Content
Project Name	Input new project name. This field is mandatory and the name must not be duplicated with any existing project names.
Author	Input the author for the new project, which is optional.
Version	Input the version for the new project, which is optional.
Comment	Input the comment for the new project, which is optional.

3.2 Load Porject

Click on the project loading icon (as shown in Figure 3-3) to display the project loading dialog (as shown in Figure 3-4). Select the desired project name and click the “Load” button to complete the project loading process. Click the Delete button to delete the project. Please refer to Table 3-2 for the functionos of the buttons in the project loading dialog.

Figure 3-3 Click Load Project icon

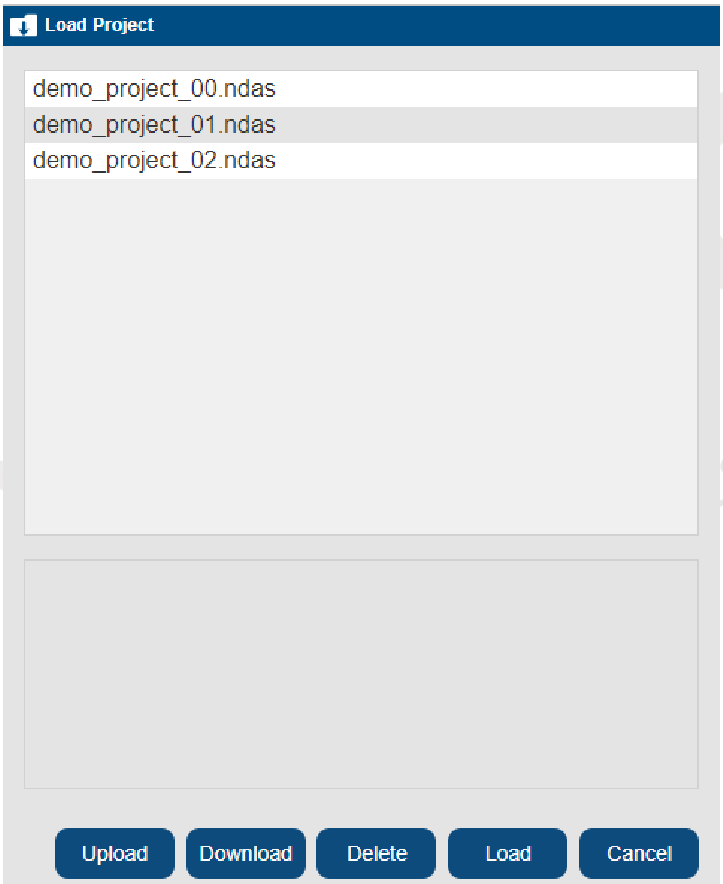


Figure 3-4 project loading dialog

Table 3-2 Load Project dialog button table

Button name	Button function
Upload	Selet “upload to remote device” to upload the local project to the remote device.
Download	Choose the project from the remote device and click “Download” to download it.
Delete	Selete the project name to be delete, click

	the "Delete" button to remove the project.
Load	Choose the project to be loaded, click the "Load" button to load the selected project.
Cancel	Click the "Cancel" button to close the Load Project dialog.

3.3 Save Project

After completing the project creation, clicking on the save project icon (as shown in Figure 3-5) will save the current content of the project.

Figure 3-5 click Save Project icon

3.4 Save As Project

Clicking on the “Save As” icon (as shown in Figure 3-6) will display the “Save As” settings dialog (as shown in Figure 3-7). Once the settings are configured, clicking “OK” will complete the project save. Please refer to Table 3-3 for the parameters of the “Save As” dialog.

Note：The project name is required and must not be duplicated with existing project names.

Figure 3-6 Click Save As Project icon

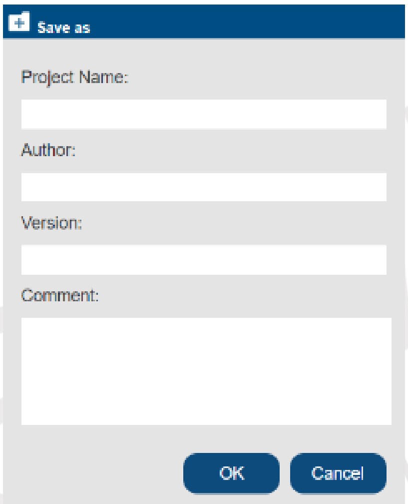


Figure 3-7 Save As Project settings dialog

Table 3-3 Save As Project parameter table

Parameter name	Content
Project Name	Input a name for the new saved project. This field is required and must not be the same as an existing project name.
Author	Input the name for the new saved project, which is optional.
Version	Input the version for the new saved project, which is optional.
Comment	Input the comment for the new saved project, which is optional.

Chapter 4. Configuration of nDAS Functions

4.1 Function Configuration

The basic functionalities of nDAS are set in administrator mode. Clicking on the settings icon (as shown in Figure 4-1), the function displayed on the screen includes system, Modbus, I/O, data logging, cloud upload, OPCUA, SECS/GEM, and other functional settings. The operating methods for these functionalities will be explained in subsequent chapters.

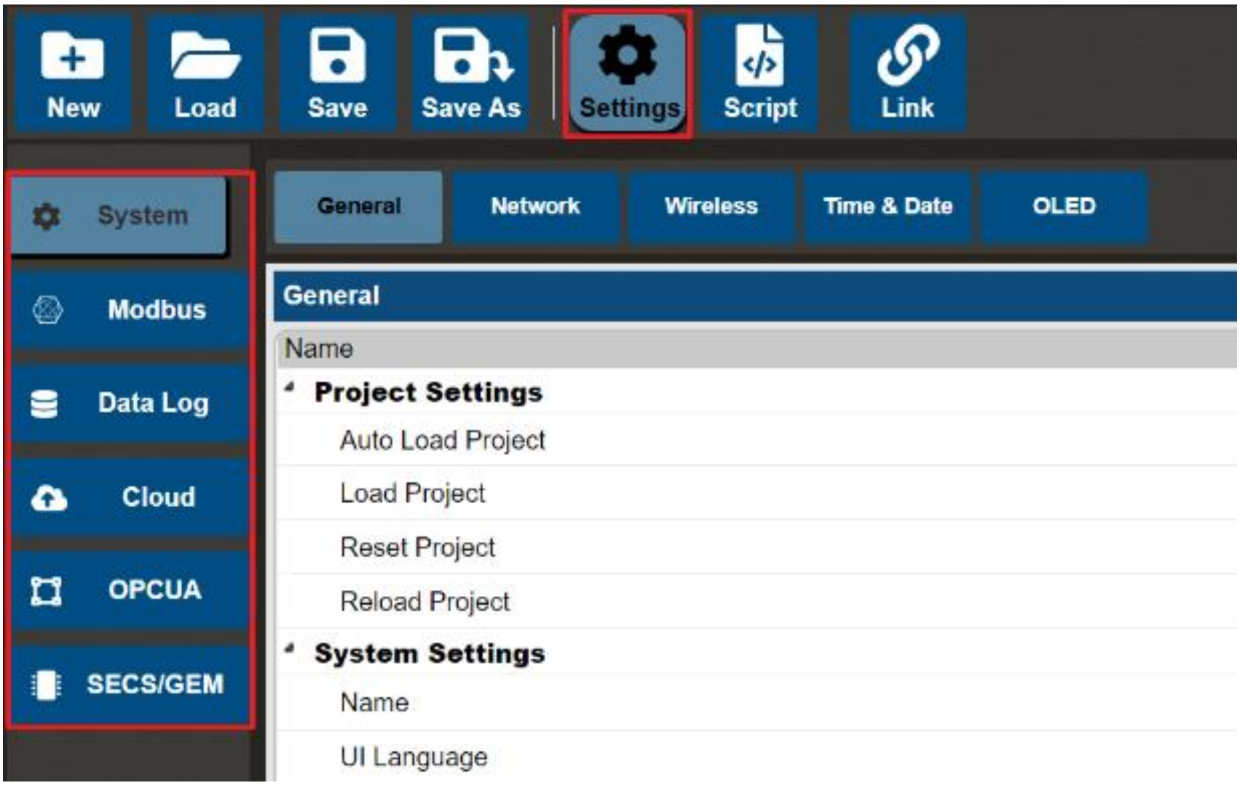


Figure 4-1 Basic Configuration for nDAS

4.1.1 System Settings

Clicking on the icon on the left menu (as show in Figure 4-2) will display the configuration parameters for nDAS' basic system (General), network settings (Network), wireless network (Wireless), time zone (Time & Date) , and OLED.

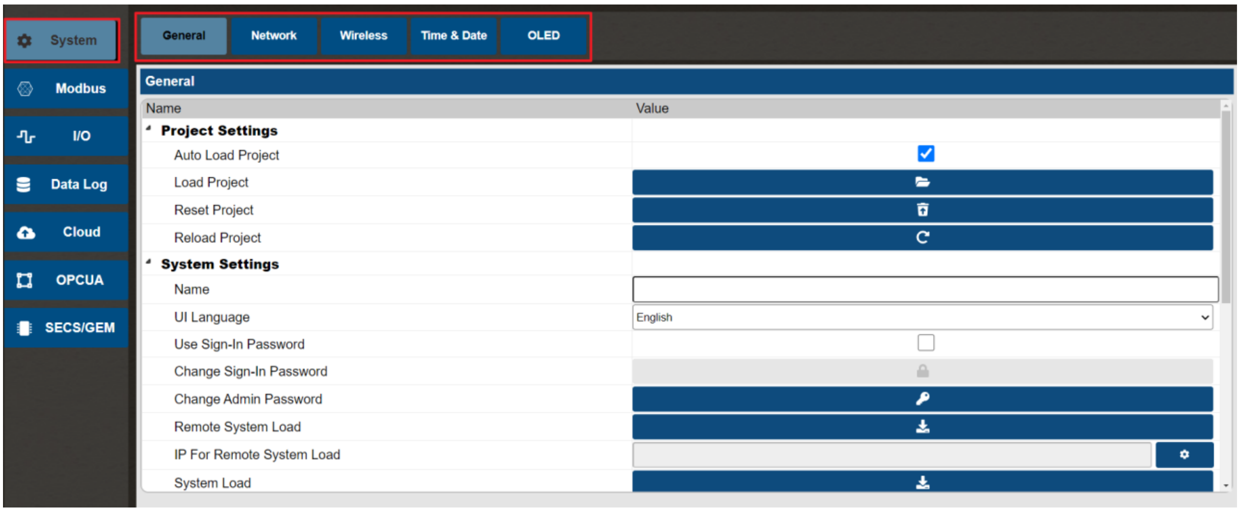


Figure 4-2 Click System Settings icon

4.1.1.1 General

Click on the icon to access the general system settings of nDAS (as show in Figure 4-3). The screen includes parameter settings for projects, systems, flows, network drive, and communication ports. Please refer to Table 4-1, Table 4-2, Table 4-3, Table 4-4, Table 4-5 for guidance.

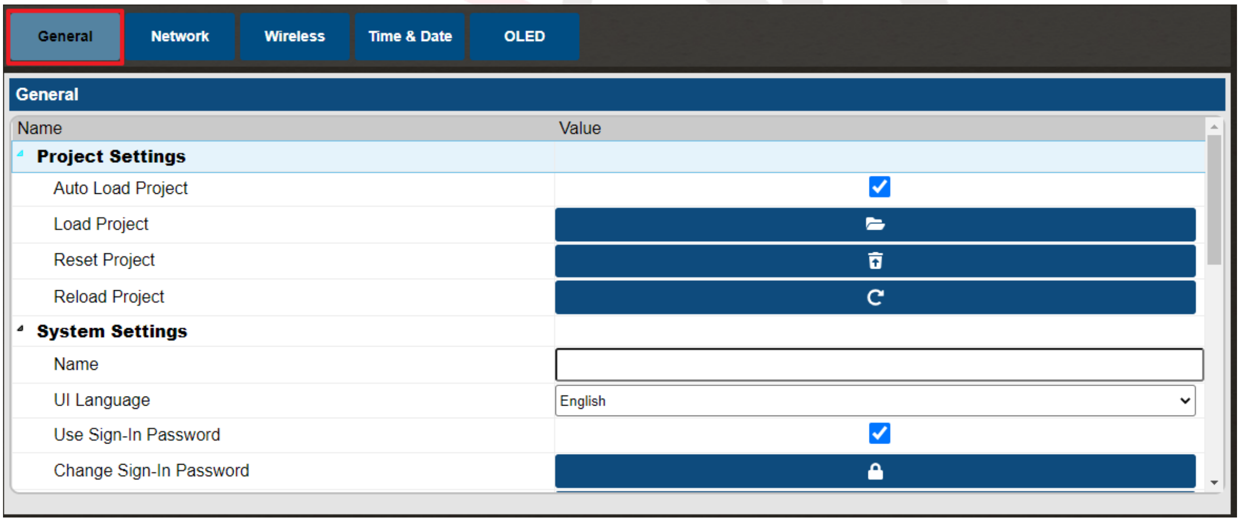


Figure 4-3 General System Configuration for nDAS

Table 4-1 Project Settings

Parameter name	Content
Auto Load Project	If selected, the program will automatically load the previous project upon restart. It is checked by default.
Load Project	Refer to section 3.2 to load a project.
Reset Project	Clicking the button will reset the project.

Reload Project	Clicking the button will reload the current project.

Table 4-2 System Settings

Parameter name	Content
Name	Set the name of the nDAS device.
UI Language	Select the interface language from the drop-down menu. The available languages are English, Traditional Chinese, Simplified Chinese, and Japanese. Please refer to section 4.1.1.1.1.
Use Sign-In Password	Enable/disable sign-in login password.
Change Sign-In Password	Change the sign-in login password.
Change Admin Password	Clicking on this option will display the "Change Admin Password" dialog. Please refer to section 4.1.1.1.2.
System Load	Clicking on this option will display the “System Load” dialog. Please refer to section 4.1.1.1.8.
System Save	Clicking on this option will display the “System Save” dialog. Please refer to section 4.1.1.1.9.
System Restart	Reboot nDAS
System Updata	Update nDAS
Simulator	Activate/Deactivate the I/O simulator.
Firmware Version	Show firmware version
Script Watchdog Timeout(ms)	Set the maximum execution time for the script. If the execution time exceeds this limit, the script will be forcefully terminated. Note that this parameter is only valid for Python scripts.

Table 4-3 Flow Setting

Parameter name	Content
Delay Time(ms) For Running Descending Operators	Input a value and set the delay time to view the display function of the flow.

Table 4-4 Network Drive

Parameter name	Content
Enable	Enable / disable network drive functionality
Anonymous	Enable / disable network drive anonymous mode
Name	Display name of network drive
Password	Enter network drive password (after entering, click the save button to apply)

Table 4-5 Communication Port

Parameter name	Content
UDP Port	After clicking, the UDP port configuration window will appear.
UDP Multicast Group Port	After clicking, the UDP multicast port configuration window will appear.
TCP Command Port	After clicking, the TCP command port configuration window will appear.
UDP Multicast Group Address	After clicking, the UDP multicast address configuration window will appear.

4.1.1.1.1 UI Language

This software supports four different interface languages, which are English, Traditional Chinese, Simplified Chinese, Japanese, and Korean. The language can be switched through the system settings by selecting the language from a drop-down menu, as shown in the red box in Figure 4-4.

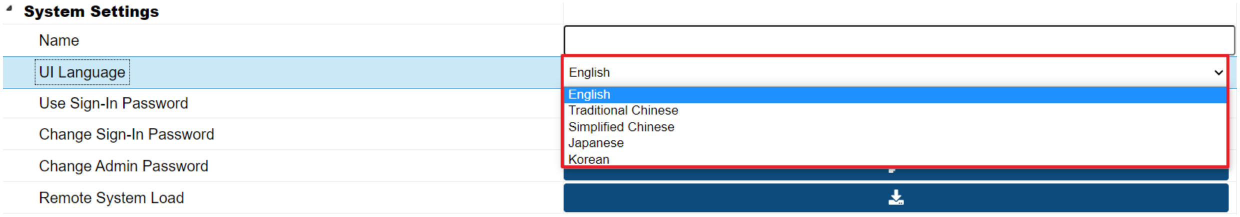


Figure 4-4 System Settings – UI Language

4.1.1.1.2 Sign-In Password

Please select the option for login password in the system settings (as shown in Figure 4-5), which will prompt for confirmation to reload the webpage and log in (as shown in Figure 4-6). After clicking the confirmation, users will be prompted to log in (as shown in Figure 4-7). Please refer to Table 4-6 for the login name and password, and once the authentication is successful, the webpage will be reloaded.

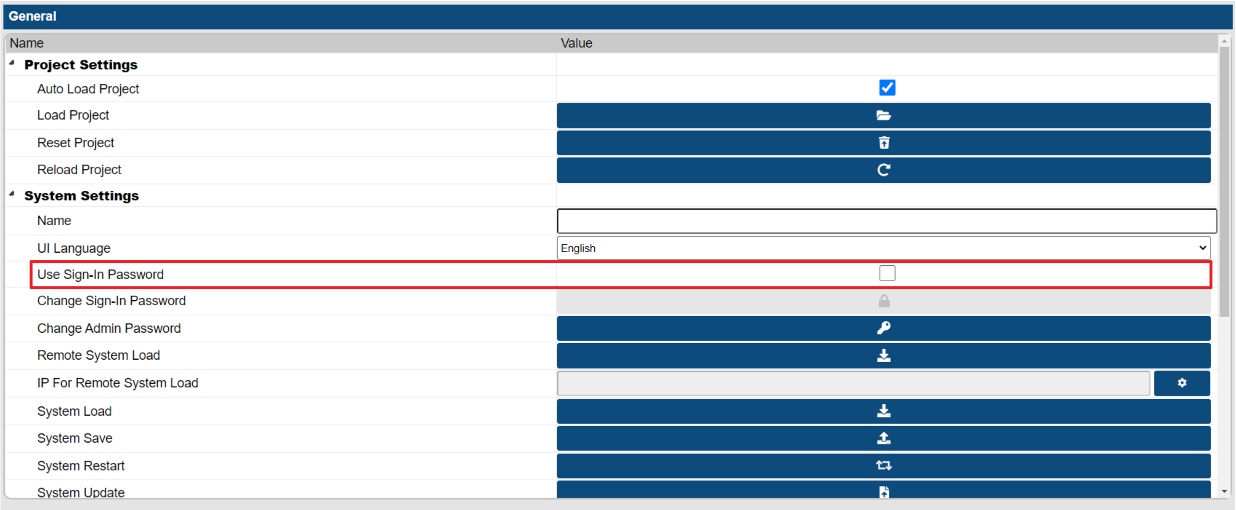


Figure 4-5 Click to start the webpage login password

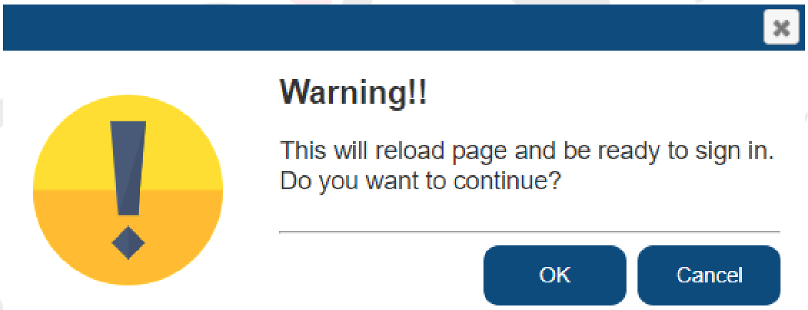


Figure 4-6 Confirm whether to reload the page and log in

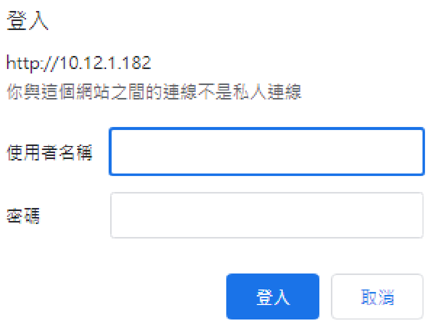


Figure 4-7 login page and reconnect

Table 4-6 RESTful API Basic Authenticationauthentication parameters

Parameter name	Content

Username	admin
Password	123456(default)

4.1.1.1.3 Change Sign-In Password

Click on the "Change Webpage Login Password" button in the system settings (as shown in Figure 4-8), which will display a window for changing the webpage login password (as shown in Figure 4-9). Once the settings are complete, click "OK" to finalize the password change, and prompt the user to log in using the updated username and password. After successful login, the webpage will be reloaded. Please refer to Table 4-7 for the parameters.

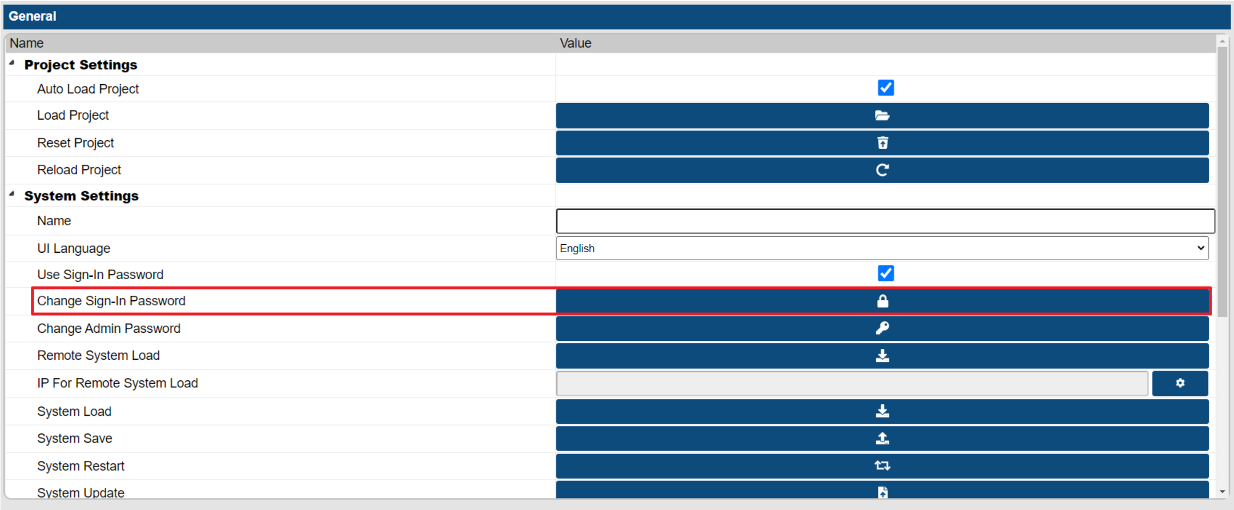


Figure 4-8 Change web login password

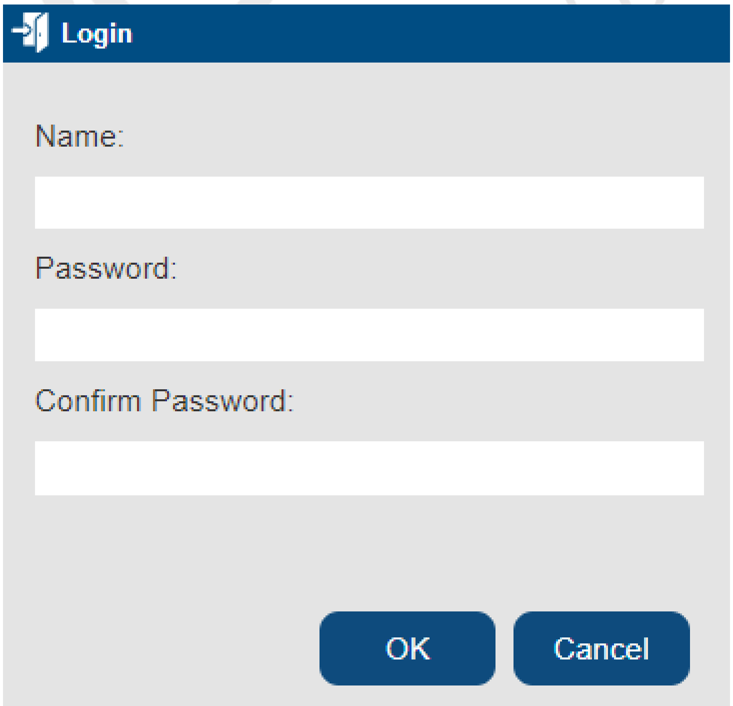


Figure 4-9 Change web page login password window

Table 4-7 Change web page login password parameter table

Parameter name	Content
Name	Enter a name.
Password	Enter a password.
Confirm Password	Confirm password

4.1.1.1.4 Change Password

By clicking the "Change Password" button in the system settings (as shown in Figure 4-10), a dialog for changing the password in the administrator mode will be displayed (as shown in Figure 4-11). After completing the settings, click OK to complete the password change. For more details about the parameters, please refer to Table 4-8.

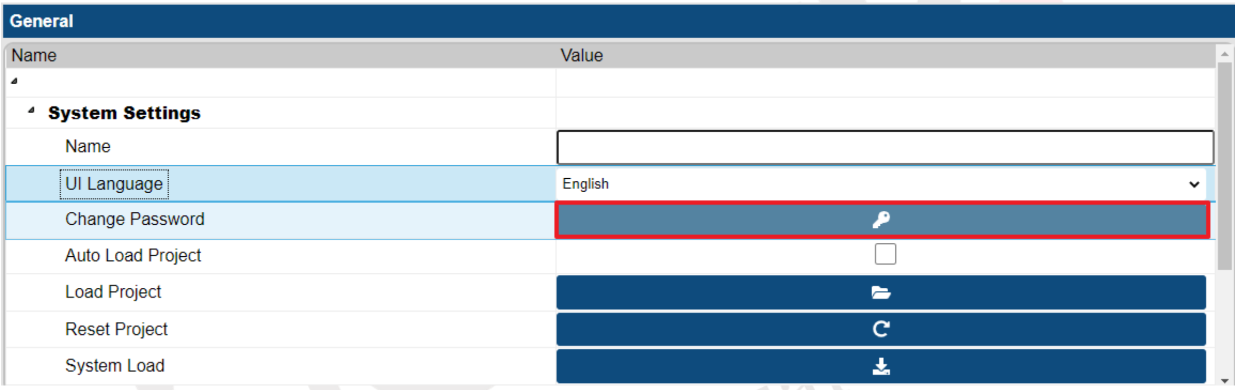


Figure 4-10 Click "Change Password" button

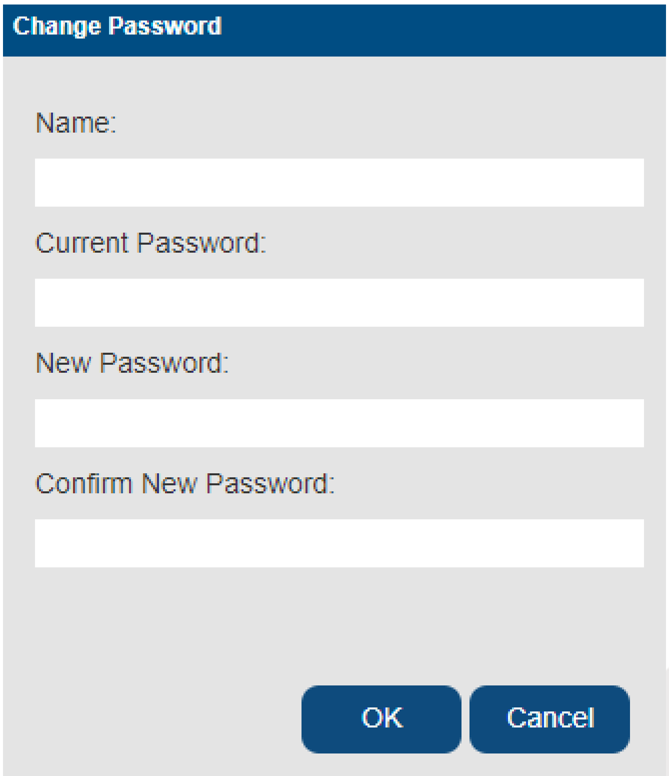


Figure 4-11 Change Login Password dialog

Table 4-8 Change the password parameter table

Parameter name	Content
Name	Input account name.
Current Password	Inputcurrent password.
New Password	Input desired new password.
Confirm New Password	Re-enter your desired new password.

4.1.1.1.5 Load Project

Click on the load project button in the system settings (as shown in Figure 4-12), please refer to section 3.2 for the functionality.

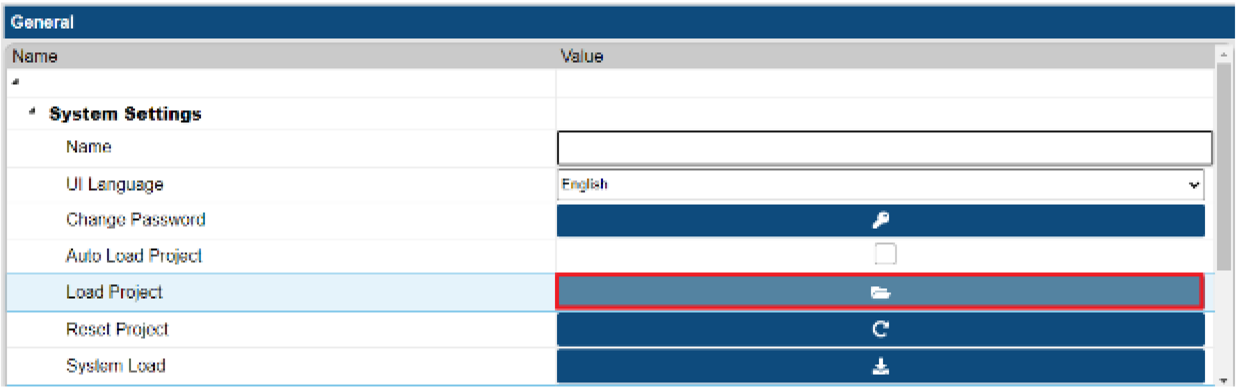


Figure 4-12 Click on “Load Project” in the system settings

4.1.1.1.6 Rest Project

Upon clicking the button and confirming, the parameter settings of the current project will be reset to initial status.

4.1.1.1.7 Reload Project

Click the button to reload the current project.

4.1.1.1.8 System Load

The software provides a system backup function. Click the "System Load" button in the system settings (as shown in Figure 4-13) to display the system load dialog (as shown in Figure 4-14). Click "OK" to confirm loading the selected file, or "Delete" to delete the file. Additionally, the software supports loading local backup files. Click the "Local File" button in the lower left corner of the system load dialog (as shown in Figure 4-15) to load the local system backup file into nDAS.

Note: If there are files with the same name in the loaded system backup file as those in the current system, they will be overwritten.

Note: To clear the current system's projects and system configurations when loading a system backup file, select the "Clear old files" option. The login password will also be changed to the one from the loaded system backup.

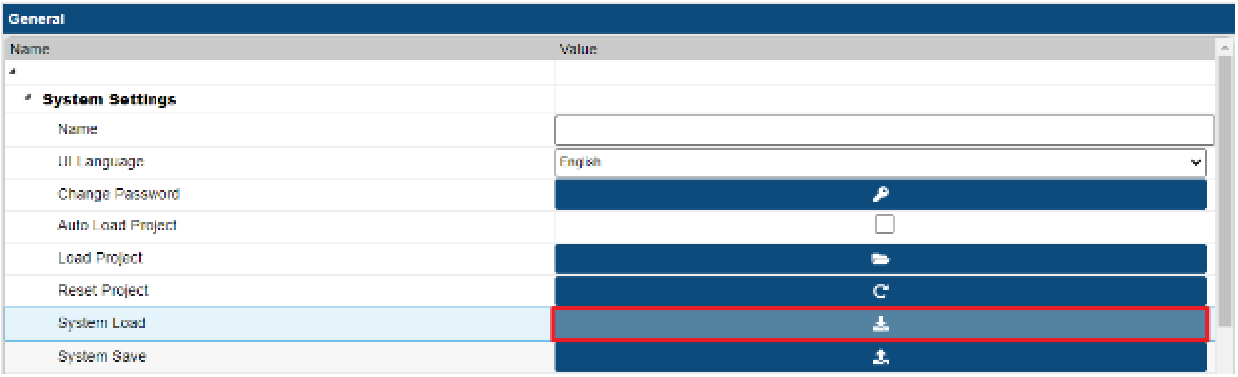


Figure 4-13 Click on “System Load” in the system settings dialog

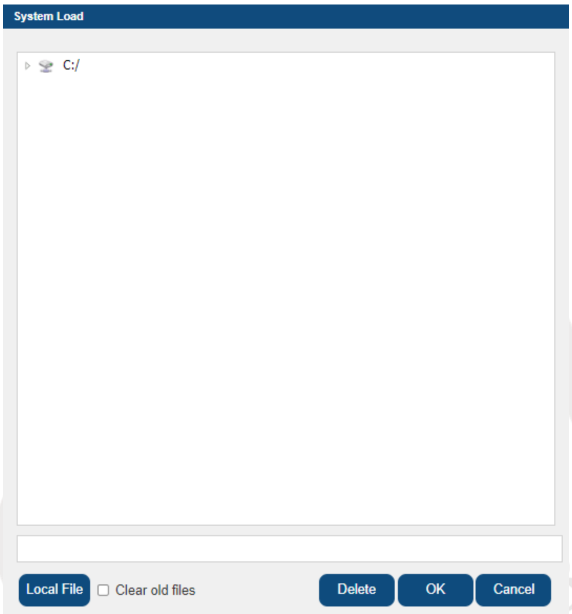


Figure 4-14 System Load dialog

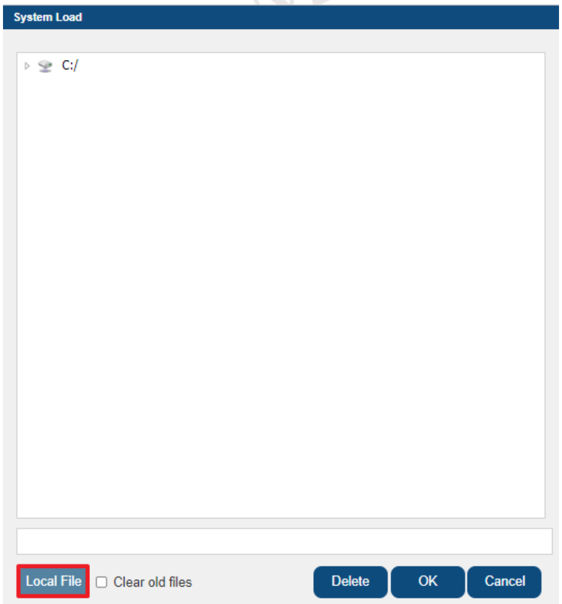


Figure 4-15 System Load dialog (Local File)

4.1.1.1.9 System Save

The software provides a system backup function. By clicking on the "System Save" button on the system settings (as shown in Figure 4-16), you can select the items to be saved, including system settings, project, py files, and OPCUA security file (as shown in Figure 4-17). After confirming the selected items, clikc OK to display the system save dialog (as shown in Figure 4-18).

Once you have confirmed the storage path, enter the desired file name and click OK to complete the creation of the system backup file. The location of the stored file will be on the remote end (nDAS device). The software also supports local save, where you can click on the "Local File" button in the lower-left corner of the system save dialog (as shown in Figure 4-19) to store the system backup file locally.

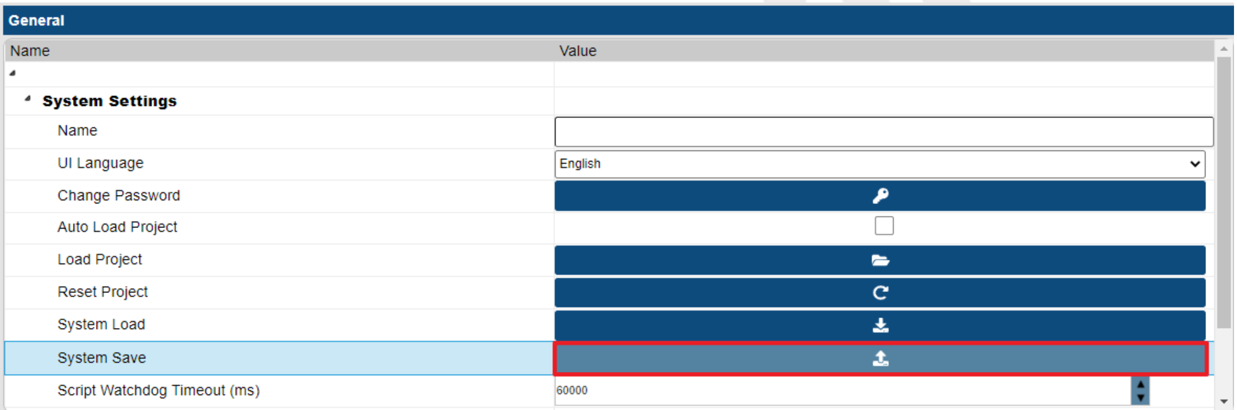


Figure 4-16 Click on the "System Save" button in the system settings dialog

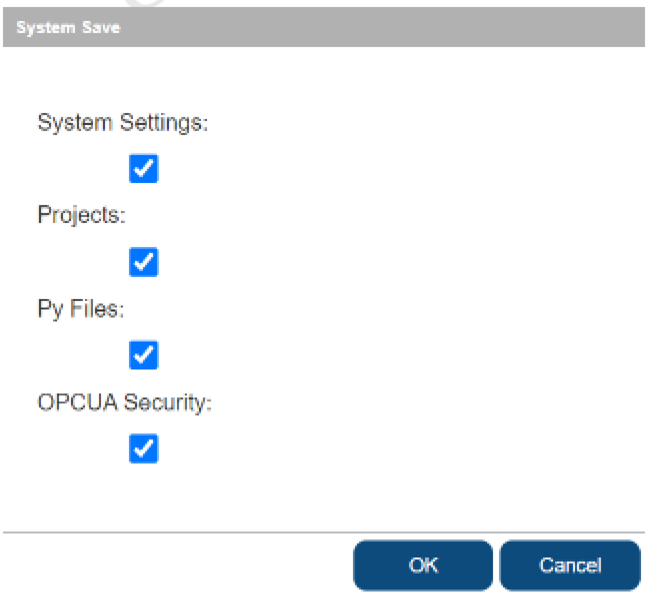


Figure 4-17 System save items

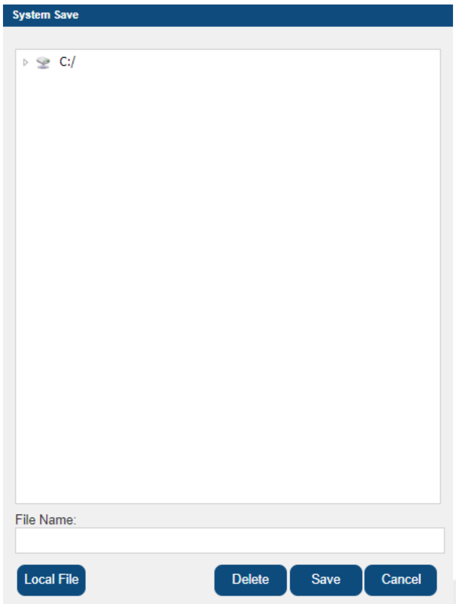


Figure 4-18 System save dialog

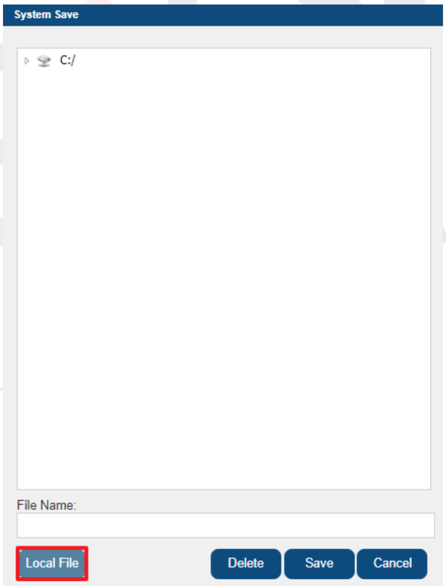


Figure 4-19 System save dialog (Local File)

4.1.1.2 Network

Click on the icon will take you to the nDAS network settings (as shown in Figure 4-20), which includes communication settings for the Ethernet network interface and the Bridge network interface.

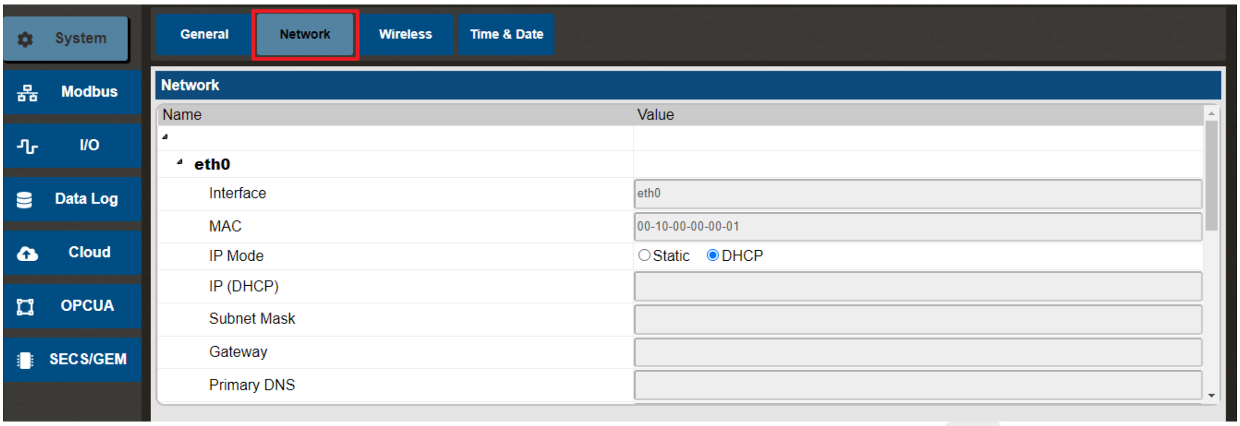


Figure 4-20 nDAS network configuration

4.1.1.2.1 eth Interface Setting

Table 4-9 contains the parameter settings for the Ethernet network interface.

Table 4-9 eth interface parameter settings

Parameter Name		Parameter Content
eth Network Interface	Interface	Display network interface name
	MAC	Display MAC address
	IP Mode	Switch to Static/DHCP
	IP	IP address (if in Static Mode, you can enter it)
	Subnet Mask	Subnet mask (if in Static Mode, you can enter it)
	Geteway	Default gateway (if in Static Mode, you can enter it)
	Primary DNS	Primary DNS server (you can enter it if in Static Mode)
	Secondary DNS	Secondary DNS server (you can enter it if in Static Mode)
	Status	Display network packet receive/transmit traffic
	Apply	Click the button to save parameter changes

4.1.1.2.2 Bridge Interface Setting

Table 4-10 contains the parameter settings for the Bridge interface, which is used to connect multiple nDAS devices in a Daisy Chain topology using physical network cables.

Please be advised that before clicking on "Apply" and subsequently "Enable," it is essential to input the IP and Subnet Mask addresses. Only then will the Bridge interface connection take effect.

Table 4-10 Bridge interface parameter settings

Parameter Name		Parameter Content
Bridge Enable	Enable	Enable/Disable Bridge
Bridge Setting	Interface	Display network interface name
	IP	IP address (you can enter it if in Static Mode)
	Subnet Mask	Subnet mask (you can enter it if in Static Mode)
	STP	Enable/Disable Bridge STP mode
	Status	Display network packet receive/transmit traffic
	Apply	Click the button to save parameter changes

4.1.1.3 Wireless

Click on the icon to access the wireless network settings (as shown in Figure 4-21), where you will find communication settings for the wireless network. Table 4-11 is the parameter settings for the wireless network.

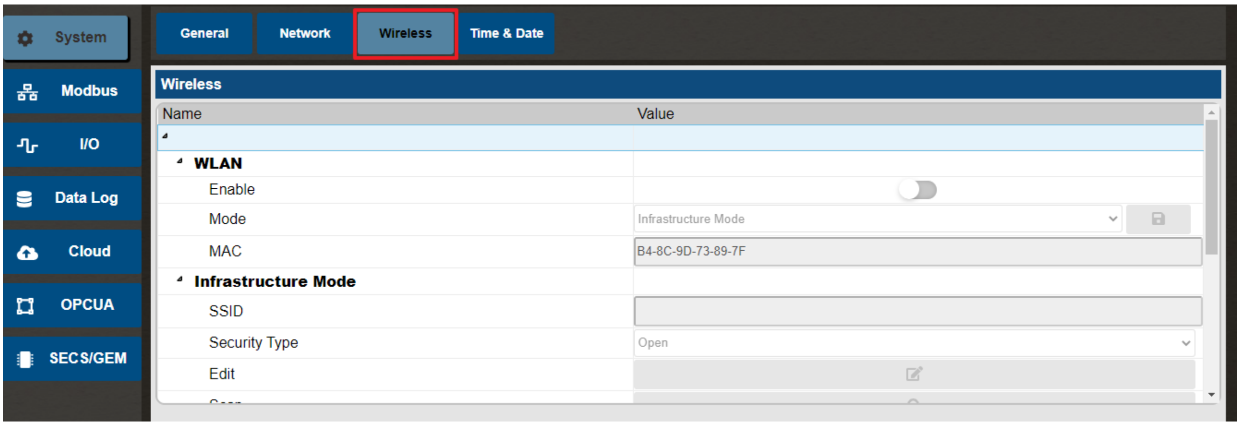


Figure 4-21 nDAS wireless network settings

Table 4-11 Parameter settings for the wireless network

Parameter name		Content
WLAN	Enable	Enable / Disable wireless network function.
	Mode	AP / Infrastructure Mode
	MAC	Display MAC address.
Infrastructure Mode	SSID	Edit SSID
	SSID Hidden	Hide SSID [Enable/Disable]
	Used Channel	Channel
	Security Type	Security Technology
	Apply	Click the button to save parameter changes.
Infrastructure Mode IP Setting	IP Mode	Fixed IP Mode
	IP	Fixed IP: 192.168.0.1
	Subnet Mask	Fixed Subnet Mask: 255.255.255.0
	Gateway	Fixed Default Gateway: 192.168.0.1
Centralized Control Mode	SSID	Display SSID.
	Security Type	Display security type.
	Edit	Edit wireless network parameters, please refer to the figure _.
	Scan	Click the button to display the list of currently available WiFi networks.
Centralized Control IP Configuration	IP Mode	Switch to Static / DHCP mode.
	IP	IP address (can be entered if in Static Mode.)
	Subnet Mask	Subnet mask (can be entered if in Static Mode.)
	Geteway	Default gateway (can be entered if in Static Mode.)
	Primary DNS	Primary DNS server (can be entered if in Static Mode.)
	Secondary DNS	Secondary DNS server (can be entered if in Static Mode.)

	Status	Display network packet receive / transmit traffic.
	Apply	Click button to save parameters change.

4.1.1.3.1 AP Mode

After switching the mode to AP Mode, as depicted in Figure 4-22, input the SSID and click on the Save button. Subsequently, nDAS will transition to AP Mode.

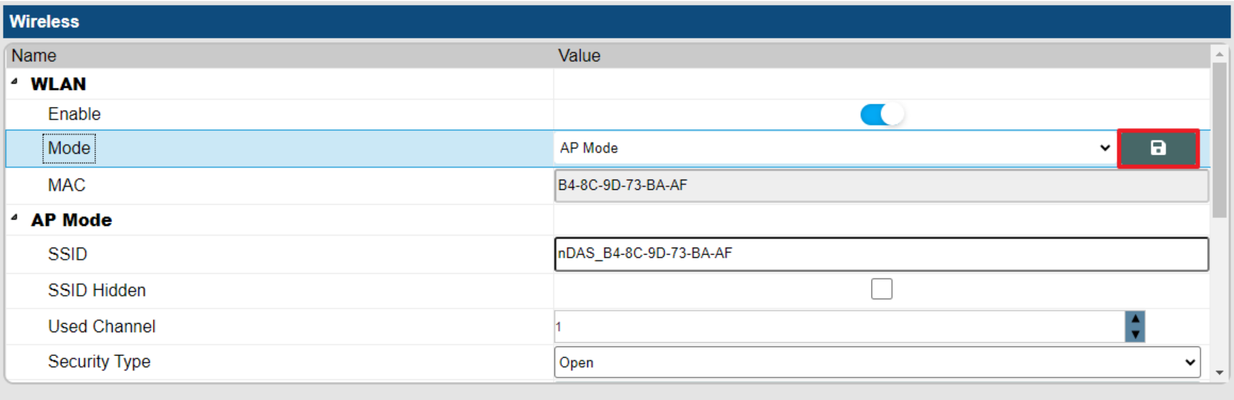


Figure 4-22 AP Mode Setting

4.1.1.3.2 Infrastructure Mode

Upon switching the mode to Infrastructure Mode, click on the Edit button for Infrastructure Mode (as shown in Figure 4-23). This action will display the editing dialog window (as depicted in Figure 4-24). Input the SSID and select the desired security protocol. Finally, click on the Confirm button to complete the configuration.

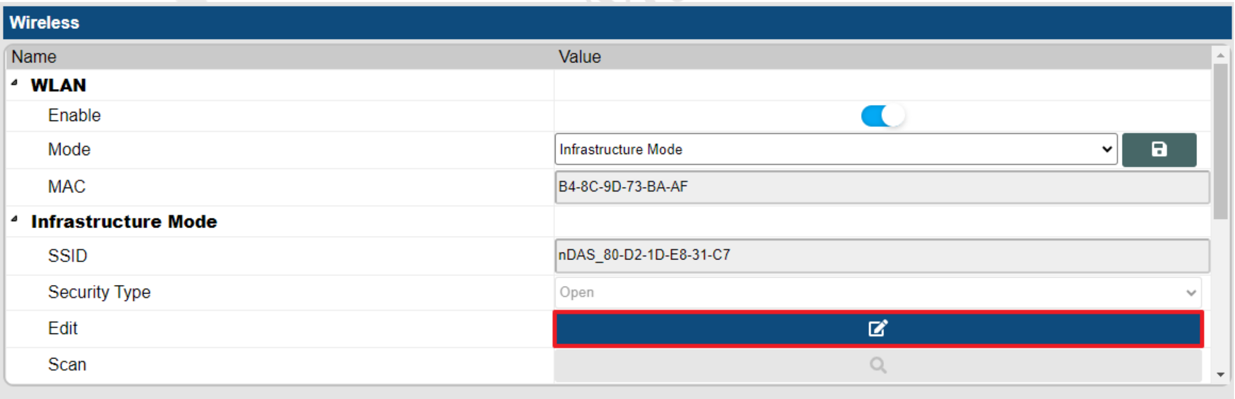


Figure 4-23 Infrastructure Mode

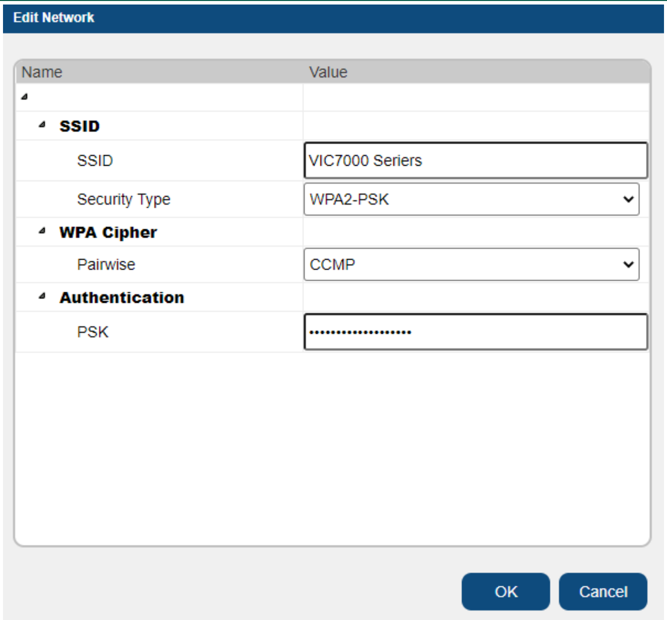


Figure 4-24 Infrastructure Mode Connection Setting

After clicking the Save button (as shown in Figure 4-25), wait for the connection and enter the IP of that domain in the web browser to reopen the settings page. The current wireless signal strength will be displayed in the toolbar at the top (as shown in Figure 4-26).

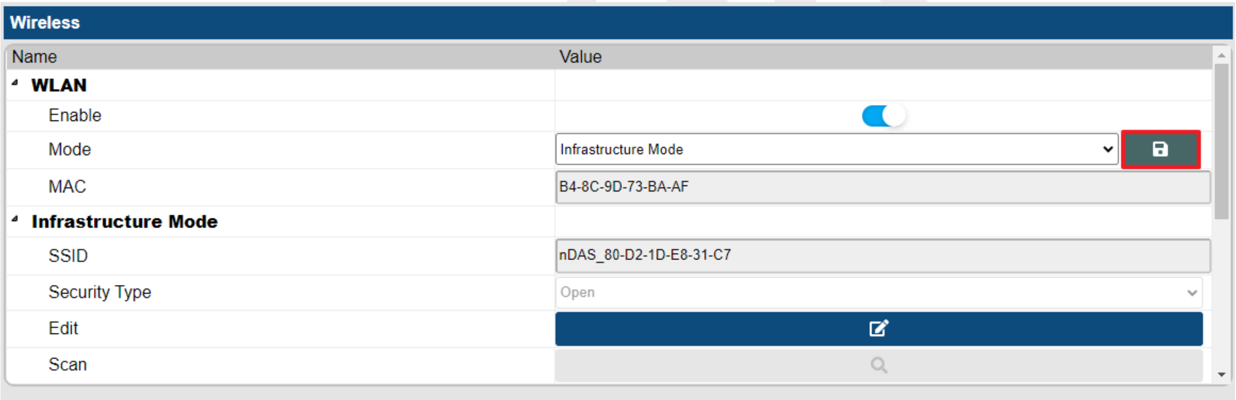


Figure 4-25 Enable Infrastructure Mode

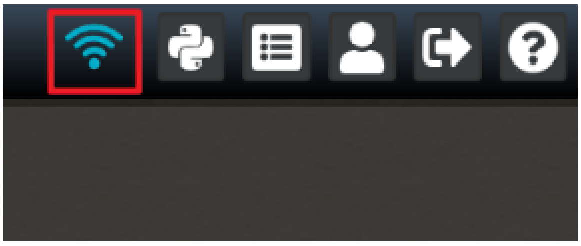


Figure 4-26 the icon of Wireless signal status

4.1.1.4 Time & Date

Click on the icon to enter the time and date settings (as shown in Figure 4-27), and Table 4-12 is the parameter settings for time and date.

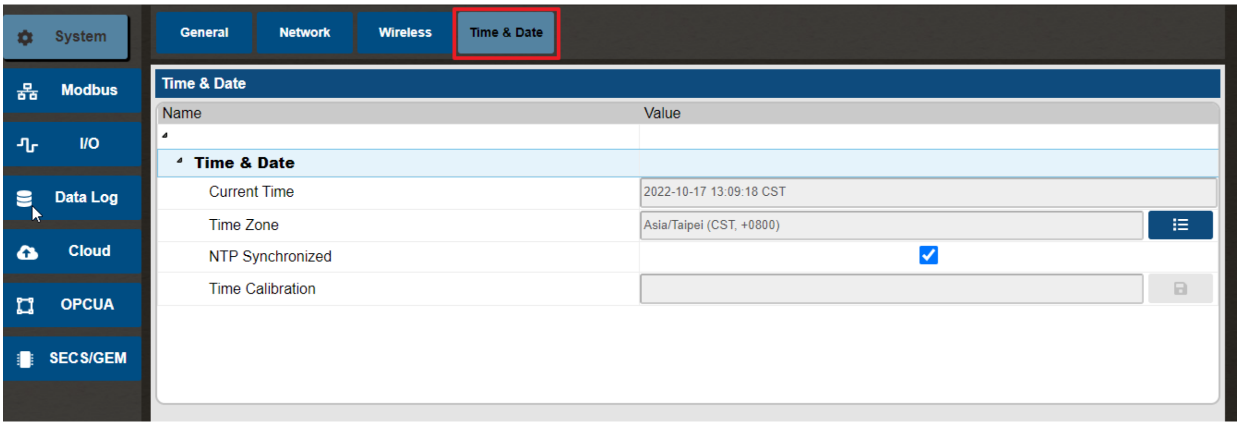


Figure 4-27 Time and date settings

Table 4-12 Parameter settings for time and date

Parameter name		Content
Time & Date	Current Time	Display the current time.
	Time Zone	Click the button to select and set the time zone.
	NTP Synchronized	Check the box to automatically execute network time NTP synchronization.
	Time Calibration	Uncheck NTP Synchronized to manually adjust and correct the time.

4.1.1.5 OLED

When you click on the icon, it will take you to the custom OLED function settings (as shown in Figure 4-28). Table 4-13 contains the parameter settings for the custom OLED.

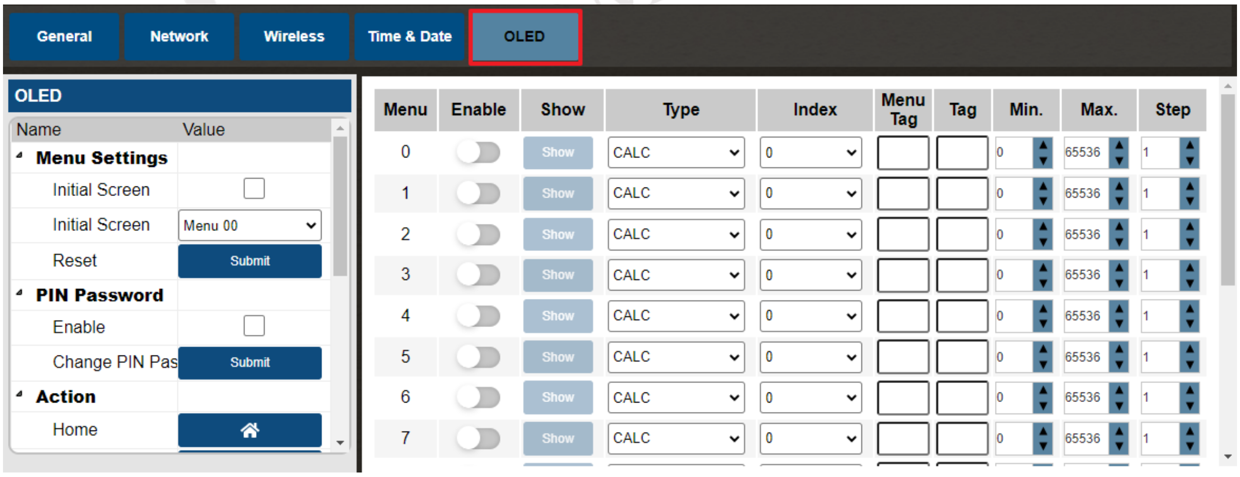


Figure 4-28 Custom OLED Function Settings

Table 4-13 Custom OLED Function Parameter Settings

Parameter name		Content
Menu Settings	Initial Screen(Enable)	Enable / Disable Initial Display Screen

		Functionality (Default: Main Screen)
	Initial Screen(Selection)	Set Custom Initial Display Screen
	Reset	Reset Custom OLED Screen
PIN Password	Enable	Enable / Disable PIN Password Functionality
	Change PIN Password	Set PIN Password
Actions	Home	Return to Main Menu Screen
	Back	Return to Previous Screen
	Knob Right	Trigger Right Rotation Button
	Knob Left	Trigger Left Rotation Button
Menu Actions	Menu1	Enter Main Screen 1
	Menu2	Enter Main Screen 2
	Menu3	Enter Main Screen 3
	Menu4	Enter Main Screen 4
Simulation Display	On / Off	Enable Auto-Switching of Custom OLED Screen Functionality
	Interval (ms)	Set Time Interval for Custom OLED Screen Switching

The custom OLED screen settings interface, as shown in Figure 4-29, includes the following parameters:

- Type: Allows selection of the display screen type, such as CALC operator, Image, Button, I/O Module, Modbus Bit channel, Modbus Word channel.
- Index: Represents the index corresponding to the selected Type.
- Menu Tag: Specifies the name displayed for the menu on the main screen.
- Tag: Specifies the name displayed after entering the menu screen.
- Min / Max: Indicates the minimum and maximum values for writing a numerical input.
- Step: Represents the basic unit for each written value.

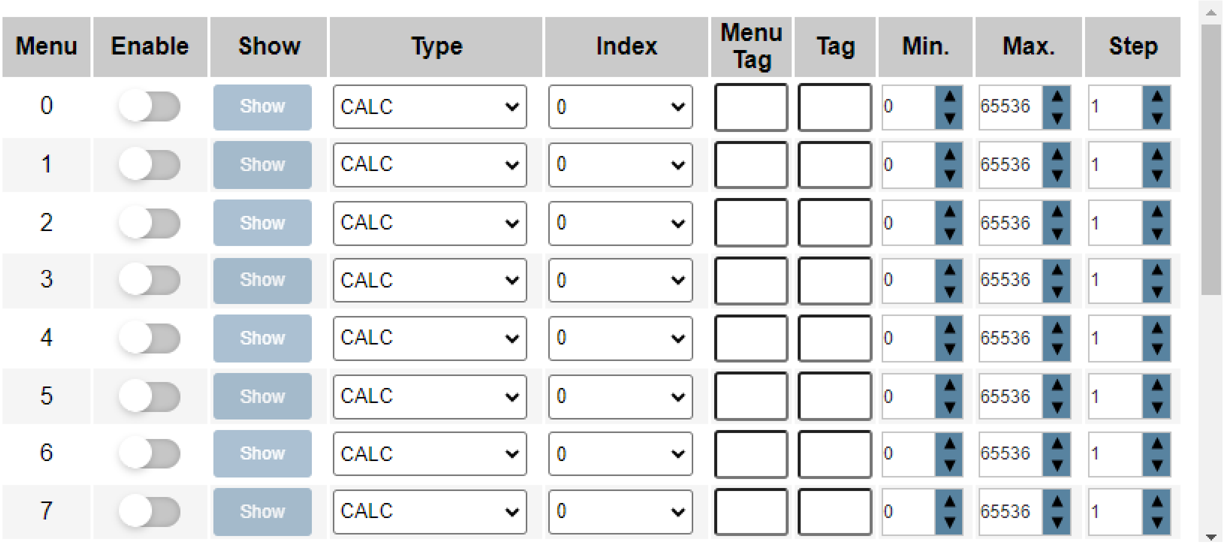


Figure 4-29 Custom OLED Display Settings

Note: If you select the "Image" option for the Type, you will need to click the "View" button to load the image. The supported image formats are *.bmp and *.tif.

4.1.2 Modbus

By clicking on the icon in the left-hand menu (as shown in Figure 4-30) you can enter the Modbus configuration screen. nDAS provides a Modbus TCP/RTU communication interface for users, which includes Modbus TCP/RTU communication parameter settings, device coil status, device holding register, Master COM Port monitoring, and Master TCP Port monitoring (as shown in Figure 4-31)。

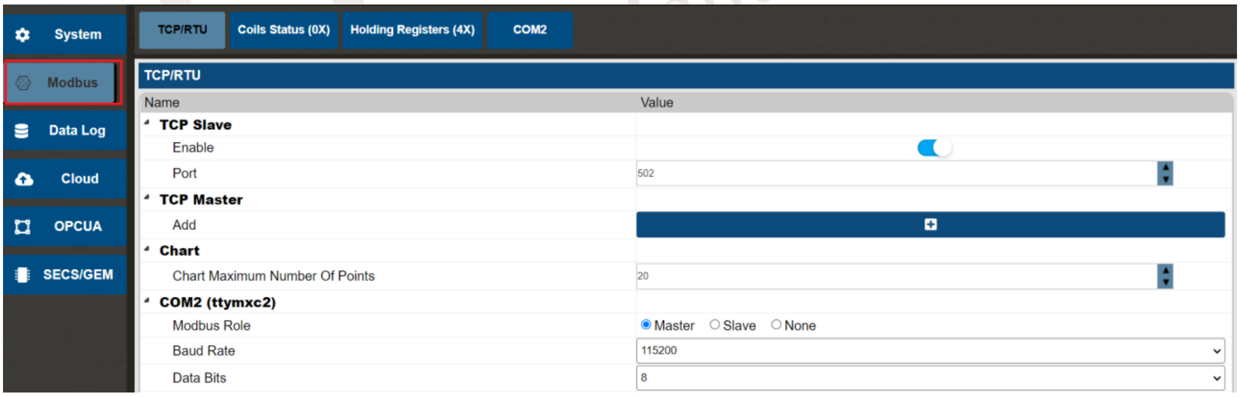


Figure 4-30 Modbus Function Settings

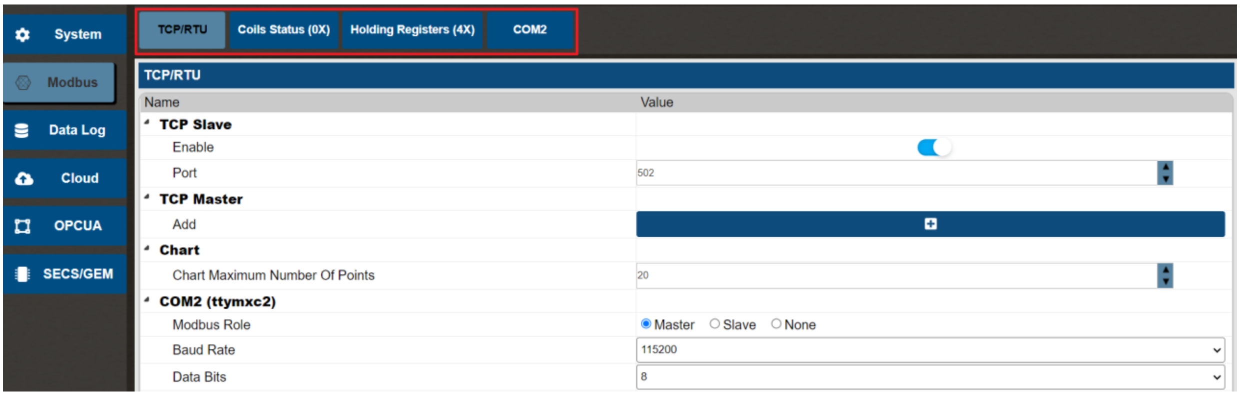


Figure 4-31 Modbus TCP/RTU Communication Parameter Settings, Local Coil Status List, Local Holding Register List, Master COM Port Monitoring, Master TCP Port Monitoring

Note：Monitoring the Master TCP Port requires prior addition in the communication parameter settings, nDAS provides a maximum of five communication ports for user utilization, as referenced in section 4.1.2.1.

4.1.2.1 TCP/RTU

Click on the icon to enter Modbus TCP/RTU Communication Parameter Settings (as shown in Figure 4-32).

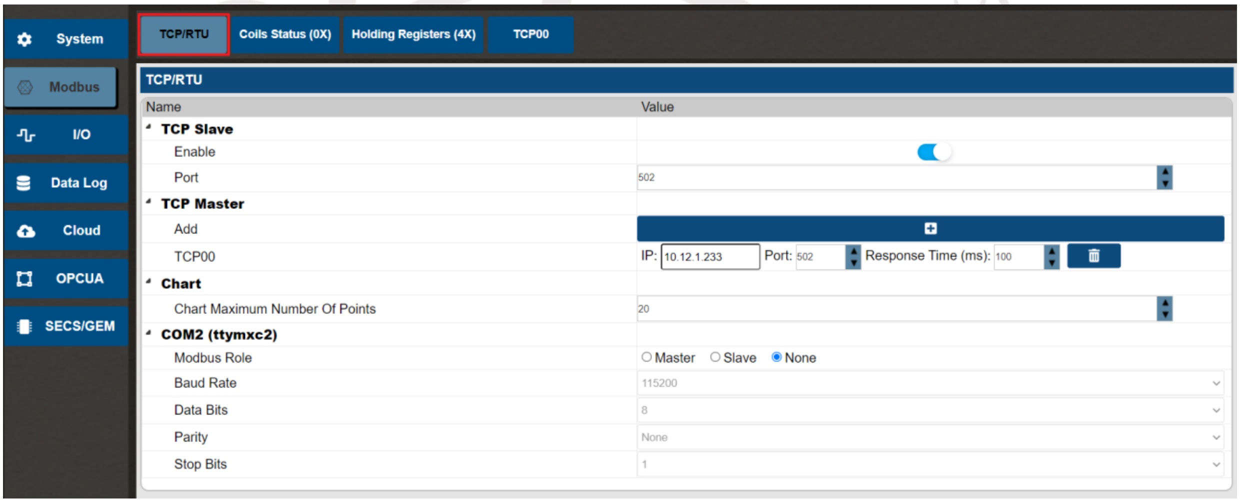


Figure 4-32 Modbus TCP/RTU communication parameter settings

Modbus TCP Slave configuration, as shown in Table 4-14.

Table 4-14 Modbus TCP Slave communication parameter settings.

Parameter name	Content
Enable	Enable/Disable Modbus TCP Slave communication.

Port	Modbus TCP Port configuration.

nDAS supports Modbus TCP Master. Click the "Add" button as shown in Figure 4-33. After clicking the button, you need to enter the IP address, port number, and response time. Additionally, a TCP Master monitoring icon will be added to the menu bar as shown in Figure 4-34.

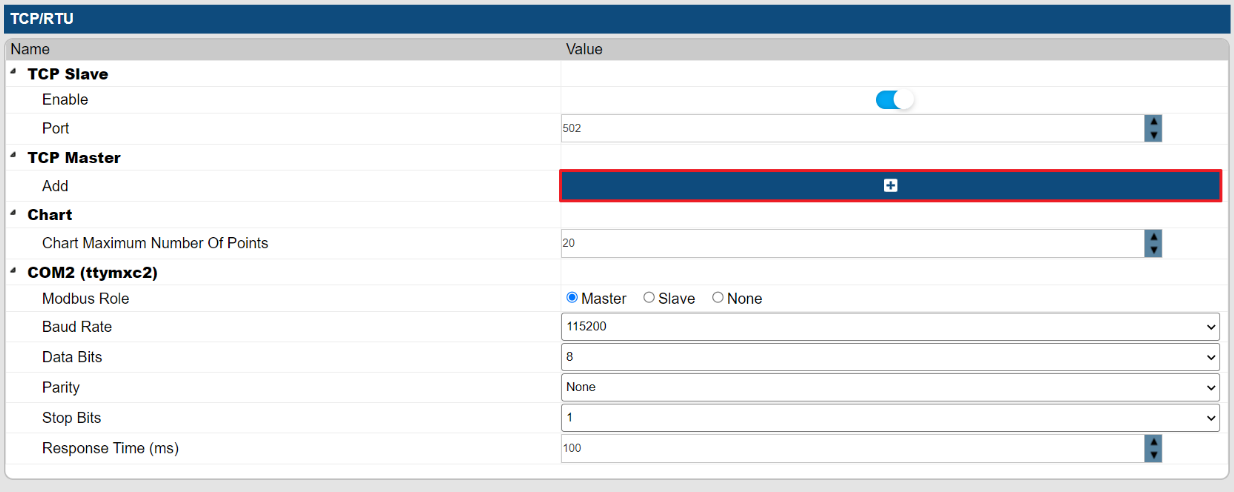


Figure 4-33 Add Modbus TCP Master communication port

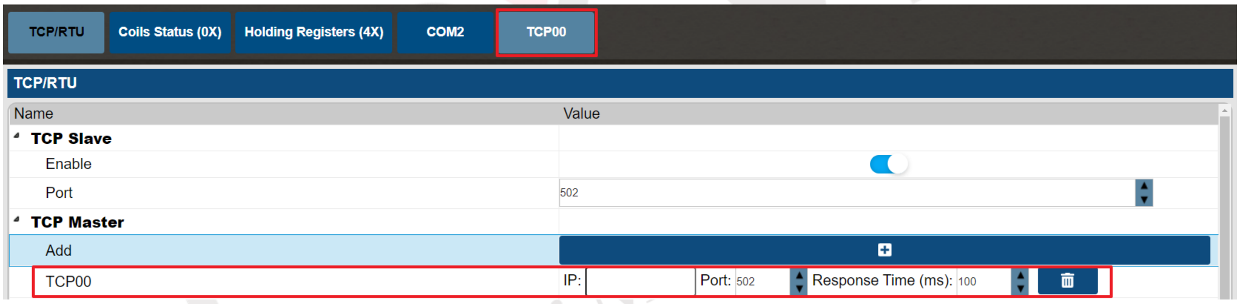


Figure 4-34 Modbus TCP Master monitoring menu

Configure the quantity of Modbus Master drawing points (as shown in Figure 4-35). Enter the maximum number of drawing points, referring to section 4.6.2.

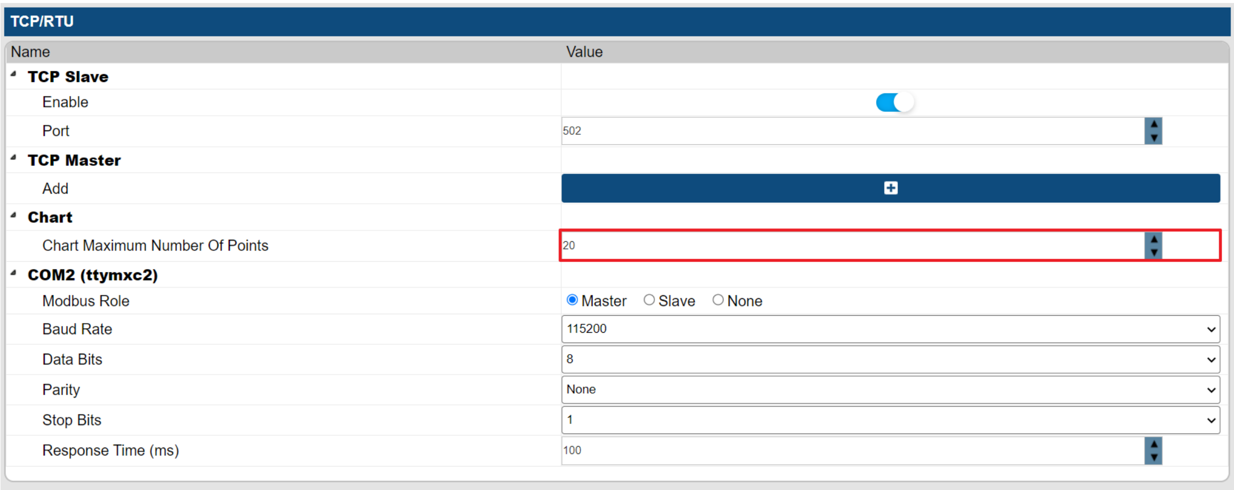


Figure 4-35 Establish the quantity of drawing points for Modbus Master.

Configure the communication parameters for Modbus RTU COM Port as specified in Table 4-15.

Table 4-15 Configure the communication parameters for Modbus RTU COM Port.

參數名稱	參數內容
Modbus 角色	切換 Master / Slave / None
鮑率	選擇鮑率
資料位元	選擇資料位元
同位元檢查	選擇同位元檢查
停止位元	選擇停止位元
反應時間	Modbus 角色切換到 Master，輸入反應時間
Slave ID	Modbus 角色切換到 Slave，輸入 Slave ID

4.1.2.2 Coils Status

Click on the icon to access the Coils Status Table for Modbus (as shown in Figure 4-36). The following section provides an explanation of the coil status addresses corresponding to the nDAS I/O Board.

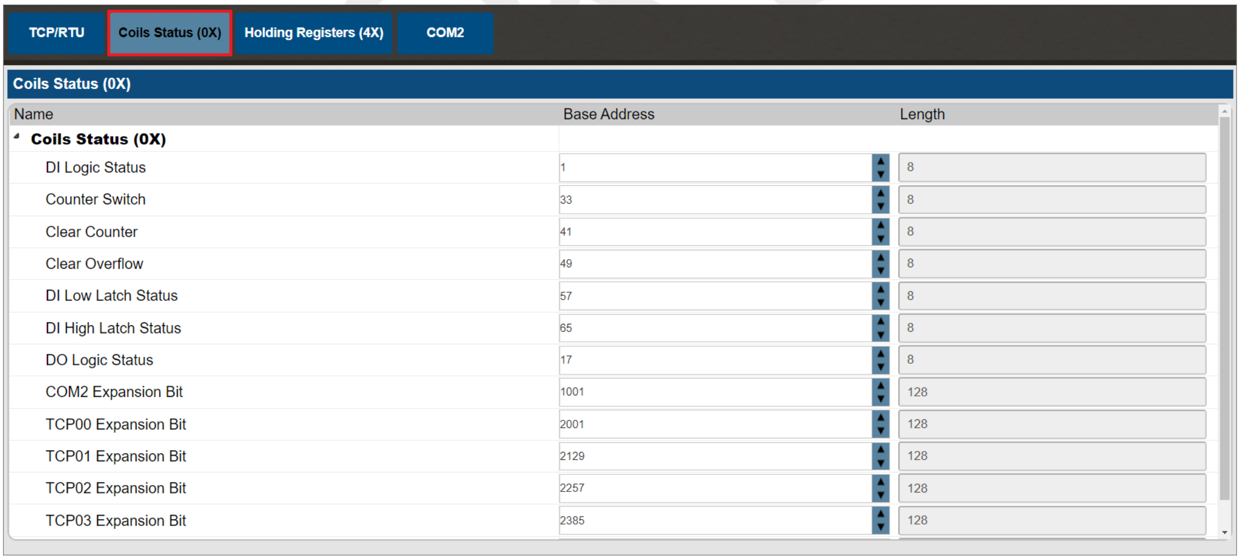


Figure 4-36 Modbus Coils Status Table

4.1.2.2.1 ns050

Name	Base Address	Length
Digital Input Logic State	1	8
Counter Switching	33	8
Clear Counter	41	8

Clear Overflow	49	8
DI Low Latch State	57	8
DI High Latch State	65	8
Digital Output Logic State	17	8
COM2 Expansion Bit	1001	128
TCP00 Expansion Bit	2001	128
TCP01 Expansion Bit	2129	128
TCP02 Expansion Bit	2257	128
TCP03 Expansion Bit	2385	128
TCP04 Expansion Bit	2513	128

4.1.2.2.2 ns051

Name	Base Address	Length
Digital Input Logical State	1	16
Counter Toggle	33	16
Counter Reset	49	16
Overflow Clear	65	16
DI Low Latching State	81	16
DI High Latching State	97	16
COM2 Expansion Bit	1001	128
TCP00 Expansion Bit	2001	128
TCP01 Expansion Bit	2129	128
TCP02 Expansion Bit	2257	128
TCP03 Expansion Bit	2385	128
TCP04 Expansion Bit	2513	128

4.1.2.2.3 ns056

Name	Base Address	Length
Digital Output Logical State	17	16
COM2 Expansion Bit	1001	128
TCP00 Expansion Bit	2001	128
TCP01 Expansion Bit	2129	128

TCP02 Expansion Bit	2257	128
TCP03 Expansion Bit	2385	128
TCP04 Expansion Bit	2513	128

4.1.2.2.4 ns017

Name	Base Address	Length
Reset Historical Maximum AI Value	101	9
Reset Historical Minimum AI Value	110	9
High Alarm Flag	131	9
Low Alarm Flag	141	9
COM2 Expansion Bit	1001	128
TCP00 Expansion Bit	2001	128
TCP01 Expansion Bit	2129	128
TCP02 Expansion Bit	2257	128
TCP03 Expansion Bit	2385	128
TCP04 Expansion Bit	2513	128

4.1.2.3 Holding Registers

Click on the icon to access the Holding Registers Table for Modbus (as shown in Figure 4-37). The following section provides an explanation of the holding register addresses corresponding to the nDAS I/O Board.

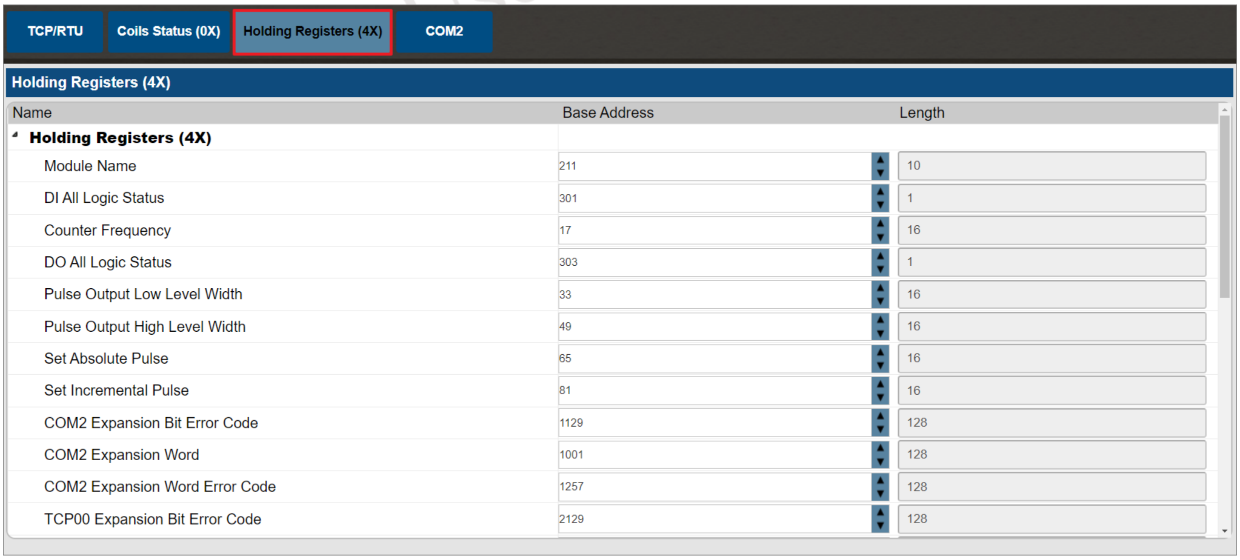


Figure 4-37 Modbus Holding Registers Table

4.1.2.3.1 ns050

Name	Base Address	Length
Module Name	211	10
All Digital Input Logical States	301	1
Counter Frequency	17	16
All Digital Output Logical States	303	1
Pulse Output Low Level Width	33	16
Pulse Output High Level Width	49	16
Set Absolute Pulse	65	16
Set Incremental Pulse	81	16
COM2 Expansion Bit Error Code	1129	128
COM2 Expansion Word	1001	128
COM2 Expansion Word Error Code	1257	128
TCP00 Expansion Bit Error Code	2129	128
TCP00 Expansion Word	2001	128
TCP00 Expansion Word Error Code	2257	128
TCP01 Expansion Bit Error Code	2513	128
TCP01 Expansion Word	2385	128
TCP01 Expansion Word Error Code	2641	128
TCP02 Expansion Bit Error Code	2897	128
TCP02 Expansion Word	2769	128
TCP02 Expansion Word Error Code	3025	128
TCP03 Expansion Bit	3281	128

Error Code		
TCP03 Expansion Word	3153	128
TCP03 Expansion Word Error Code	3409	128
TCP04 Expansion Bit Error Code	3665	128
TCP04 Expansion Word	3537	128
TCP04 Expansion Word Error Code	3793	128
Operation	4001	200
Data Recording Status	5101	1
Wi-Fi RSSI Status	5302	1

4.1.2.3.2 ns051

Name	Base Address	Length
Module Name	211	10
All Digital Input Logical States	301	1
Counter Frequency	17	32
COM2 Expansion Bit Error Code	1129	128
COM2 Expansion Word	1001	128
COM2 Expansion Word Error Code	1257	128
TCP00 Expansion Bit Error Code	2129	128
TCP00 Expansion Word	2001	128
TCP00 Expansion Word Error Code	2257	128
TCP01 Expansion Bit Error Code	2513	128
TCP01 Expansion Word	2385	128

TCP01 Expansion Word Error Code	2641	128
TCP02 Expansion Bit Error Code	2897	128
TCP02 Expansion Word	2769	128
TCP02 Expansion Word Error Code	3025	128
TCP03 Expansion Bit Error Code	3281	128
TCP03 Expansion Word	3153	128
TCP03 Expansion Word Error Code	3409	128
TCP04 Expansion Bit Error Code	3665	128
TCP04 Expansion Word	3537	128
TCP04 Expansion Word Error Code	3793	128
Operation	4001	200
Data Recording Status	5101	1
Wi-Fi RSSI Status	5302	1

4.1.2.3.3 ns056

Name	Base Address	Length
Module Name	211	10
All Digital Output Logical States	303	1
Pulse Output Low Level Width	17	32
Pulse Output High Level Width	49	32
Set Absolute Pulse	81	32
Set Incremental Pulse	113	32

COM2 Expansion Bit Error Code	1129	128
COM2 Expansion Word	1001	128
COM2 Expansion Word Error Code	1257	128
TCP00 Expansion Bit Error Code	2129	128
TCP00 Expansion Word	2001	128
TCP00 Expansion Word Error Code	2257	128
TCP01 Expansion Bit Error Code	2513	128
TCP01 Expansion Word	2385	128
TCP01 Expansion Word Error Code	2641	128
TCP02 Expansion Bit Error Code	2897	128
TCP02 Expansion Word	2769	128
TCP02 Expansion Word Error Code	3025	128
TCP03 Expansion Bit Error Code	3281	128
TCP03 Expansion Word	3153	128
TCP03 Expansion Word Error Code	3409	128
TCP04 Expansion Bit Error Code	3665	128
TCP04 Expansion Word	3537	128
TCP04 Expansion	3793	128

Word Error Code		
Operation	4001	200
Data Recording Status	5101	1
Wi-Fi RSSI Status	5302	1

4.1.2.3.4 ns017

Name	Base Address	Length
Module Name	211	10
AI Value	1	9
AI Status	10	16
Historical Maximum AI Value	111	9
Historical Minimum AI Value	121	9
AI Engineering Value	131	18
Historical Maximum AI Engineering Value	151	18
Historical Minimum AI Engineering Value	171	18
AI Scale Value	191	9
AI Range Code	201	9
AI Channel Mask	221	1
AI Physical Status	231	18
COM2 Expansion Bit Error Code	1129	128
COM2 Expansion Word	1001	128
COM2 Expansion Word Error Code	1257	128
TCP00 Expansion Bit Error Code	2129	128
TCP00 Expansion Word	2001	128
TCP00 Expansion Word Error Code	2257	128

TCP01 Expansion Bit Error Code	2513	128
TCP01 Expansion Word	2385	128
TCP01 Expansion Word Error Code	2641	128
TCP02 Expansion Bit Error Code	2897	128
TCP02 Expansion Word	2769	128
TCP02 Expansion Word Error Code	3025	128
TCP03 Expansion Bit Error Code	3281	128
TCP03 Expansion Word	3153	128
TCP03 Expansion Word Error Code	3409	128
TCP04 Expansion Bit Error Code	3665	128
TCP04 Expansion Word	3537	128
TCP04 Expansion Word Error Code	3793	128
Operation	4001	200
Data Recording Status	5101	1
Wi-Fi RSSI Status	5302	1

4.1.2.4 COM Port & TCP Rules

nDAS provides monitoring for both Master COM Port and Master TCP Port. To add TCP Master, please refer to section 4.1.2.1. Click on the icon to access the monitoring dashboard, as shown in Figure 4-38.

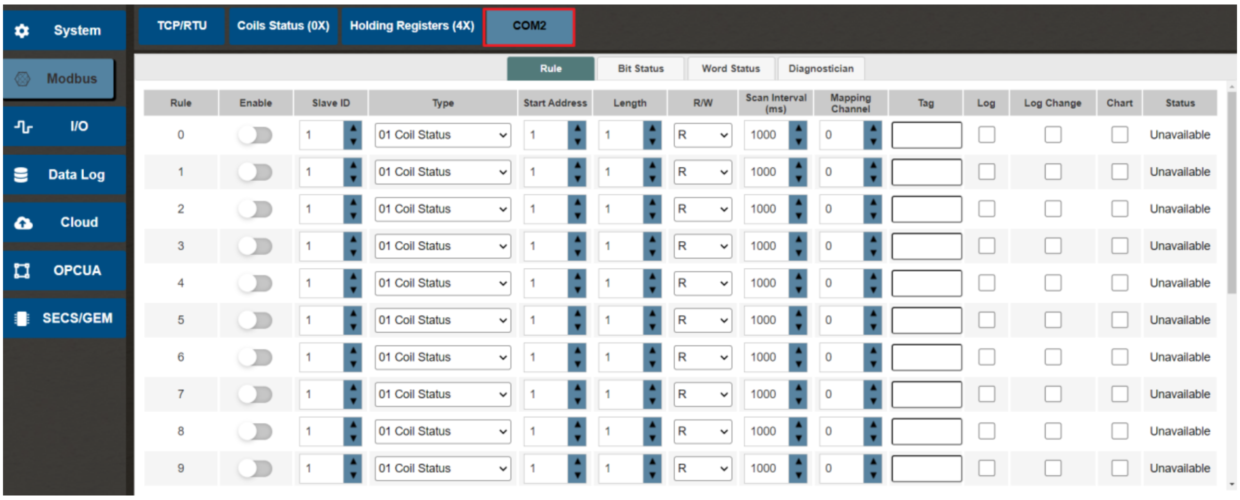


Figure 4-38 Modbus Master Monitoring Instrument Panel Configuration

4.1.2.4.1 Rule

Click on the icon to switch to the Modbus Master Rule Parameter Configuration screen (as shown in Figure 4-39). On this page, you can set rules to enable/disable Modbus channels. Each port provides 32 rules for user configuration. The parameter explanations can be found in Table 4-16.

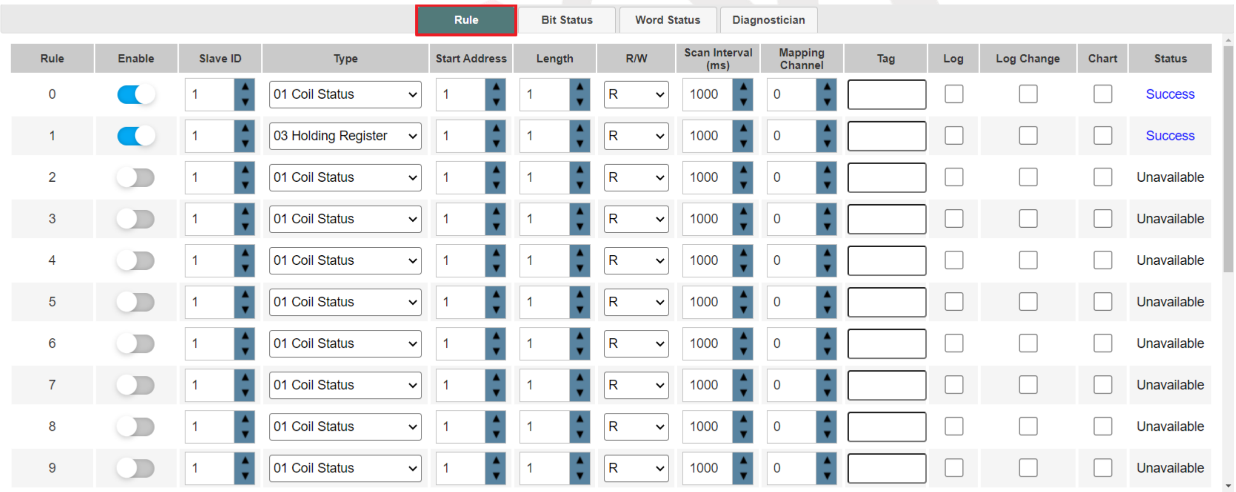


Figure 4-39 Modbus Master Rule Parameter Configuration

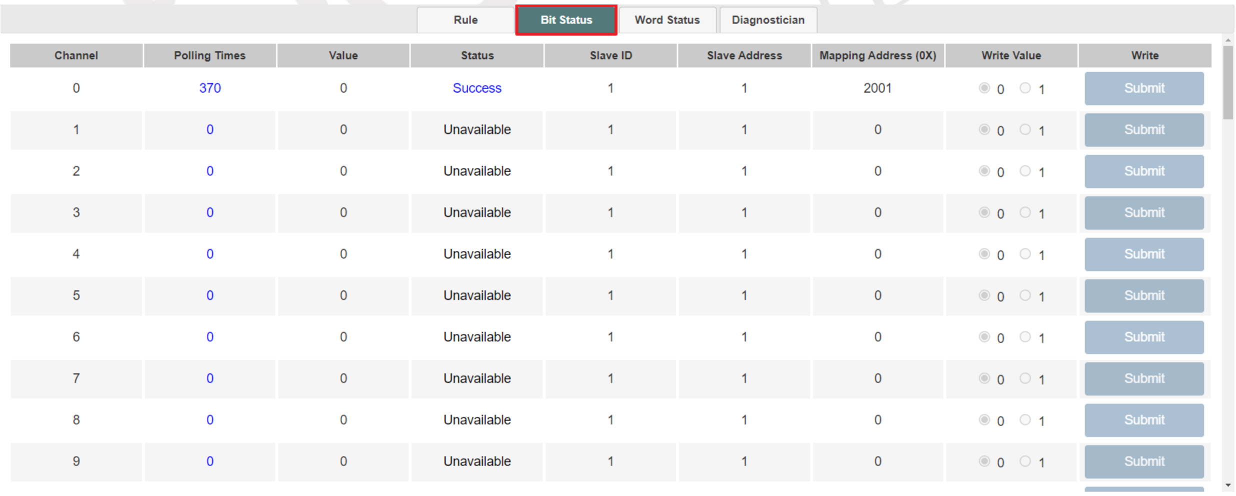
Table 4-16 Modbus Master Rule Parameter Configuration Table

Name	Description
Rule	Display rule number.
Enable	Enable/Disable rule.
Slave ID	Set connected Slave ID
Type	Modbus commonly used function codes 01~04.
Start Address	Set Modbus starting address.

Length	Set Modbus data length.
R/W	Set read/write status.
Scan Interval (ms)	Set scan time.
Mapping Channel	Each data corresponds to a channel. Note ： nDAS provides 128 channels.
Tag	Display rule tag in Data Log.
Log	Log to Data Log or not
Log Change	Log when there’s a change in reading value to Data Log.
Chart	Plot or not, refer to 4.6.2.
Status	Display Modbus read/write status.

4.1.2.4.2 Bit Status

Click on the icon to switch to the Modbus Bit Status Parameter Configuration screen (as shown in Figure 4-40). This screen allows you to monitor the Modbus bit status. If you set the Modbus Master Rule data type to 01 or 02 and activate the rule, it will be displayed on this screen.



Channel	Polling Times	Value	Status	Slave ID	Slave Address	Mapping Address (0X)	Write Value	Write
0	370	0	Success	1	1	2001	0 1	Submit
1	0	0	Unavailable	1	1	0	0 1	Submit
2	0	0	Unavailable	1	1	0	0 1	Submit
3	0	0	Unavailable	1	1	0	0 1	Submit
4	0	0	Unavailable	1	1	0	0 1	Submit
5	0	0	Unavailable	1	1	0	0 1	Submit
6	0	0	Unavailable	1	1	0	0 1	Submit
7	0	0	Unavailable	1	1	0	0 1	Submit
8	0	0	Unavailable	1	1	0	0 1	Submit
9	0	0	Unavailable	1	1	0	0 1	Submit

Figure 4-40 Modbus Bit Status Monitoring Screen

4.1.2.4.3 Word Status

Click on the icon to switch to the Modbus Word Status Monitoring screen (as shown in Figure 4-41). This screen allows you to monitor the Modbus word status. If you set the Modbus Master Rule data type to 03 or 04 and activate the rule, it will be displayed on this screen.

Rule | Bit Status | Word Status | Diagnostician

Channel	Polling Times	Value	Status	Slave ID	Slave Address	Mapping Address (4X)	Write Value	Write
0	409	0	Success	1	1	2001	0	Submit
1	0	0	Unavailable	1	1	0	0	Submit
2	0	0	Unavailable	1	1	0	0	Submit
3	0	0	Unavailable	1	1	0	0	Submit
4	0	0	Unavailable	1	1	0	0	Submit
5	0	0	Unavailable	1	1	0	0	Submit
6	0	0	Unavailable	1	1	0	0	Submit
7	0	0	Unavailable	1	1	0	0	Submit
8	0	0	Unavailable	1	1	0	0	Submit
9	0	0	Unavailable	1	1	0	0	Submit

Figure 4-41 Modbus Word Status Monitoring Screen

4.1.2.4.4 Diagnostician

Click on the icon to switch to the Modbus Master Rule Monitoring screen (as shown in Figure 4-42). This screen displays the current communication status of each rule.

Rule | Bit Status | Word Status | Diagnostician

Rule	Polling Times	Current Response Time (ms)	Max. Response Time (ms)	Min. Response Time (ms)	Status
0	449	0.912	25.602	0.674	Success
1	453	0.686	41.325	0.671	Success
2	0	0	0	60000	Unavailable
3	0	0	0	60000	Unavailable
4	0	0	0	60000	Unavailable
5	0	0	0	60000	Unavailable
6	0	0	0	60000	Unavailable
7	0	0	0	60000	Unavailable
8	0	0	0	60000	Unavailable
9	0	0	0	60000	Unavailable

Figure 4-42 Modbus Master Rule Monitoring Screen

4.1.3 I/O

By clicking on the icon in the left-side menu, you can access the I/O Parameter Configuration as well as the screens for each I/O module. nDAS provides different settings for different modules under the I/O menu (as shown in Figure 4-43). Table 4-17 provides the common parameter settings for the I/O modules.

Figure 4-43 I/O Module(Simulator)

Table 4-17 Common parameter settings for I/O modules

Parameter name		Content
I/O Settings	Name	Show current module name
	Interval (ms)	Set the time interval for each I/O read. Default value is 1000 ms.
	Chart Maximum Number Of Points	Set the maximum number of data points that can be plotted by the oscilloscope.

4.1.3.1 Digital Input

Click on the icon to access the DI module (as shown in Figure 4-44). The interface encompasses DI parameter configuration and oscilloscope signal display. Table 4-18 presents the fundamental DI parameter settings.

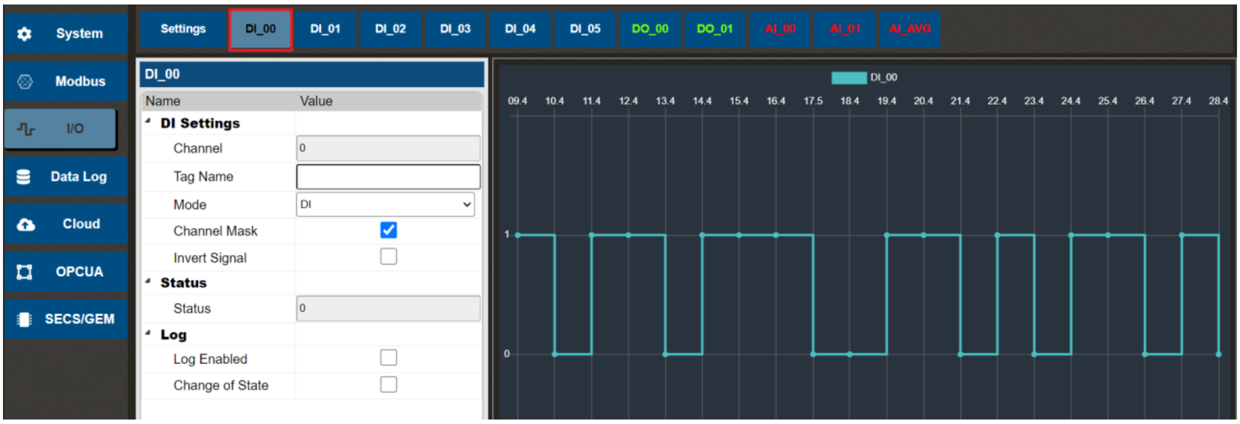


Figure 4-44 Digital Input Settings Screen

Table 4-18 Digital Input Settings

Parameter name		Content
DI Settings	Channel	Display channel ID
	Tag Name	Configure label names
	Mode	Present the following mode options: Digital Input (DI) Counter Low-to-High Latch High-to-Low Latch Frequency
	Channel Mask	Enable/Disable channel masking
	Invert Signal	Activate/Deactivate signal inversion
Status	Status	Display DI signal status
Log	Log Enable	Enable/Disable recording of DI channel
	Change of State	Record changes in DI channel signal
Counter	Start	Activate/Deactivate counter function
	Reset	Clicking the button will reset the current channel count
	Overflow Auto Reset	Enable/Disable automatic reset on counter overflow
	Overflow	Display overflow status
Frequency	Precision	0.1Hz
		0.01Hz
	Reset Time(ms)	Enter reset time

The logging feature must be enabled by selecting the logging option to write the channel data into the database. The channel masking option allows you to choose whether to hide the data of that channel. If the channel masking is enabled, the channel data will not be written into the database and will cease to be displayed on the oscilloscope.

Note: Channel data will be written into the database only when both the logging feature and channel masking are enabled.

The status change option allows for an additional annotation in the database when the channel signal status changes.

Note: This feature requires the logging function to be enabled as well.

4.1.3.2 Digital Output

Click on the icon to access the DO module (as shown in Figure 4-45). The interface encompasses DO parameter configuration and oscilloscope signal display. Table 4-19 presents the fundamental DO parameter settings.

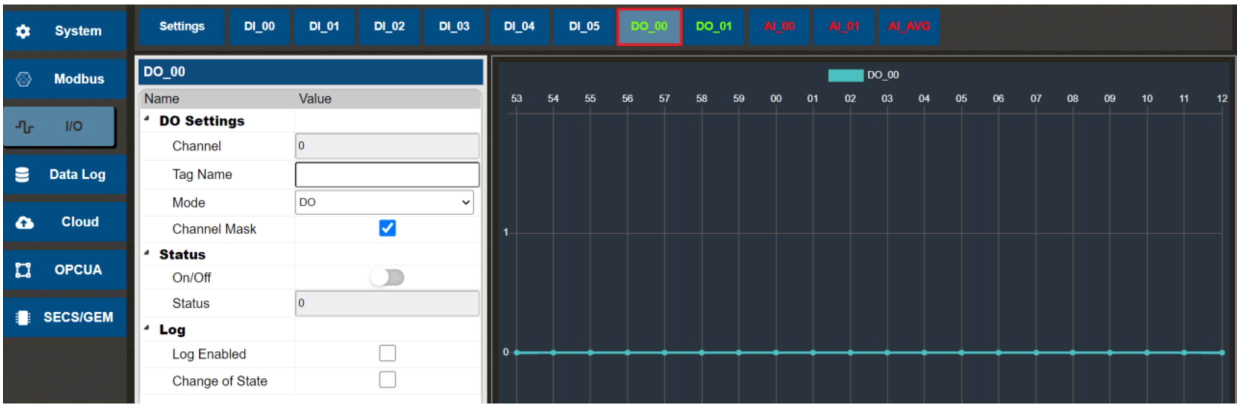


Figure 4-45 Digital Output Settings Screen

Table 4-19 Digital Output Settings

Parameter name		Content
DO Settings	Channel	Display channel ID
	Tag Name	Configure label names
	Mode	Present the following mode options: Digital Output (DO) Pulse Output AI Alarm Driven
	Channel Mask	Enable channel masking
Status	On/Off	Activate/Deactivate DO signal output
	Status	Display DO status
Log	Log Enable	Enable/Disable recording of DO channel
	Change of State	Record changes in DO channel signal
Pulse Output(Pulse Output Mode)	Low Signal Width (ms)	Set low signal output time
	High Signal Width(ms)	Set high signal output time
	Output Frequency	Set output frequency based on signal width
	Duty Cycle	Set duty cycle based on signal width
Enable/Disable(Pulse Output Mode)	Continuous	Activate continuous pulse signal output
	Fixed Total	Set pulse output count

	Start	Start pulse output
	Stop	Stop pulse output
AI Alarm Driven (AI Alarm Driven Mode)	Mapping Channel	Map to AI channel
	Trigger Mode	Deactivate High Alarm Low Alarm

The logging feature must be enabled by selecting the logging option to write the channel data into the database. The channel masking option allows you to choose whether to hide the data of that channel. If the channel masking is enabled, the channel data will not be written into the database and will cease to be displayed on the oscilloscope.

Note: Channel data will be written into the database only when both the logging feature and channel masking are enabled.

The status change option allows for an additional annotation in the database when the channel signal status changes.

Note: This feature requires the logging function to be enabled as well.

4.1.3.3 Analog Input

Click on the icon to access the AI module (as shown in Figure 4-46). The interface encompasses AI parameter configuration and oscilloscope signal display. Table 4-20 presents the AI parameter settings.

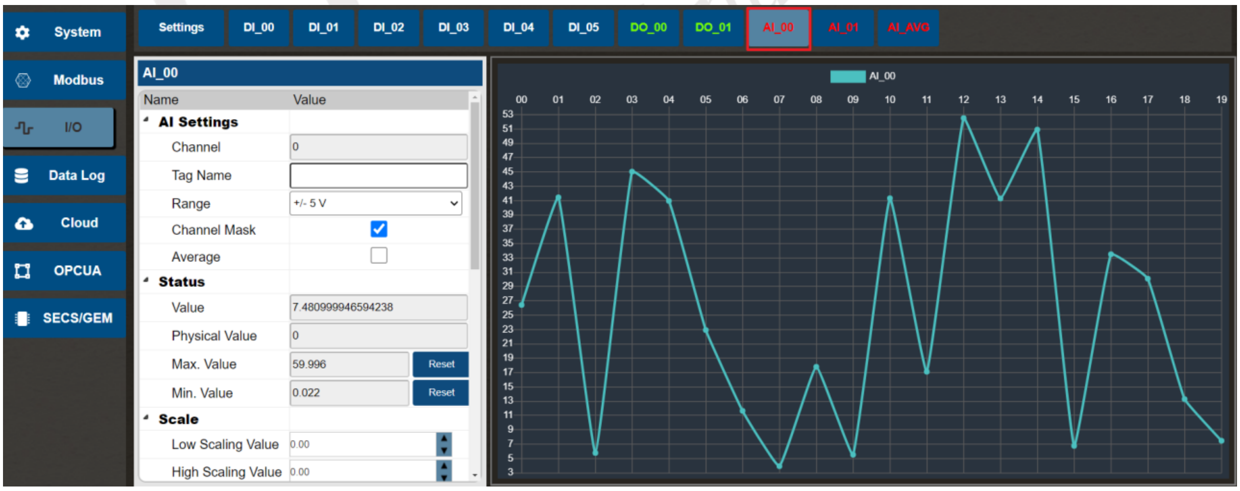


Figure 4-46 Analog Input Settings Screen

Table 4-20 Analog Input Settings

Parameter name		Content
AI Settings	Channel	Display channel ID

	Tag Name	Configure label names
	Range	Present the following voltage and current range options: +- 11 V +- 10 V +- 5 V +- 2.5 V +- 1.25 V +- 0.625 V +- 0.312 V +- 0.156 V +- 0.0781 V +- 0.0391 V +- 0.0195 V 0 ~ 20 mA +- 20 mA 4 ~ 20 mA
	Channel Mask	Enable/Disable masking
	Average	Enable/Disable calculating average value
Status	Value	Current measured value
	Physical Value	Physical mapped value
	Max. Value	Historical maximum value
	Min. Value	Historical minimum value
Scale	Low Scaling Value	Voltage or current scaling lower limit
	High Scaling Value	Voltage or current scaling upper limit
	Physical Min. Scaling Value	Physical value scaling lower limit
	Physical Max. Scaling Value	Physical value scaling upper limit
High Alarm	Enable	Enable High Threshold Alarm
	Mode	Present two mode options: Momentary Latch
	Value	Configure alarm threshold values
	Status	Display alarm status

	Clear	Clear alarm status
Low Alarm	Enable	Enable Low Threshold Alarm
	Mode	Present two mode options: Momentary Latch
	Value	Configure alarm threshold values
	Status	Display alarm status
	Clear	Clear alarm status
Log	Log Enable	Enable/Disable channel logging function
	Change of State	Record AI channel signal changes. When the amplitude exceeds the error value, the event will be logged.
	Deviation(%)	Set error value

The logging feature must be enabled by selecting the logging option to write the channel data into the database. The channel masking option allows you to choose whether to hide the data of that channel. If the channel masking is enabled, the channel data will not be written into the database and will cease to be displayed on the oscilloscope.
Note: Channel data will be written into the database only when both the logging feature and channel masking are enabled.

The status change option allows for an additional annotation in the database when the channel signal status changes.
Note: This feature requires the logging function to be enabled as well.

In the given scenario, if we assume the mapping of 0 to 50 ℃ to 0.5 to 3 VDC, the parameter settings would be as shown in Figure 4-47.

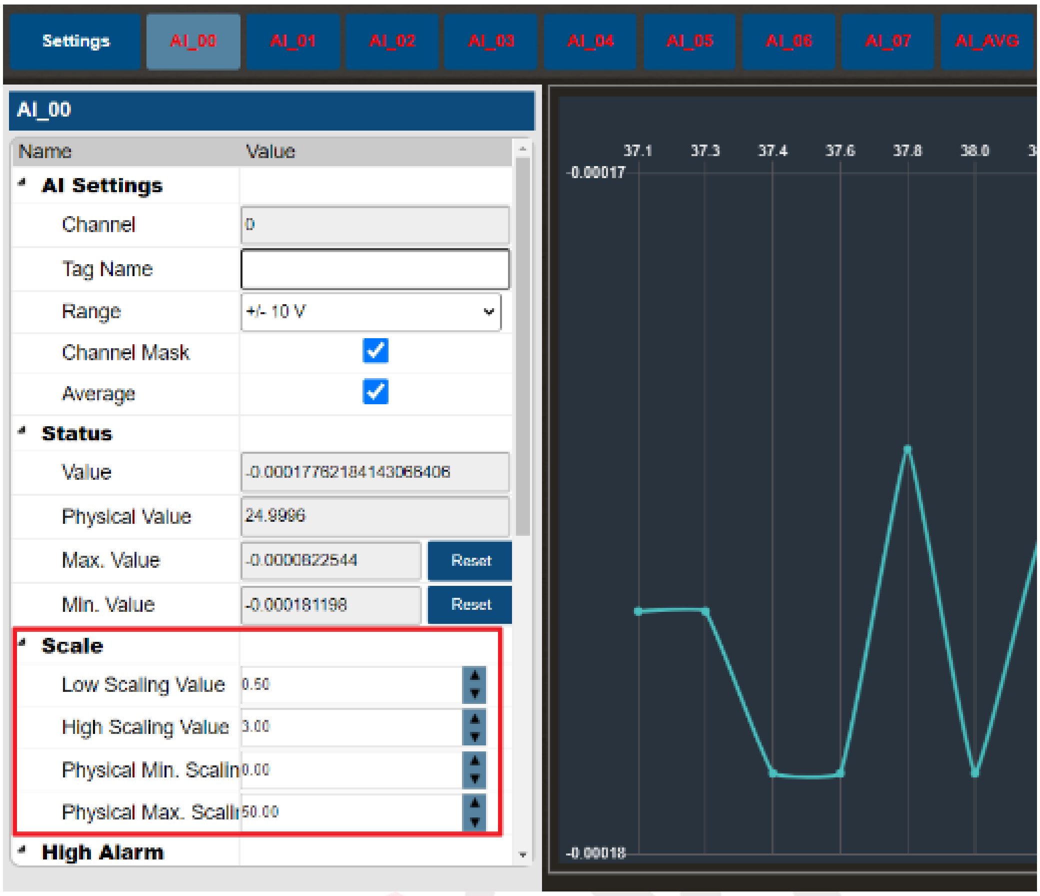


Figure 4-47 Physical value configuration

If an input voltage value of 2 VDC is applied to channel 0, the measured value would be as shown in Figure 4-48. Referring to Modbus Table 4.1.2.3.4, the value read from the Modbus address would be as depicted in Figure 4-49. The unit of the voltage value is in millivolts (mV).

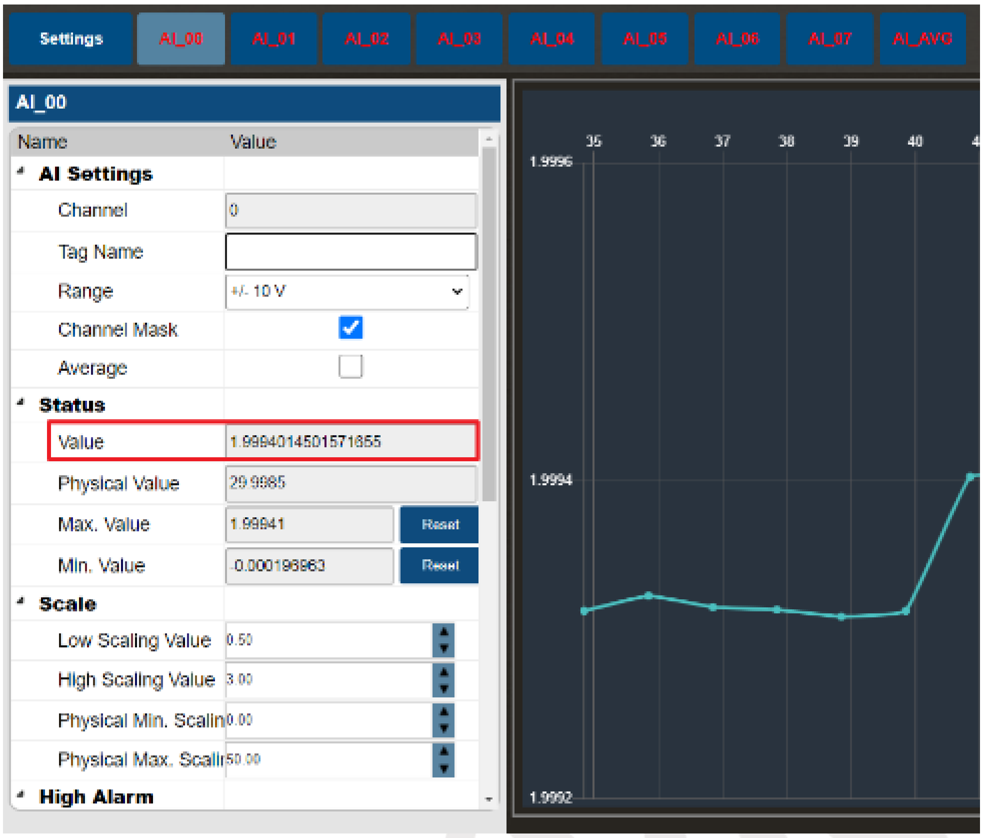


Figure 4-48 The actual measured voltage

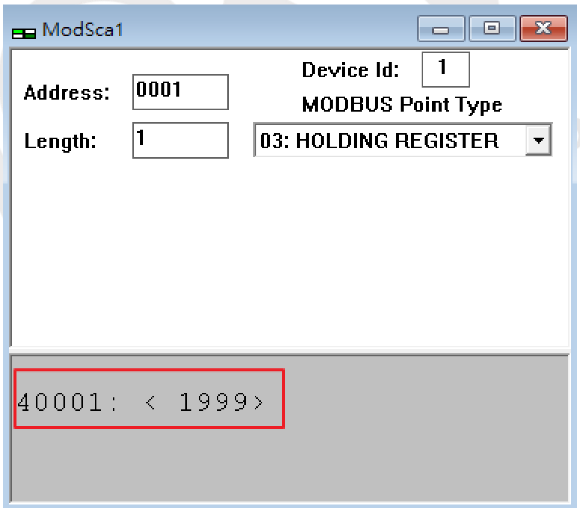


Figure 4-49 Modbus AI Value

The AI scaling value can be calculated using a proportional relationship between the current voltage value and the voltage range. The formula is as follows:

$$\frac{Range^{+} - Range^{-}}{Scaling_{High} - Scaling_{Low}} = \frac{Range^{+} - V_{Input}}{Scaling_{High} - V_{Scaling}}$$

In the given formula, V_{Input} represents the current measured voltage value, $V_{Scaling}$ represents the mapped voltage value, $Range^{+}$ represents the current high voltage range, $Range^{-}$ represents the current low voltage range, $Scaling_{High}$ represents the defined maximum scaling value, $Scaling_{Low}$ represents the defined minimum scaling value.

The AI Physical Value can be calculated using a proportional relationship between the AI Scaling Value and the Low/High Scaling Value range. The formula is as follows:

$$\frac{Scaling_{High} - Scaling_{Low}}{Physical_{Max} - Physical_{Min}} = \frac{V_{Scaling} - Scaling_{Low}}{Physical_{Mapping} - Physical_{Min}}$$

In the given formula, $V_{Scaling}$ represents the mapped voltage value (AI Scaling Value), $Physical_{Mapping}$ represents the mapped physical value (AI Physical Value), $Physical_{Max}$ represents the defined maximum physical scaling value, $Physical_{Min}$ represents the defined minimum physical scaling value, $Scaling_{High}$ represents the defined maximum scaling value, $Scaling_{Low}$ represents the defined minimum scaling value.

Referring to Modbus Table 4.1.2.3.4, the AI scaling value read from the Modbus address is depicted in Figure 4-50. The unit of the voltage value is in millivolts (mV).

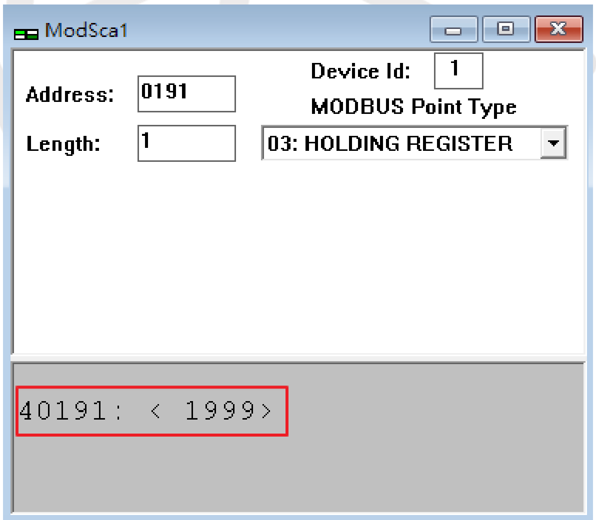


Figure 4-50 Modbus AI Scaling Value

If an input voltage value of 2 VDC is applied to channel 0, and the mapping scenario is from 0 to 50 ℃ to 0.5 to 3 VDC, the mapped physical value would be as shown in Figure 4-51. Referring to Modbus Table 4.1.2.3.4, the AI physical value read from the Modbus address is depicted in Figure 4-52. The unit of the voltage value is volts (V), and the data type is Float.

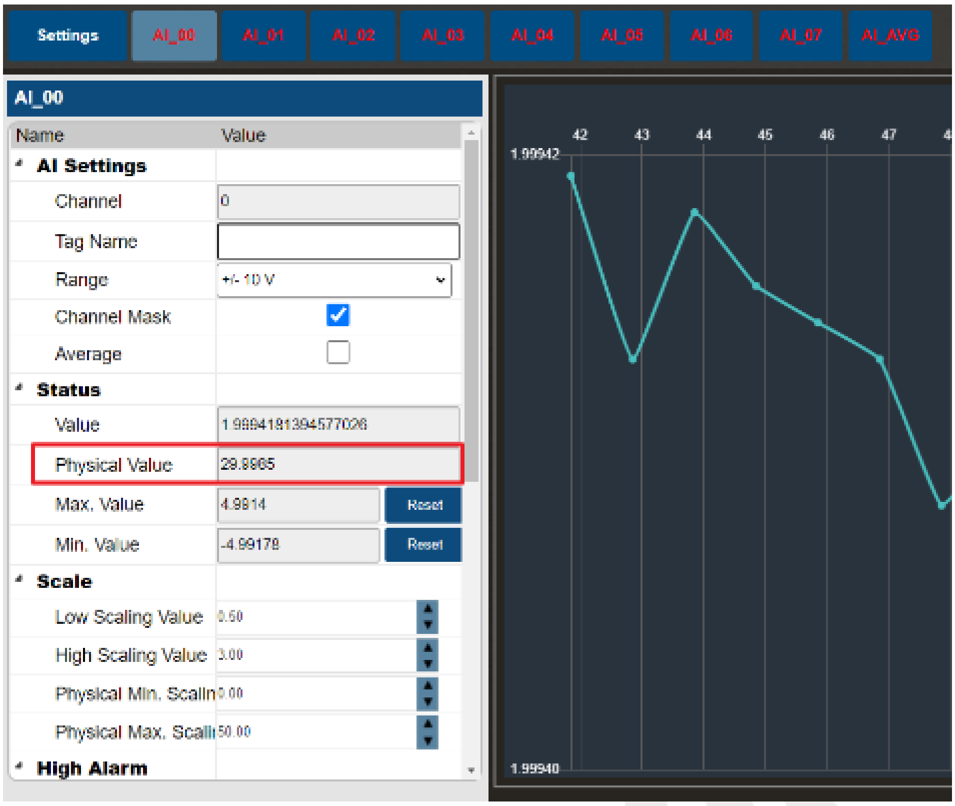


Figure 4-51 Mapping to physical value

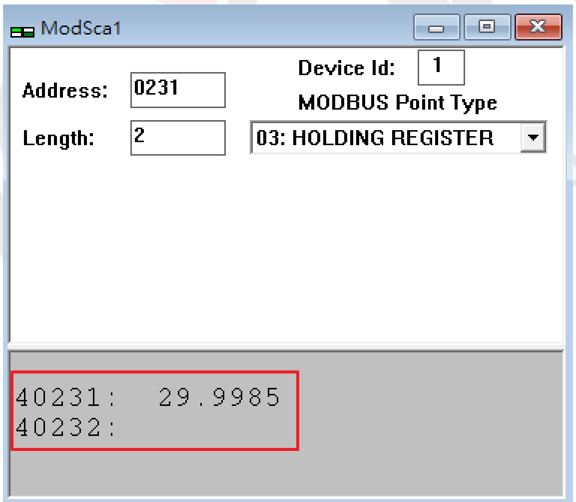


Figure 4-52 Modbus AI Physical Value

4.1.4 Data Log

By clicking on the icon in the left-side menu (as shown in Figure 4-53), you will access the screen for Data Log configuration and historical data query. nDAS provides the functionality of storing data in a database. By selecting the desired recording method through the parameter selection in the top menu, data will be automatically stored in the database. It also offers a database query feature, where you can set the query method in the top menu and input the query range to retrieve a data list on the webpage. For detailed instructions, please refer to sections 4.1.4.1 and 4.1.4.2.

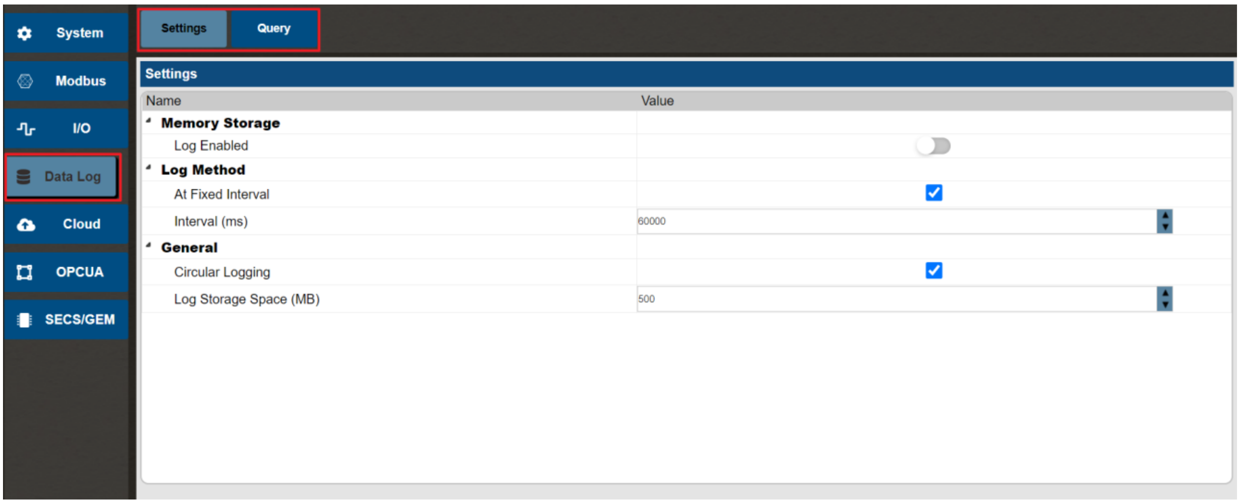


Figure 4-53 Data Log Module

4.1.4.1 Data Log Settings

By clicking on the configuration icon in the top menu, you can access the Data Log parameter settings (as depicted in Figure 4-53). Table 4-21 represents the fundamental parameters for Data Log configuration.

Table 4-21 Data Log Settings

Parameter name		Content
Memory Storage	Log Enable	Enable/Disable database logging.
Log Method	At Fixed Interval	Enable/Disable fixed interval logging mode.
	Interval(ms)	Set the time parameter value for fixed interval (100~6000000).
General	Circular Logging	Enable/Disable cyclic logging.
	Log Storage Space(MB)	Set the upper limit for stored data size.

The data storage functionality is set to Disable by default. The fixed interval parameter is checked by default, and when this parameter is enabled, nDAS will write data to the database at a uniform time interval, with a default value of 1 minute. If the cyclic logging option is checked, when the .db database size exceeds 10GB, the oldest records will be deleted to make space for new data. If the option is unchecked and the database is full, data writing to the database will stop. The record storage space can be configured to set the upper limit for the size of stored data in the database. For example, if the nDAS database has already stored data up to 500MB and the setting is changed to 100MB, no further data will be written to the database. The default value is 500MB.

4.1.4.2 Data Log Query

By clicking on the configuration icon in the top menu, you can access the screen for Data Log query method settings (as shown in Figure 4-54). Table 4-22 represents the parameters for Data Log query configuration.

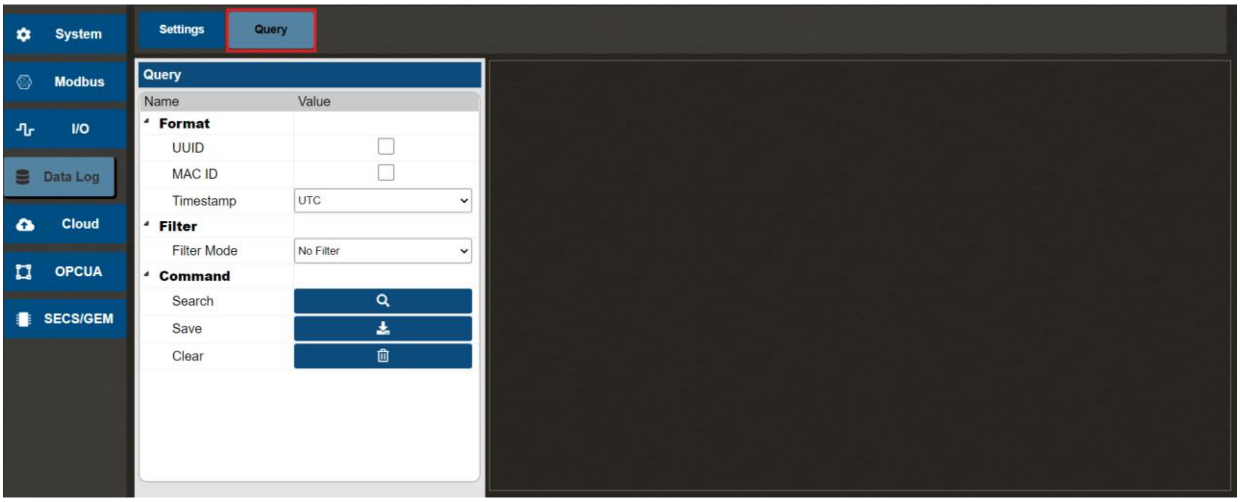


Figure 4-54 Data Log Query method settings

Table 4-22 Data Log query configuration

Parameter name		Content
Format	UUID	Display the field in the list.
	MAC ID	Display the field in the list.
	Timestamp	Provide the following mode selections: Coordinated Universal Time (UTC) Local Time
Filter	Filter Mode	Provide the following mode selections: No Filter Time Filter Amount of Latest Data Filter
	Start Time & End Time	Based on the Time Filtering mode, input the time range.
	Amount	Based on the Quantity Filtering for Latest Data mode, input the quantity.
Command	Search	Click the button to perform a database search.
	Save	Click the button to save the current list data as a .json file.
	Clear	Click the button to clear the database data.

The database query feature presents the content of the database in a list format on the webpage. If the Universal Unique Identifier (UUID) and MAC address are checked, the corresponding field will be displayed; otherwise, it will be hidden. The timestamp determines the display format of the time as either Coordinated Universal Time (UTC) or local time. The filtering mode controls the search conditions. nDAS offers two query modes: time-based and latest data count. If the time filtering mode is selected, the parameters for start and end time need to be filled in, or only one of them can be entered. If the latest data count filtering mode is selected, the parameter for quantity needs to be filled in. The search functionality will list the database content based on the set conditions and present it in a table format (as shown in Figure 4-55).

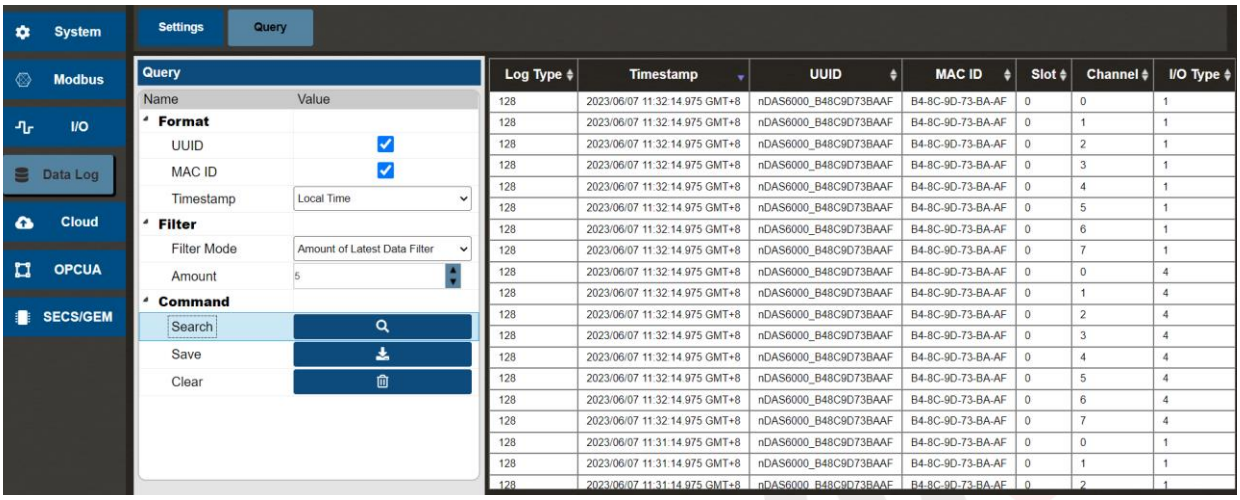


Figure 4-55 Data Log Query Result

4.1.5 Cloud

By clicking on the icon in the left-side menu, you can access the screen for cloud functionality settings. nDAS provides various cloud data automatic upload services, including Private Server, Dropbox, Google Drive, OneDrive, Azure Storage Accounts, AWS S3, and AWS IoT (as shown in Figure 4-56).

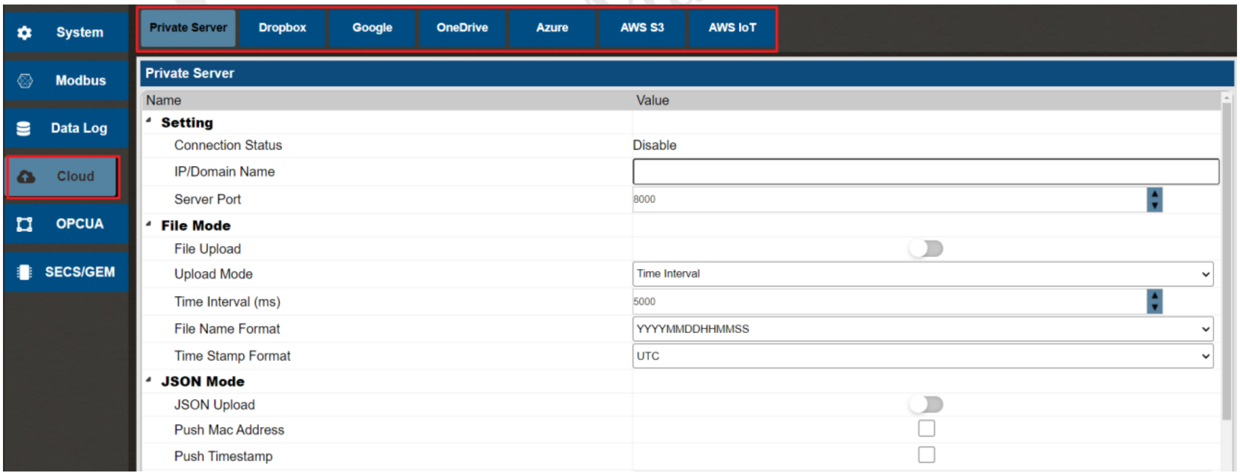


Figure 4-56 Cloud Module

4.1.5.1 Private Server

4.1.5.1.1 Basic Settings

Clicking on the icon will take you to the configuration screen for the Private Server service (as shown in Figure 4-57). Table 4-23 provides the settings for Private Server connection and upload methods. nDAS supports outputting data in JSON format strings as well as CSV files to the Private Server. Users need to provide their own private server as the receiving end and input the private server's IP and Port in the nDAS settings. Afterward, they can initiate file upload or transmission of JSON strings to begin the data transfer process.

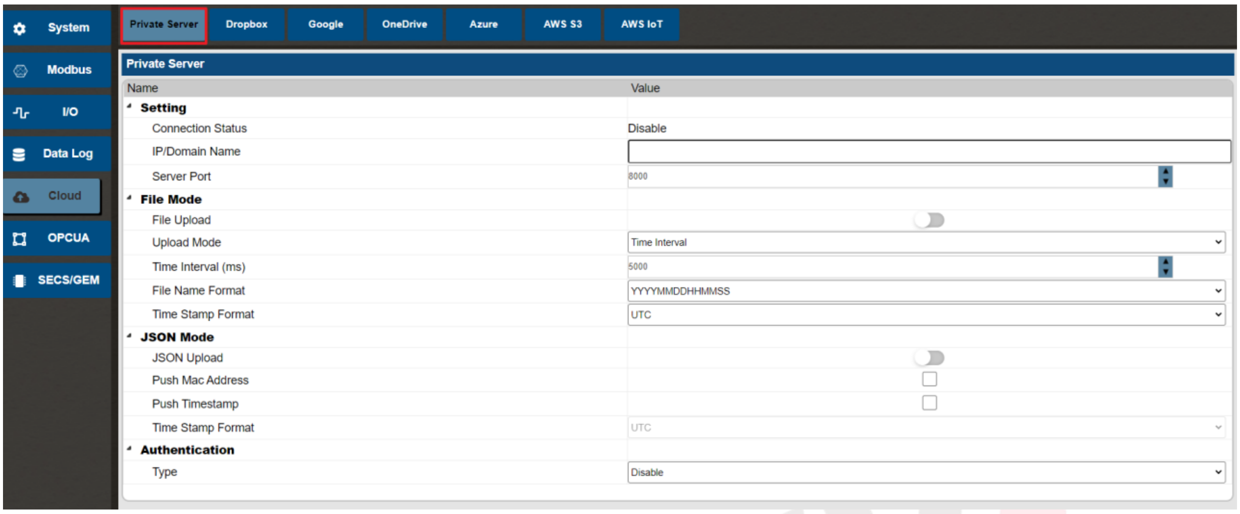


Figure 4-57 Private Server Connection Settings

Table 4-23 Private Server Connection Settings

Parameter name		Content
Settings	Connection status.	Display connection status.
	IP/Domain name.	Enter Server IP.
	Server port.	Enter Server Port.
File Mode	File upload.	Activate/Deactivate file upload.
	Upload method.	Time interval/Item cycle interval.
	Time interval (ms).	Enter time based on upload method time interval mode.
	Item cycle interval (quantity).	Enter quantity based on upload method item cycle interval mode.
	File name format.	YYYYMMDDHHMMSS / YYYYMMDD.
	Timestamp format.	UTC / Local Time.
JSON Mode	JSON upload.	Activate/Deactivate JSON message upload.
	Include MAC address.	Display MAC address in JSON message.
	Include timestamp.	Display timestamp in JSON message.
	Timestamp format.	UTC / Local Time.
Authentication	Type.	Disable/Enable Basic Authorization.
	Username.	Enter username for Basic Authorization.

	Password.	Enter password for Basic Authorization.

4.1.5.2 Dropbox

4.1.5.2.1 Basic Setting

Click on the icon to access the configuration for connecting to the Dropbox cloud service (as shown in Figure 4-58). Table 4-24 provides settings for the Dropbox cloud connection and upload methods.

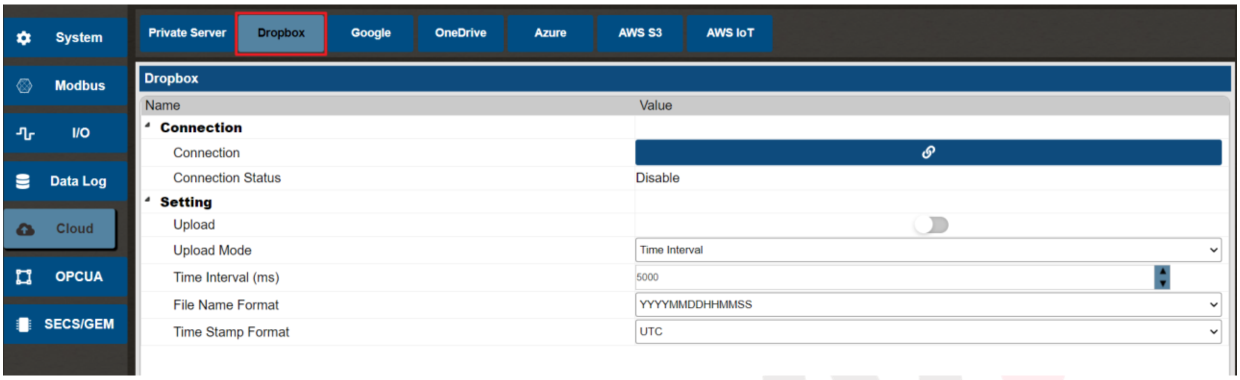


Figure 4-58 Dropbox Connection Settings Screen

Table 4-24 Dropbox Connection Settings

Parameter name		Content
Connection	Connection.	Click the button to access cloud permissions settings.
	Connection status.	Display connection status.
Settings	File upload.	Activate/Deactivate file upload.
	Upload method.	Time interval/Item cycle interval.
	Time interval (ms).	Enter time based on upload method time interval mode.
	Item cycle interval (quantity).	Enter quantity based on upload method item cycle interval mode.
	File name format.	YYYYMMDDHHMMSS / YYYYMMDD.
	Timestamp format.	UTC / Local Time.

4.1.5.2.2 Connection Process

Click on the "Connect" button (as shown in Figure 4-59) to display the cloud authorization dialog window (as shown in Figure 4-60).

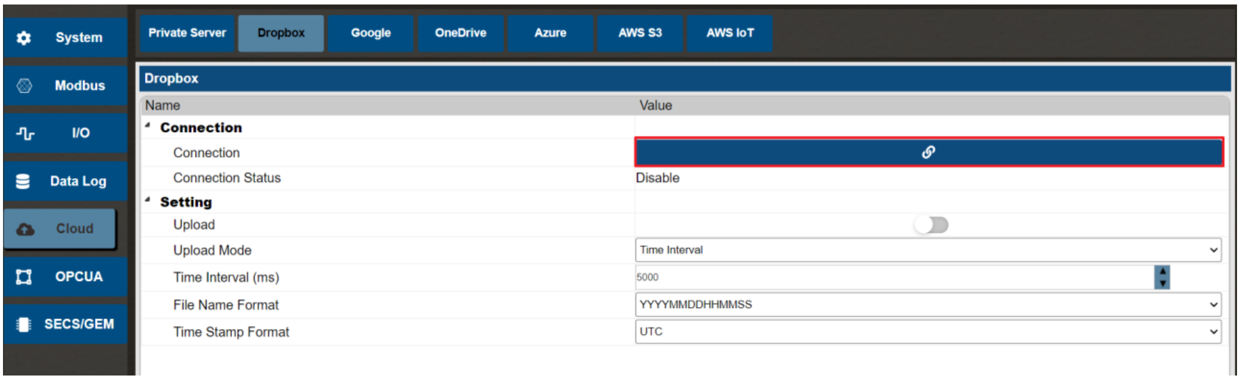


Figure 4-59 Click Button of Connection

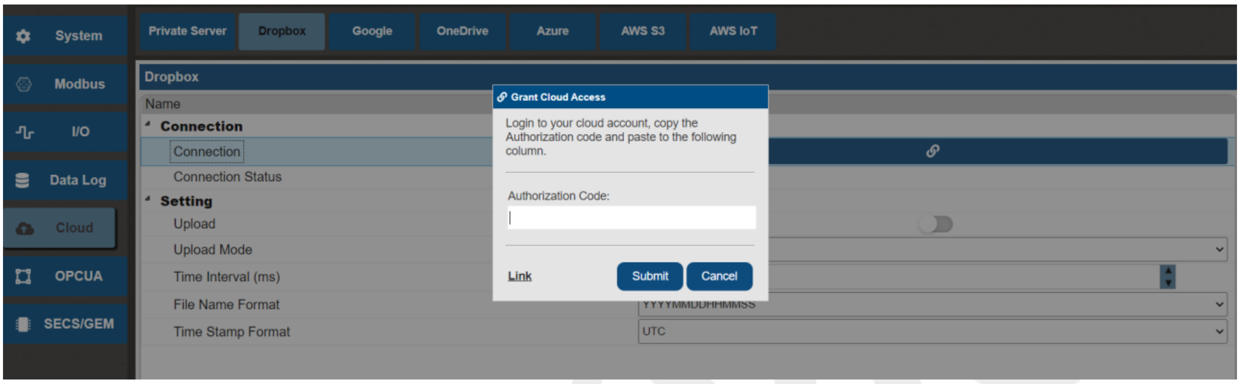


Figure 4-60 Cloud Authorization Dialog

Establishing the connection requires the user to log in to their personal Dropbox account. It uses OAuth 2.0 authentication method, which requires the user to log in and authorize nDAS to upload data to their personal cloud storage space.

Next, click on the external link in the cloud authorization dialog window (as shown in Figure 4-61). After clicking the link, you will be redirected to the Dropbox user login screen (as shown in Figure 4-62), where you will be prompted to log in to your personal account.

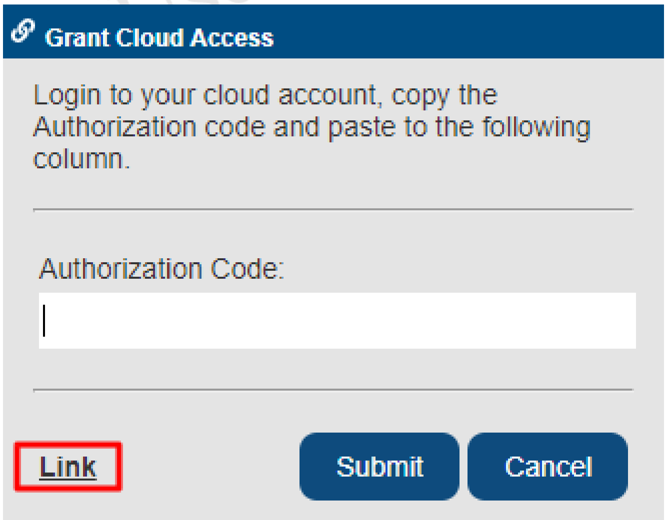


Figure 4-61 Click on the external link in the cloud authorization dialog window

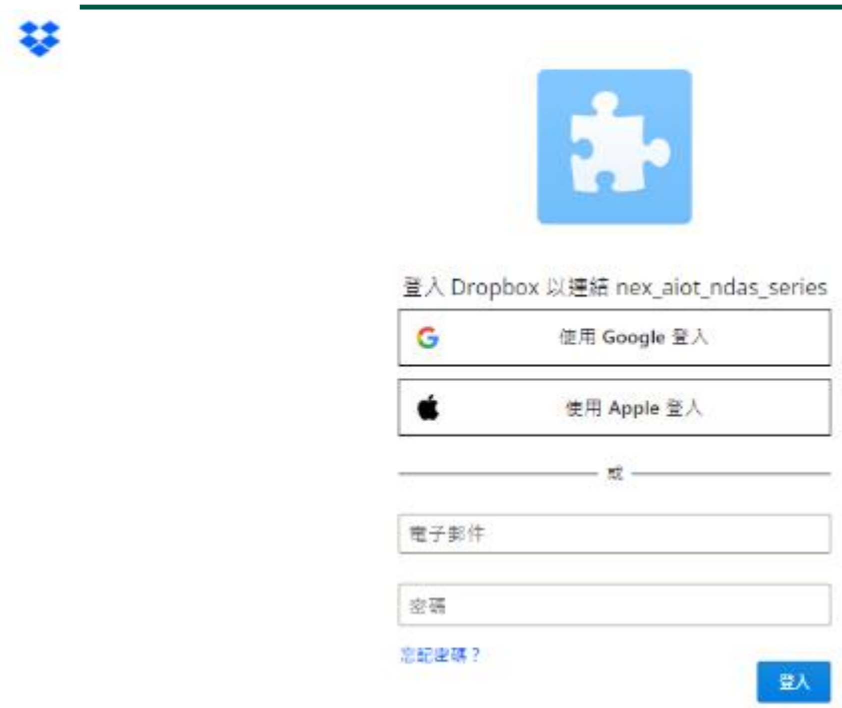


Figure 4-62 Dropbox user login screen

After successfully logging in, the web page will redirect you to the user confirmation screen (as shown in Figure 4-63). Click on the "Continue" button to proceed to the authorization screen.

Figure 4-63 Dropbox user confirmation screen

Next, the web page will redirect you to the user permission authorization screen (as shown in Figure 4-64). Click on the "Allow" button. After granting permission, a authorization code will be generated (as shown in Figure 4-65). Copy the authorization code and paste it into the field in the cloud authorization dialog window, then click on the "Execute" button (as shown in Figure 4-60).

Figure 4-64 Dropbox user permission authorization screen

Figure 4-65 Dropbox generates authorization code screen

Upon successful authorization, a dialog window will appear indicating that the authorization was successful. The connection status will also be updated to "Connected" (as shown in Figure 4-66, Figure 4-67).

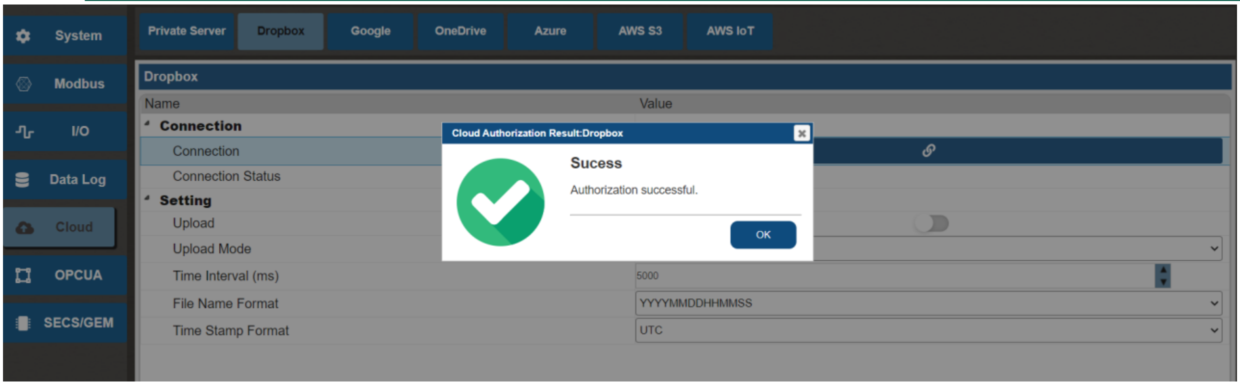


Figure 4-66 Dropbox authorization success dialog window

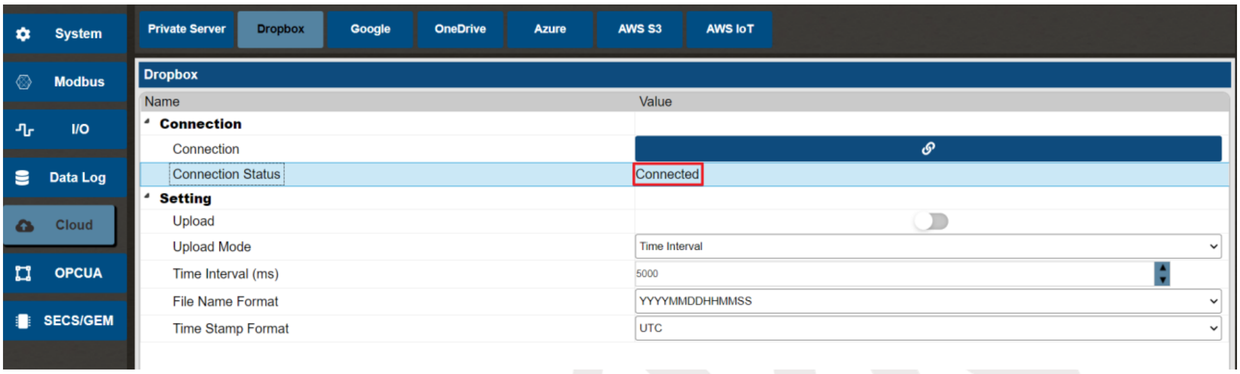


Figure 4-67 Dropbox connection status

4.1.5.3 GoogleDrive

4.1.5.3.1 Basic Setting

Click on the icon to access the configuration for connecting to the Google Drive cloud service (as shown in Figure 4-68). Table 4-25 provides settings for the Google Drive cloud connection and upload methods.

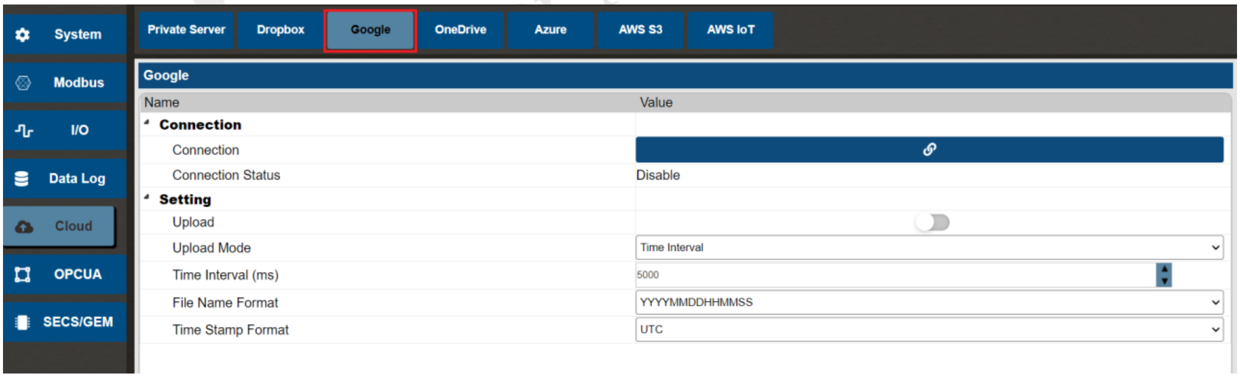


Figure 4-68 Google Drive cloud service connection settings

Table 4-25 Google Drive cloud connection setting parameters

Parameter name		Content
Connection	Connection.	Click the button to access cloud permissions settings.

	Connection status.	Display connection status.
Settings	File upload.	Activate/Deactivate file upload.
	Upload method.	Time interval/Item cycle interval.
	Time interval (ms).	Enter time based on upload method time interval mode.
	Item cycle interval (quantity).	Enter quantity based on upload method item cycle interval mode.
	File name format.	YYYYMMDDHHMMSS / YYYYMMDD.
	Timestamp format.	UTC / Local Time.

4.1.5.3.2 Connection Process

Click on the "Connect" button (as shown in Figure 4-69) to display the cloud authorization dialog window (as shown in Figure 4-70).

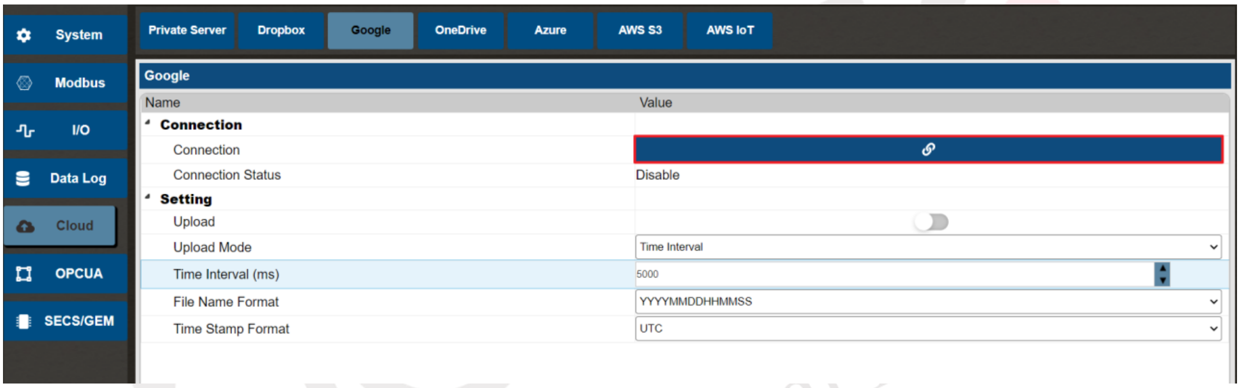


Figure 4-69 Click on the "Connect" button

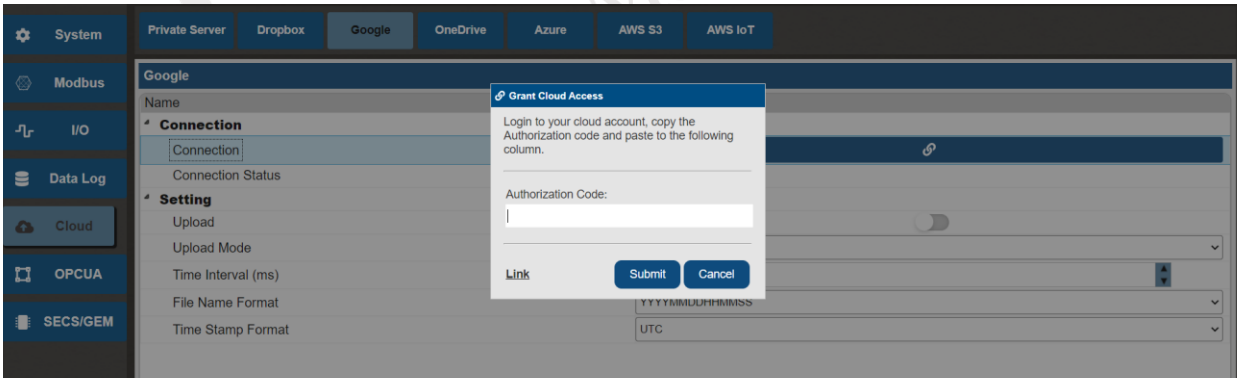


Figure 4-70 Cloud authorization dialog window

Establishing the connection requires the user to log in to their personal Google Drive account. It uses OAuth 2.0 authentication method, which requires the user to log in and authorize nDAS to upload data to their personal cloud storage space.

Next, click on the external link in the cloud authorization dialog window (as shown in Figure 4-71). After clicking the link, you will be redirected to a new web page where you will be prompted to log in to your personal Google Drive account (as shown in Figure 4-72).

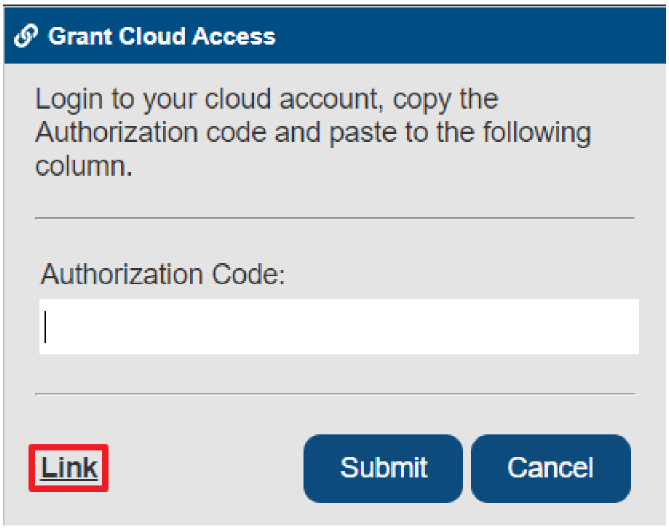


Figure 4-71 Click on the external link in the cloud authorization dialog window

Figure 4-72 Google Drive user login screen

After successfully logging in, the web page will redirect you to the user permission authorization screen (as shown in Figure 4-73). Check the desired options and click on the "Continue" button to proceed to the authorization screen.

Figure 4-73 Google Drive user permission authorization screen

Next, the web page will redirect you to the authorization code screen (as shown in Figure 4-74). Copy the authorization code from the "code" parameter in the URL and paste it into the field in the cloud authorization dialog window. Then, click on the "Execute" button (as shown in Figure 4-70).

Figure 4-74 Google Drive generates an authorization code screen

Upon successful authorization, a dialog window will appear indicating that the authorization was successful. The connection status will also be updated to "Connected" (as shown in Figure

4-75, Figure 4-76).

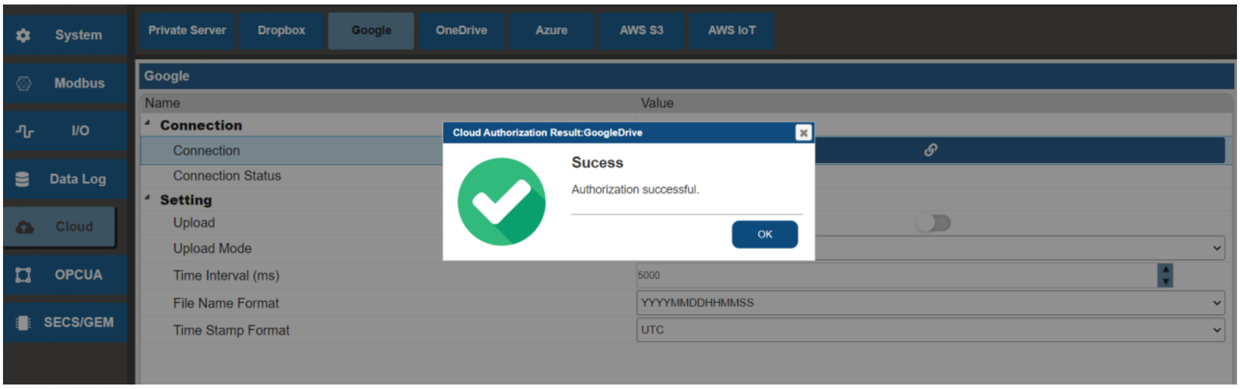


Figure 4-75 Google Drive authorization success dialog window

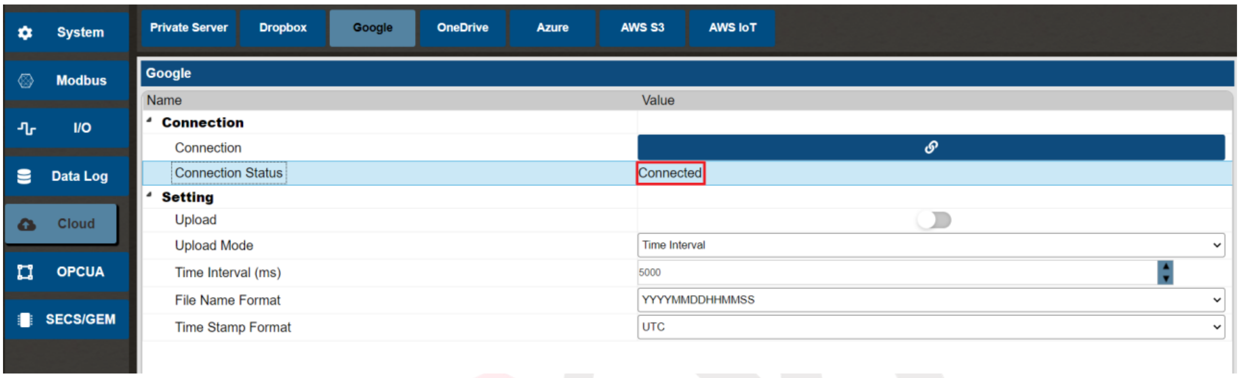


Figure 4-76 Google Drive connection status

4.1.5.4 OneDrive

To automatically upload data to OneDrive using nDAS, you will need to register an application in Azure Active Directory.

4.1.5.4.1 Application process for registering with Azure Active Directory

First, go to the homepage of the Microsoft Azure development platform and log in to your Microsoft account. Then, click on the Azure Active Directory icon (as shown in Figure 4-77).

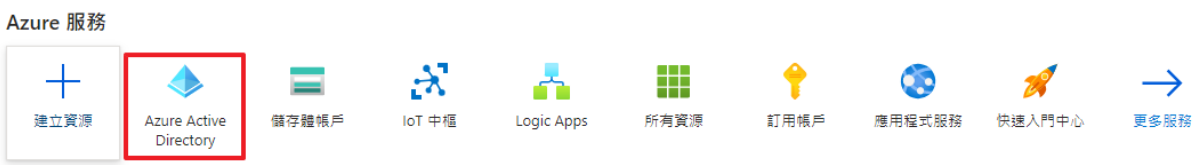


Figure 4-77 Azure Active Directory

Click the application registration menu in the menu options (as shown in Figure 4-78).

Figure 4-78 Azure Active Directory Application Registration Menu

The screen will display the registered applications in Azure Active Directory. Click on "New registration" to add a new registration (as shown in Figure 4-79).

Figure 4-79 Added registration of Azure Active Directory application

Enter the application name, for example, "nDAS6000." Select "Web" as the Redirect URI and enter **"http://localhost/module_upload_redirect_url.html"**. Finally, click on "Register" (as shown in Figure 4-80).

Figure 4-80 Application name, redirect URI settings

Note: The redirect URI must be "http://localhost/module_upload_redirect_url.html".

After adding the application, proceed to add a new secret and credential (as shown in Figure 4-81, Figure 4-82).

Figure 4-81 Application overview

Figure 4-82 Add new credential or secret

Click on "New client secret" and enter the password description and the expiration period for the password (as shown in Figure 4-83).

Figure 4-83 Client password description and password validity period

When the password is successfully added, the client secret will be displayed. Please make sure to copy and backup this value (as shown in Figure 4-84) as you will need it later.

Figure 4-84 OneDrive client password

4.1.5.4.2 Basic Setting

Please follow these steps:

1. Go to the homepage of the Microsoft Azure development platform and log in to your Microsoft account.
2. Click on the Azure Active Directory icon (as shown in Figure 4-77).
3. In the Azure Active Directory, navigate to the Registered App list and select your registered application (as in 4.1.5.4.1).
4. Copy the Tenant ID and the Client ID (as shown in Figure 4-85).

Make sure to keep the Tenant ID and Client ID secure and confidential as they are sensitive credentials.

Figure 4-85 Copy the renter ID and client ID

Click the icon to enter the OneDrive cloud service connection settings (as shown in Figure 4-86), Table 4-26 is the OneDrive connection and upload method settings.

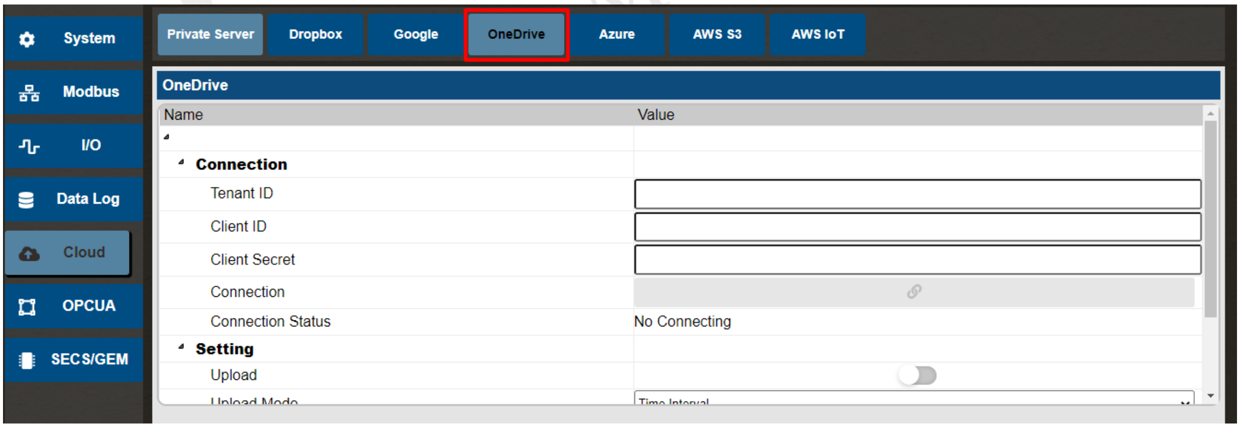


Figure 4-86 OneDrive cloud service connection settings

Table 4-26 OneDrive cloud connection setting parameters

Parameter name		Content
Connection	Rental user identifier	Enter the rental user identifier.

Settings	Client identifier	Enter the client identifier.
	Client password	Enter the client password.
	Connection	Click the button to access cloud authorization settings.
	Connection status	Display the connection status.
	File upload	Activate/Deactivate file upload.
	Upload method	Time interval/Item cycle interval.
	Time interval (in milliseconds)	Enter the time based on the upload method time interval mode.
	Item cycle interval (in quantity)	Enter the quantity based on the upload method item cycle interval mode.
	File name format	YYYYMMDDHHMMSS/YYYYMMDD timestamp format.
	Timestamp format	UTC/Local Time.

4.1.5.4.3 Connection Process

Upon registering the application in Azure Active Directory, retrieve the rental user identifier (Tenant ID), client identifier (Client ID), and client password (Client Secret) from the application information. Paste them and click the "Connect" button (as shown in Figure 4-87) to display the cloud authorization dialog window (as shown in Figure 4-88).

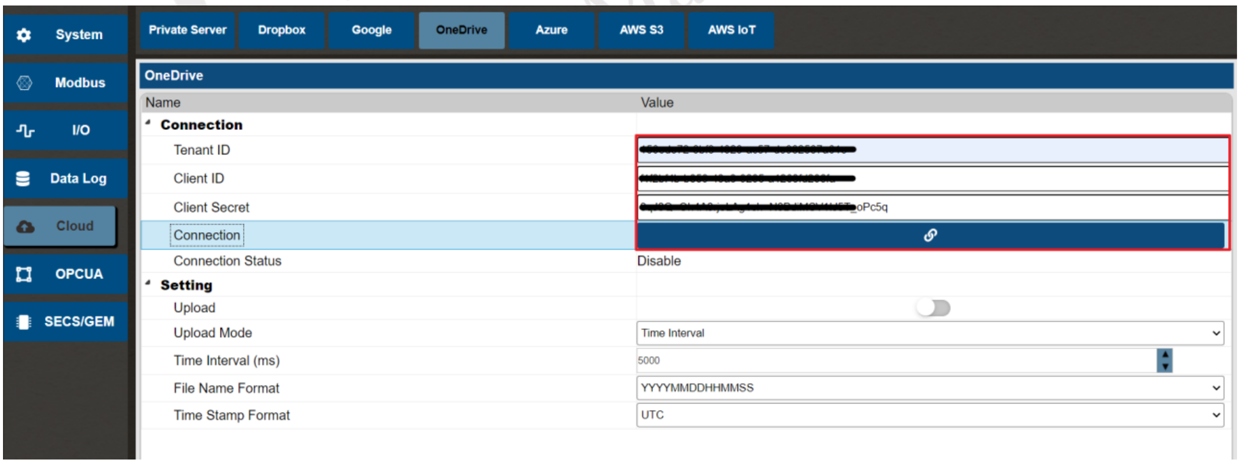


Figure 4-87 Enter the ID and password and click the connect button

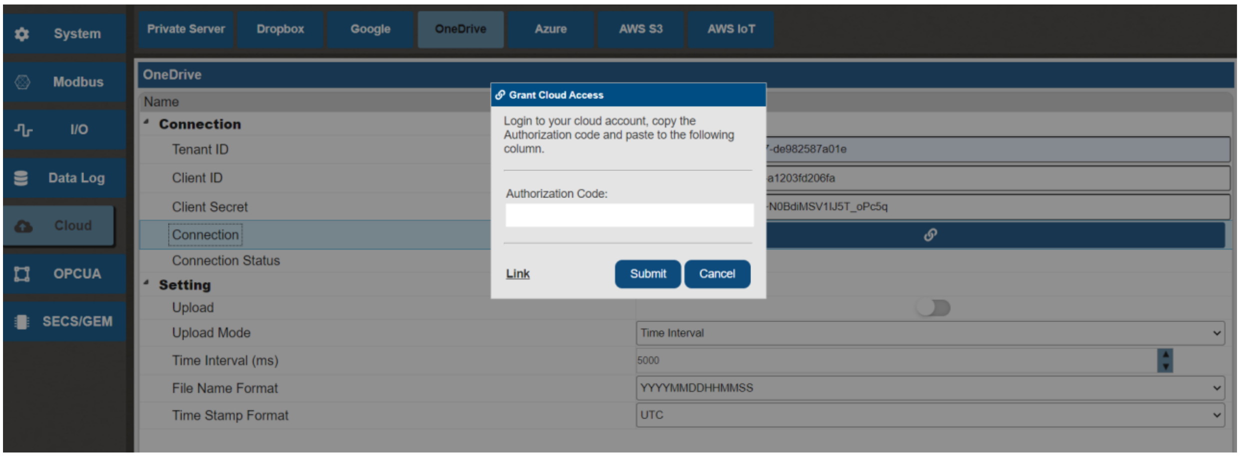


Figure 4-88 Cloud authorization dialog window

Please note that if any of the identifiers, client identifier, or client password are not entered, the "Connect" button will be disabled and cannot be clicked.

Establishing a connection requires the user to log in with their Microsoft personal account. It utilizes OAuth 2.0 authentication method, which requires the user to log in and grant authorization to nDAS for data upload to their personal cloud storage space.

Next, click the external link in the cloud authorization dialog window (as shown in Figure 4-89). Upon clicking the link, you will be redirected to a new webpage where you will be prompted to log in to your personal account (as shown in Figure 4-90).

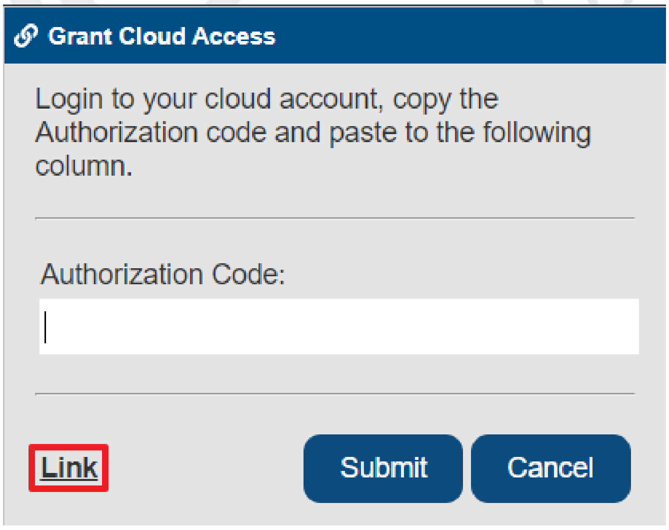


Figure 4-89 Click on the external link in the cloud authorization dialog window

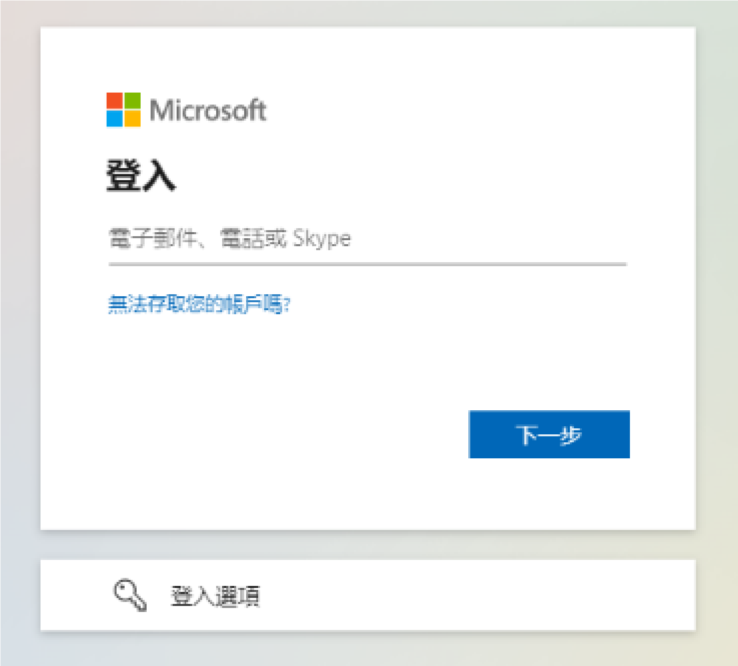


Figure 4-90 OneDrive user login screen

After a successful user login, the webpage will redirect you to the user permission authorization screen (as shown in Figure 4-91). Click "Accept" to proceed to the authorization screen.

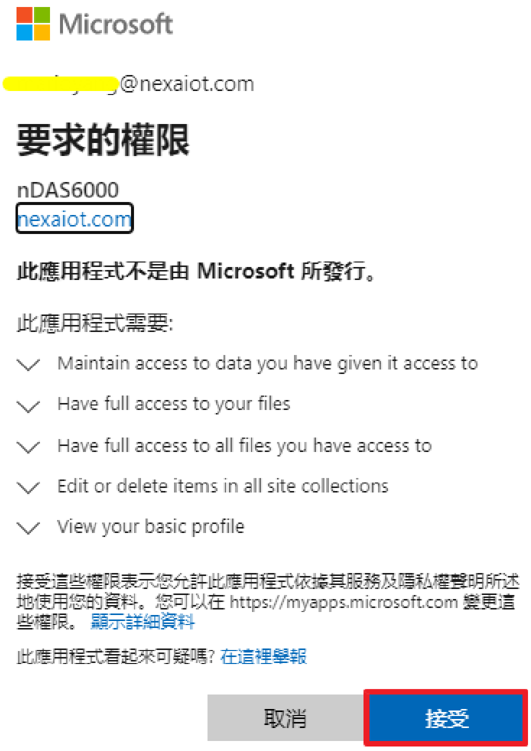


Figure 4-91 OneDriveUser Permission Authorization Screen

Next, the webpage will redirect you to the authorization code screen (as shown in Figure 4-92). Copy the authorization code from the "code" parameter in the URL and paste it into the field in the cloud authorization dialog window. Then click "Execute" (as shown in Figure 4.60).

Figure 4-92 Google Drive generates an authorization code screen

Upon successful authorization, a dialog window will appear indicating the authorization was successful. The connection status will also be updated to "Connected" (as shown in Figure 4-93, Figure 4-94).

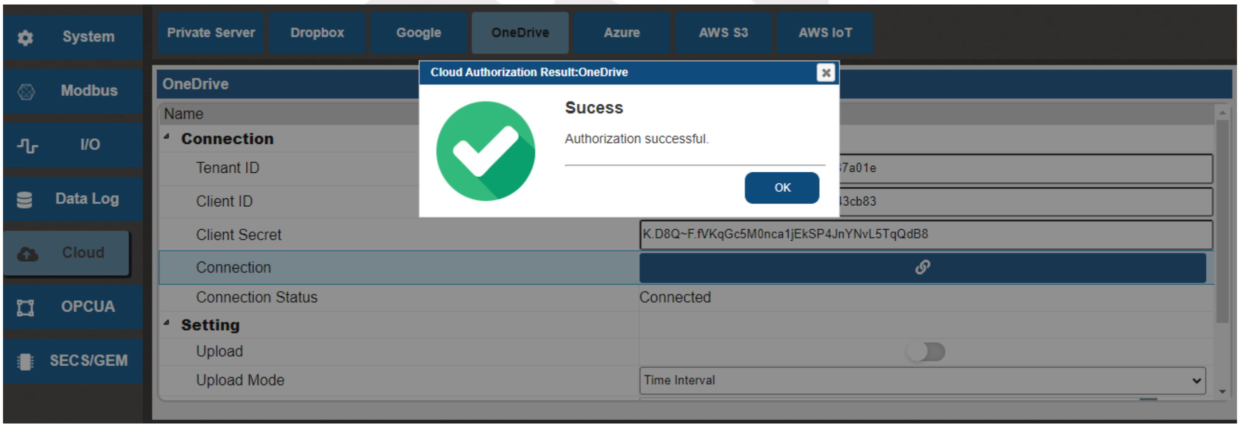


Figure 4-93 Google Drive authorization success dialog window

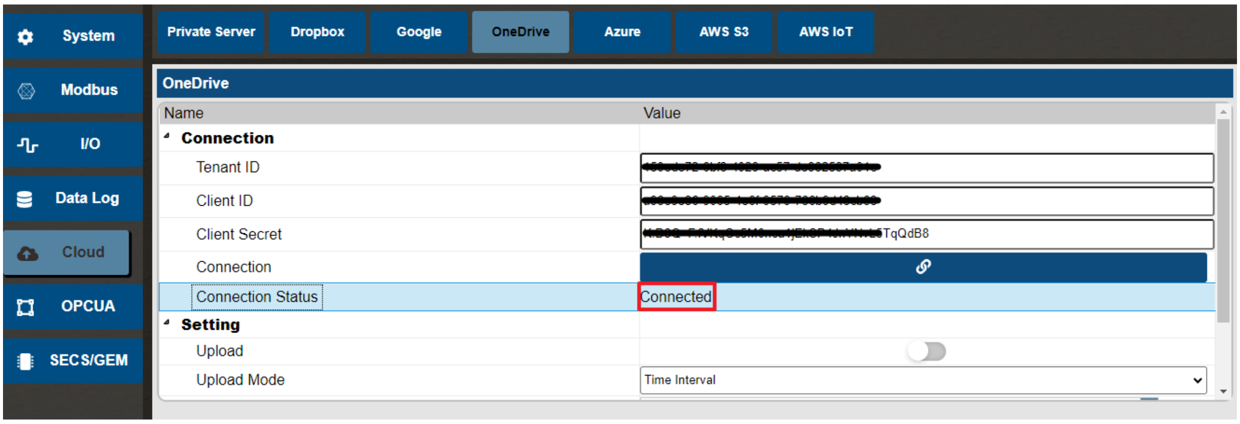


Figure 4-94 Google Drive connection status

4.1.5.5 Azure

To automatically upload data to an Azure Storage account using nDAS, you will need to establish a device connection to the Azure Storage account through Azure IoT Hub.

4.1.5.5.1 Azure IoT Hub and Azure Storage Account Setup Process

First, go to the Microsoft Azure developer portal homepage and log in with your Microsoft account. Then, click on the IoT Hub icon (as shown in Figure 4-95).

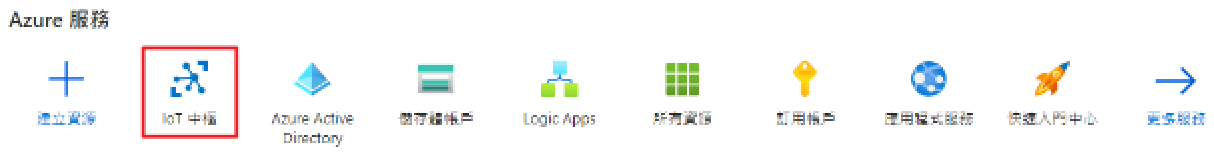


Figure 4-95 Azure IoT Hub

Enter IoT Hub and click "Create" to create an IoT Hub (as shown in Figure 4-96).

Figure 4-96 Ready to build IoT Hub

Select your Azure subscription account and enter the IoT Hub name. Choose the appropriate pricing and scale tier and click on "Review + Create" (as shown in Figure 4-97). Next, review the configuration settings and click on "Create" (as shown in Figure 4-98) to proceed.

Figure 4-97 Set IoT Hub

Figure 4-98 Confirm and build IoT Hub

Navigate to the IoT Hub endpoint that you have created. In the menu, select "Devices" and click on "New" to add a new device (as shown in Figure 4-99).

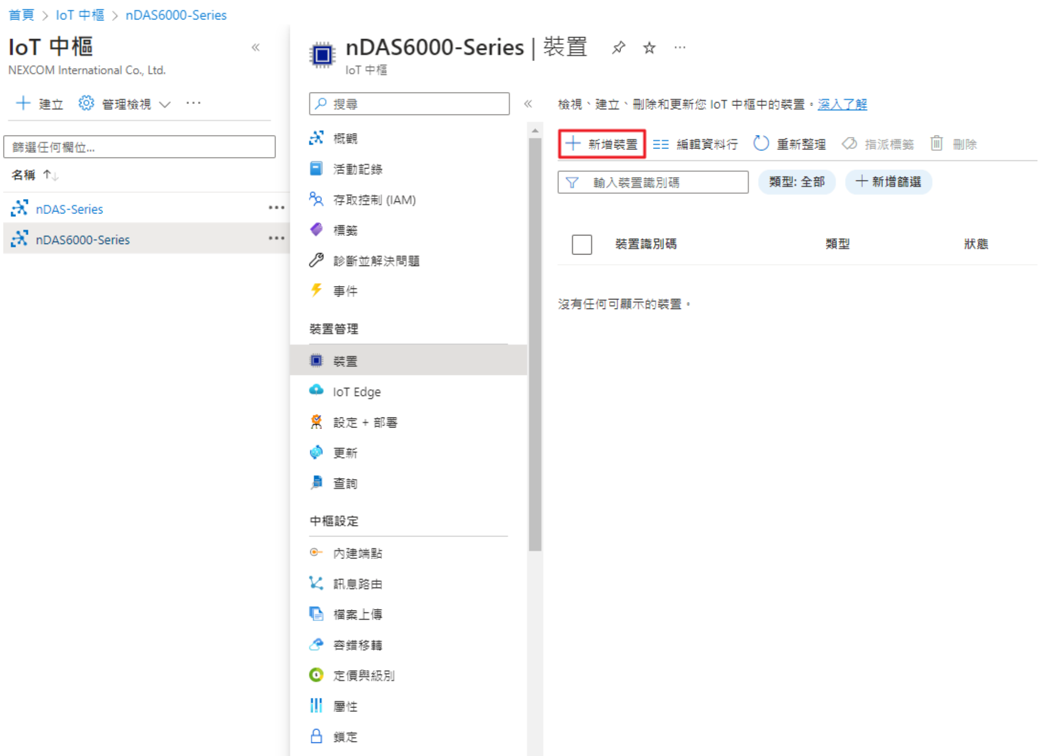


Figure 4-99 Add IoT Hub device

During the device creation process, enter a device identifier, which will be used for subsequent connection authentication. After entering the identifier, click on "Save" (as shown in Figure 4-100).

Figure 4-100 IoT Hub device identification code

Go back to the Microsoft Azure developer portal homepage and click on the Storage account icon (as shown in Figure 4-101).

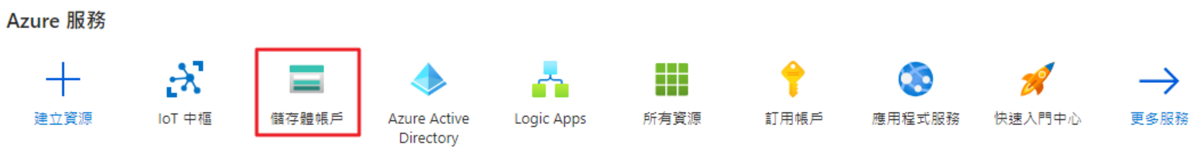


Figure 4-101 Azure storage account

Navigate to the Storage account menu and click on "Create" (as shown in Figure 4-102).

Figure 4-102 Ready to create storage account

Enter the name for the storage account and click on "Review". Wait for the review process to complete. Once the information is confirmed, click on "Create" (as shown in Figure 4-103).

Figure 4-103 Review and create storage accounts

Click the account at the end of the storage account menu (as shown in Figure 4-104).

Figure 4-104 Storage Account

Navigate to the storage account and click on the "Containers" option in the menu (as shown in Figure 4-105).

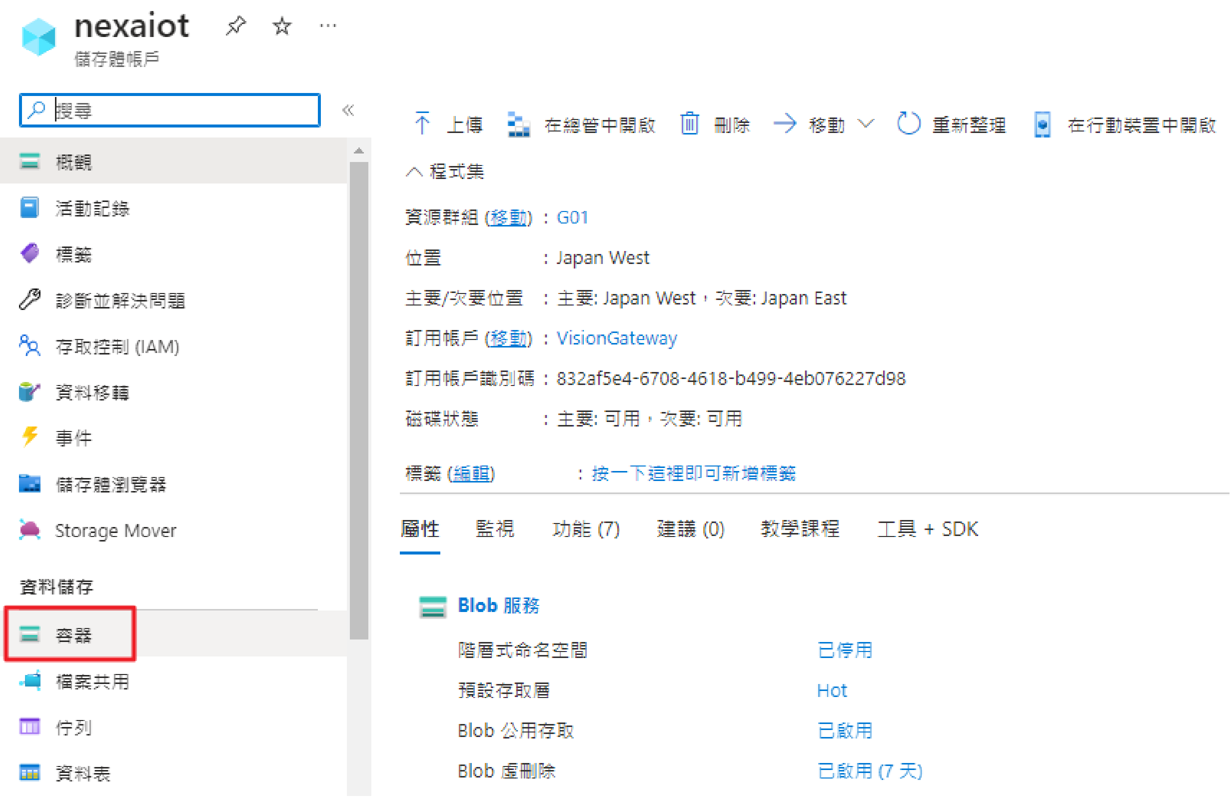


Figure 4-105 Storage account container menu

Then click to add a new container (as shown in Figure 4-106).

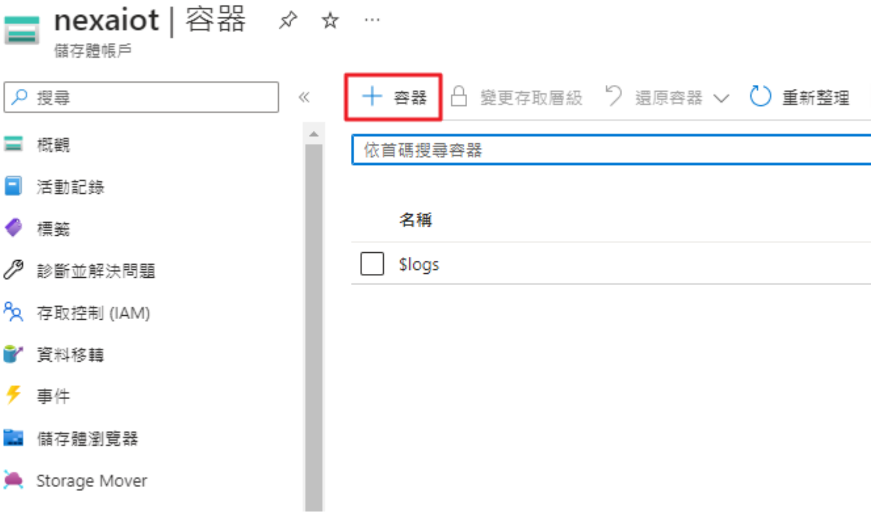


Figure 4-106 ready to build container

Enter the container name and create a new container (as shown in Figure 4-107).

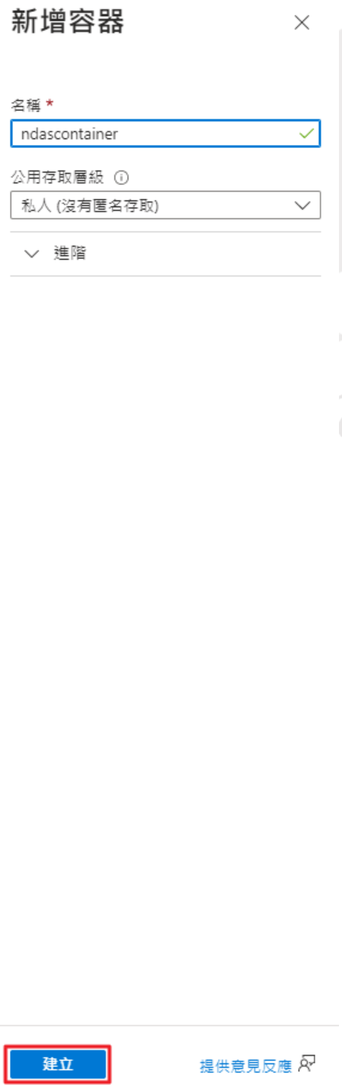


Figure 4-107 Add container

Go back to the IoT Hub menu and select "File upload" from the menu. Then click on "Select Azure Storage container" (as shown in Figure 4-108).

Figure 4-108 Configure IoT Hub file upload to storage container

Next, select the desired storage account (as shown in Figure 4-109), choose the specified container, and click on "Select" (as shown in Figure 4-110).

Figure 4-109 IoT Hub select storage account

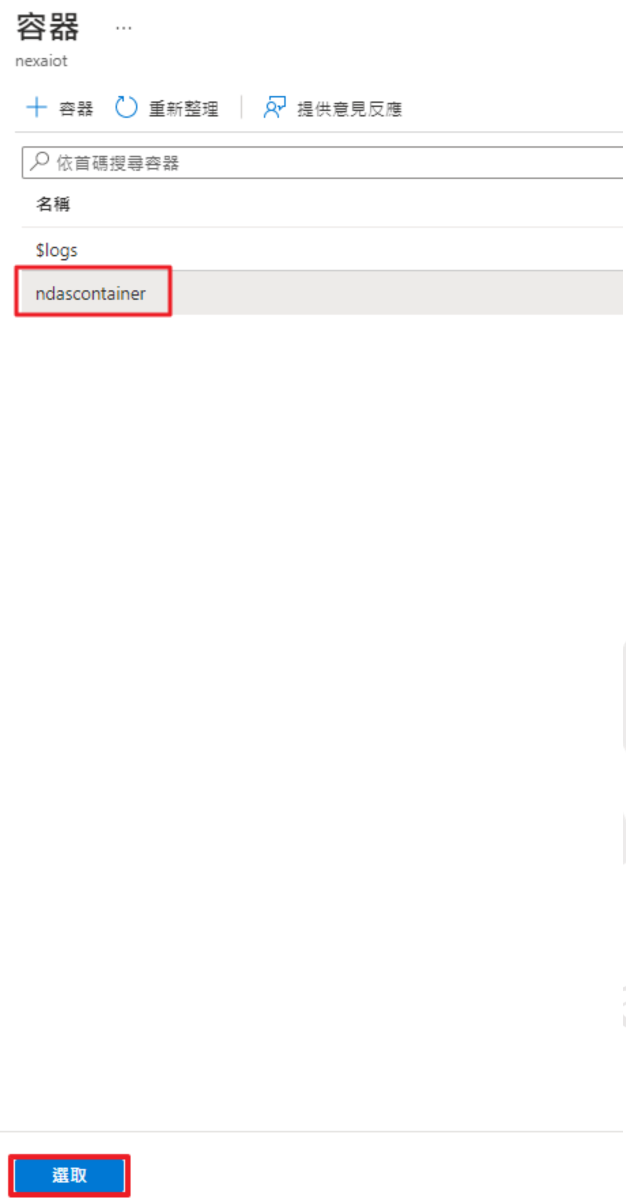


Figure 4-110 IoT Hub selects the container of the storage account

After completing the configuration for uploading files to the specified storage container, go back to the IoT Hub menu and click on "Save" (as shown in Figure 4-111).

Figure 4-111 Settings for uploading IoT Hub storage files to specified storage containers

4.1.5.5.2 Basic Setting

First, go to the Microsoft Azure developer portal homepage and log in with your Microsoft account. Then, click on the IoT Hub icon (as shown in Figure 4-95). In the IoT Hub menu, select "Shared access policies" and click on "iothubowner" (as shown in Figure 4-112). Copy the primary connection string (as shown in Figure 4-113).

Figure 4-112 IoT Hub Shared Access Policy

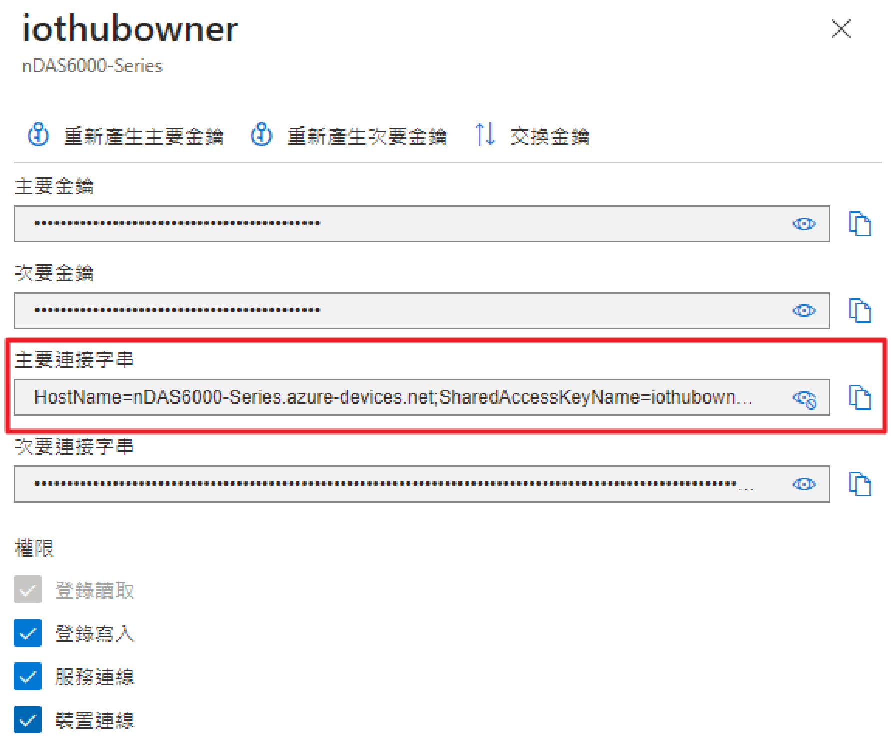


Figure 4-113 shared access policy iothubowner main connection string

Click on the icon to access Azure Cloud Service Connection Settings (as shown in Figure 4-114). Table 4-27 provides the Azure connection and upload method settings.

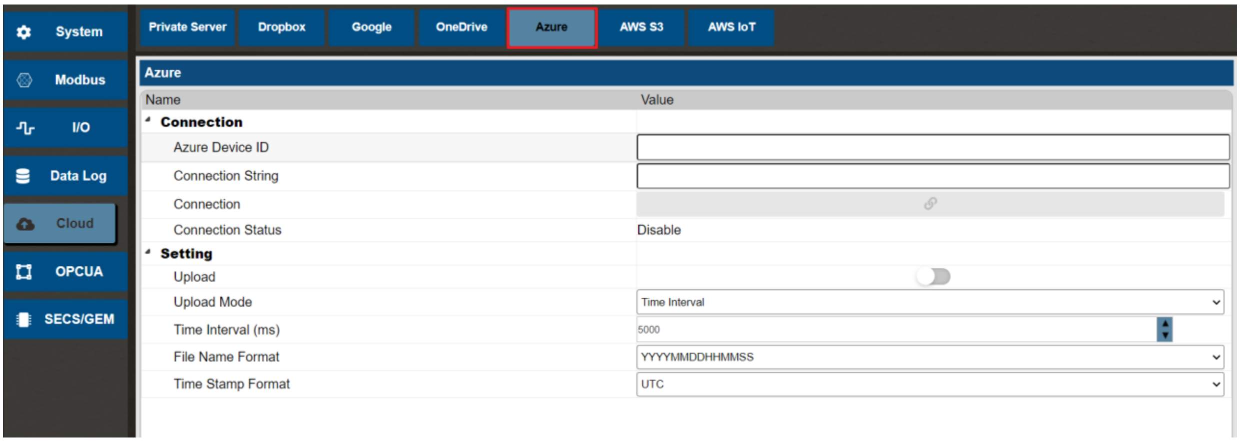


Figure 4-114 Azure cloud service connection settings

Table 4-27 Azure Cloud Connection Setting Parameters

Parameter name		Content
Connection	Azure Device Identifier	Enter the Azure device identifier.
	Connection String	Enter the connection string.
	Connection	Click the button to enter cloud authorization settings.
	Connection Status	Display the connection status.
Settings	File Upload	Enable/Disable file upload.
	Upload Method	Time interval/Item cycle interval.
	Time Interval (ms)	Enter the time based on the upload method time interval mode.
	Item Cycle Interval (quantity)	Enter the quantity based on the upload method item cycle interval mode.
	File Name Format	YYYYMMDDHHMMSS / YYYYMMDD
	Timestamp Format	UTC / Local Time.

4.1.5.5.3 Connection Process

After creating the connection endpoint in Azure IoT Hub and adding a device connected to the storage container, go to the "Devices" section in Azure IoT Hub and find the device identifier for the data to be uploaded (as shown in Figure 4-100). Enter this device identifier in the Azure Device Identifier field. In the "Shared access policies" section of Azure IoT Hub, copy the primary connection string for "iothubowner" (as shown in Figure 4-113) and paste it into the Connection String field. Finally, click on the Connect button (as shown in Figure 4-115).

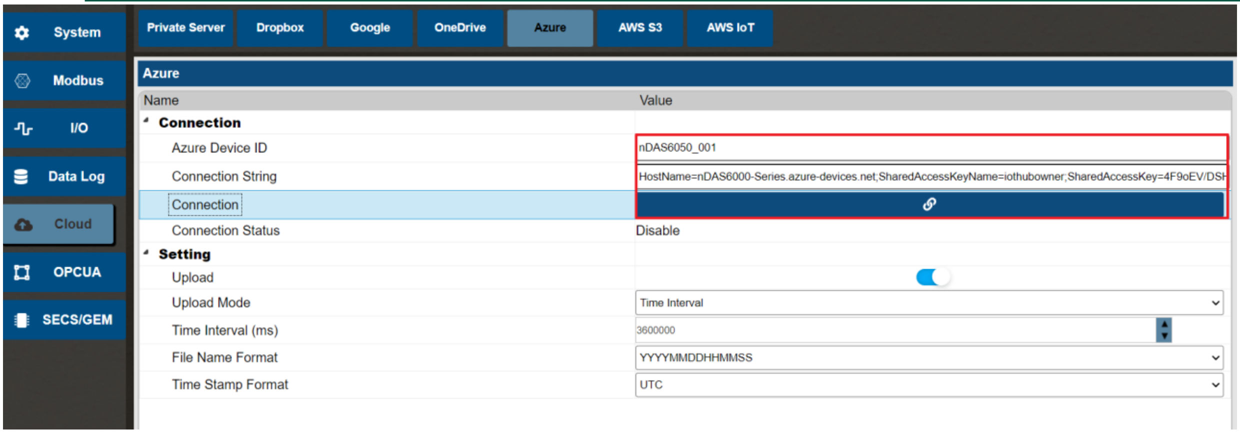


Figure 4-115 Enter the device identification code and the main connection string and click the connect button

After the verification process is completed, a dialog window will appear indicating successful authorization, and the connection status will be updated to "Connected" (as shown in Figure 4-116, Figure 4-117).

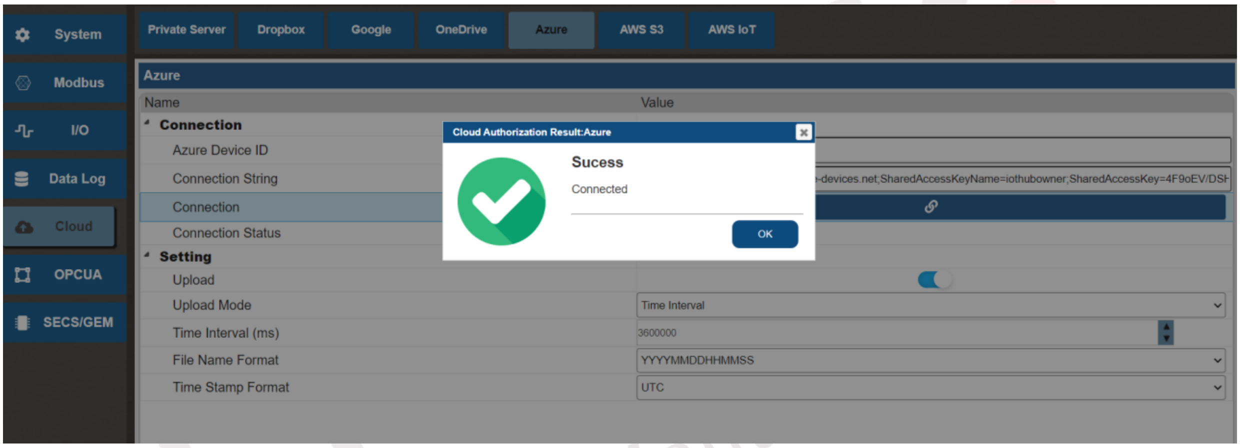


Figure 4-116 Azure connection success dialog window

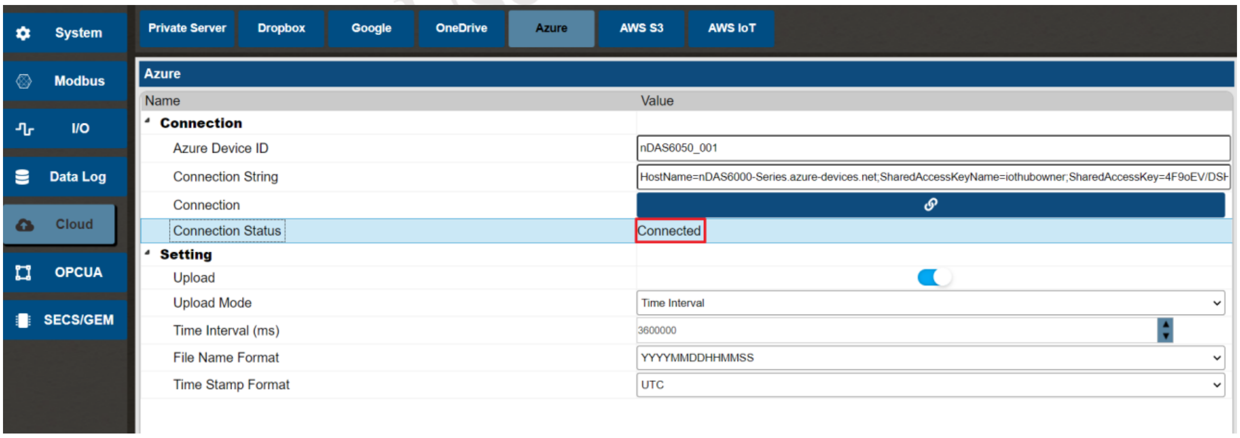


Figure 4-117 Azure connection status

4.1.5.6 AWS S3

4.1.5.6.1 AWS S3 Account Setup Process

Users need to obtain the Access Key ID and Secret Access Key for their AWS account security credentials in the AWS service (as shown in Figure 4-118).

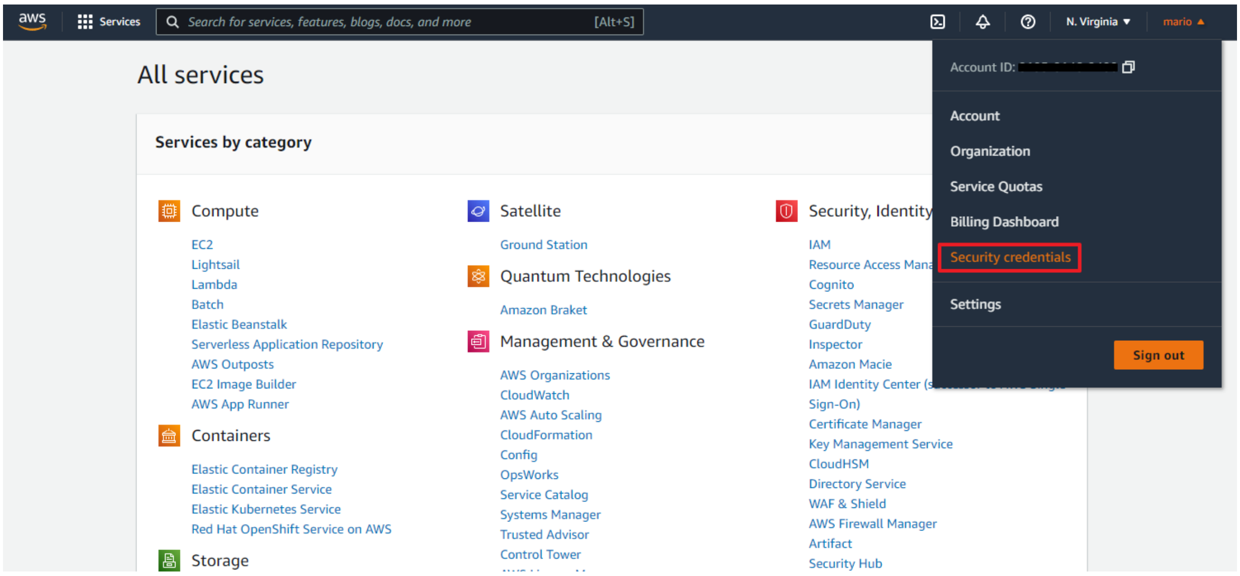


Figure 4-118 account security credentials

Next, expand the Access Keys and create a new access key (as shown in Figure 4-119, Figure 4-120).

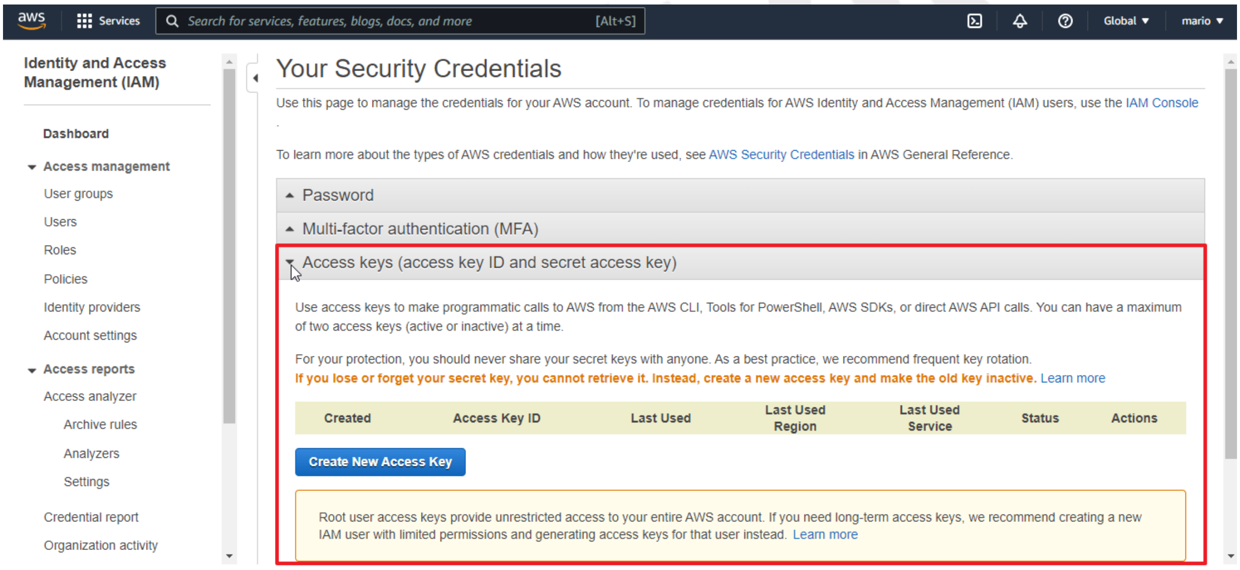


Figure 4-119 expand the Access Keys

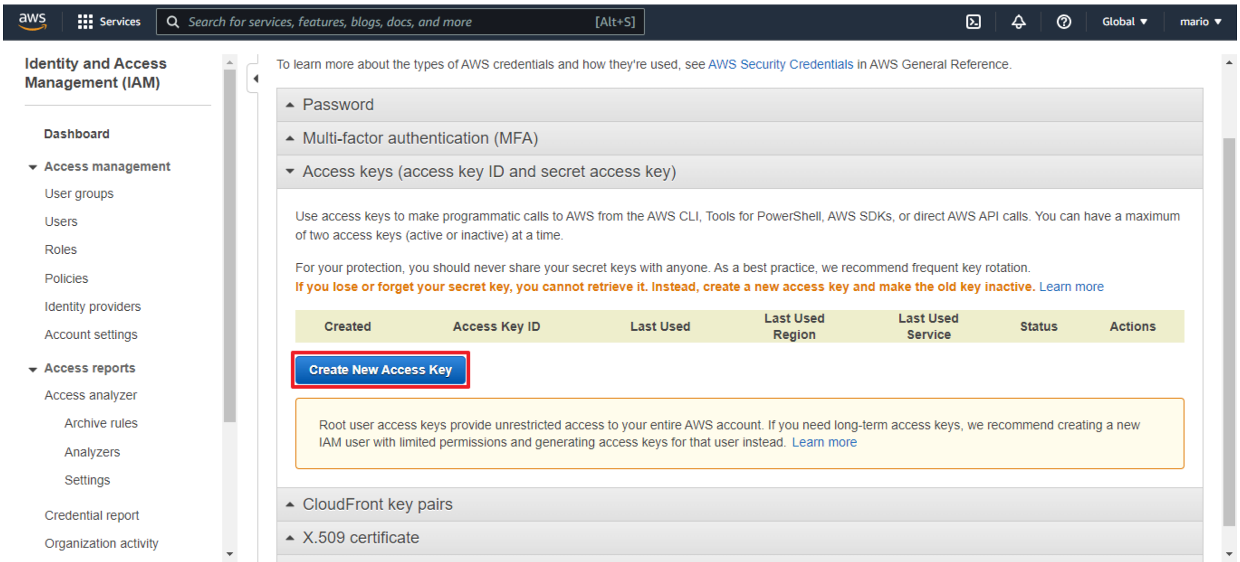


Figure 4-120 create a new access key

After clicking the button, a new set of Access Key ID and Secret Access Key will be generated. It is recommended to either backup the keys or directly download the key file (as shown in Figure 4-121). These credentials will be needed when connecting to AWS S3 later on.

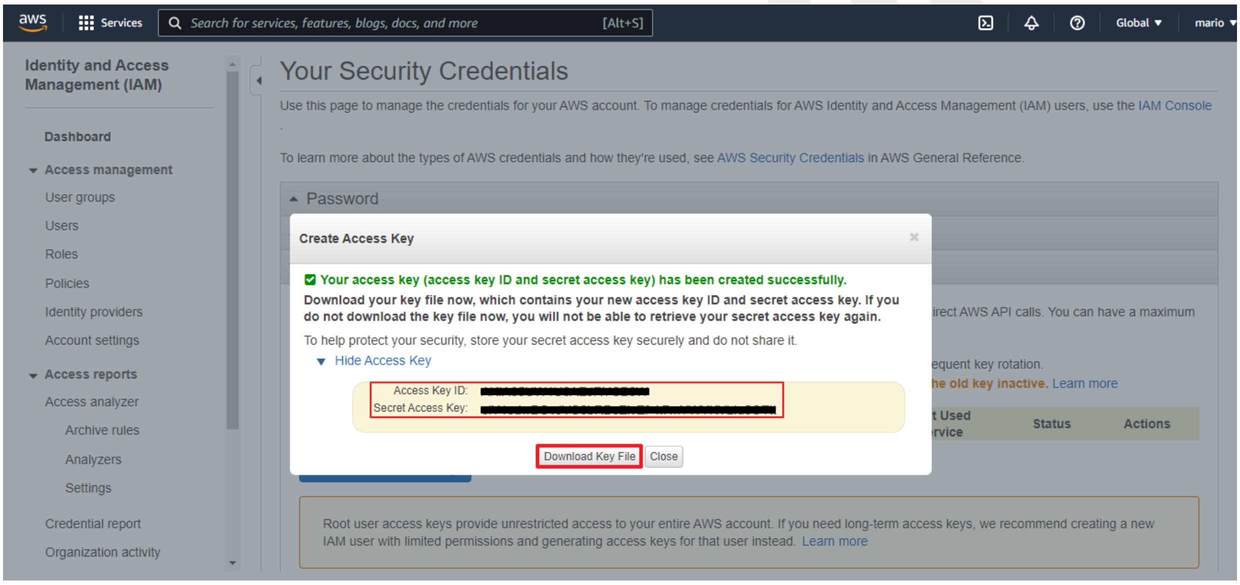


Figure 4-121 Store access key ID and private access key

Next, in the AWS cloud service, select the storage bucket feature of S3 (as shown in Figure 4-122). Click on "Create Bucket" (as shown in Figure 4-123) and follow the prompts to select and enter the desired attributes for the storage bucket. This will complete the process of creating the storage bucket based on the user's requirements.

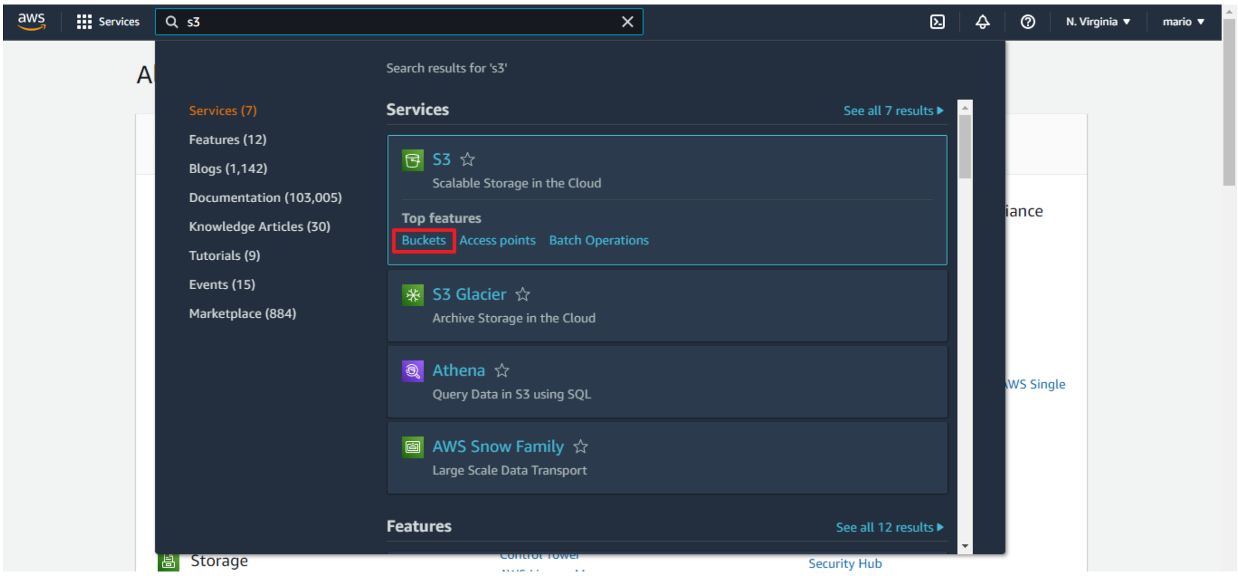


Figure 4-122 AWS S3 storage

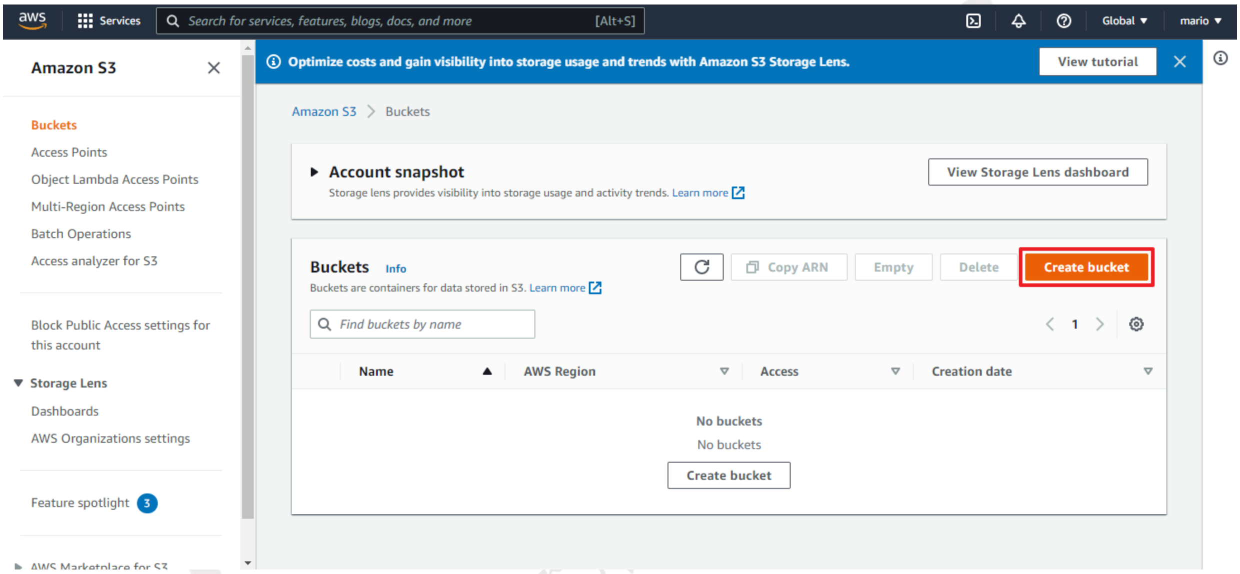


Figure 4-123 Create Storage

4.1.5.6.2 Basic Setting

Click on the icon to access the AWS S3 cloud service connection settings (as shown in Figure 4-124). Table 4-28 provides the settings for AWS S3 cloud connection and upload methods.

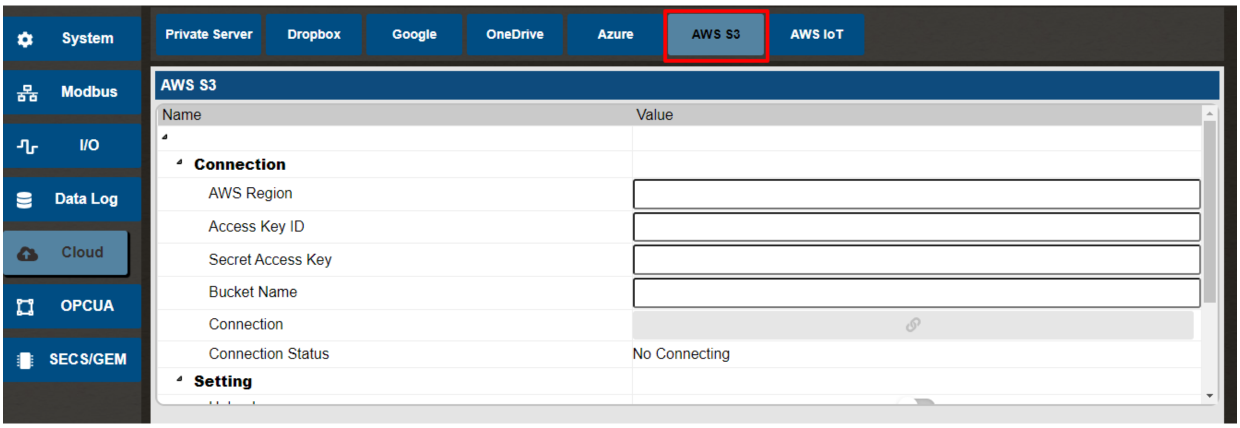


Figure 4-124 AWS S3 cloud service connection settings

Table 4-28 AWS S3 cloud connection setting parameters

Parameter name		Content
Connection	AWS Region	Enter the AWS region.
	Access Key ID	Input the access key ID.
	Confidential Access Key	Input the confidential access key.
	Storage Container Name	Input the name of the storage container.
	Connection	Click the button to access cloud authorization settings.
	Connection Status	Display the connection status.
Settings	File Upload	Activate/Deactivate file uploading.
	Upload Method	Time interval/Periodic interval for the project.
	Time Interval (milliseconds)	Enter the time based on the upload method time interval mode.
	Periodic Interval (quantity)	Enter the quantity based on the upload method periodic interval mode.
	File Name Format	YYYYMMDDHHMMSS/YYYYMMDD format.
	Timestamp Format	UTC/Local Time.

4.1.5.6.3 Connection Process

After adding a storage container in AWS S3, kindly copy and paste the following information:

- AWS Region
- Access Key ID
- Secret Access Key
- Bucket Name

Once you have pasted the information, click on the "Connect" button (as shown in Figure 4-125).

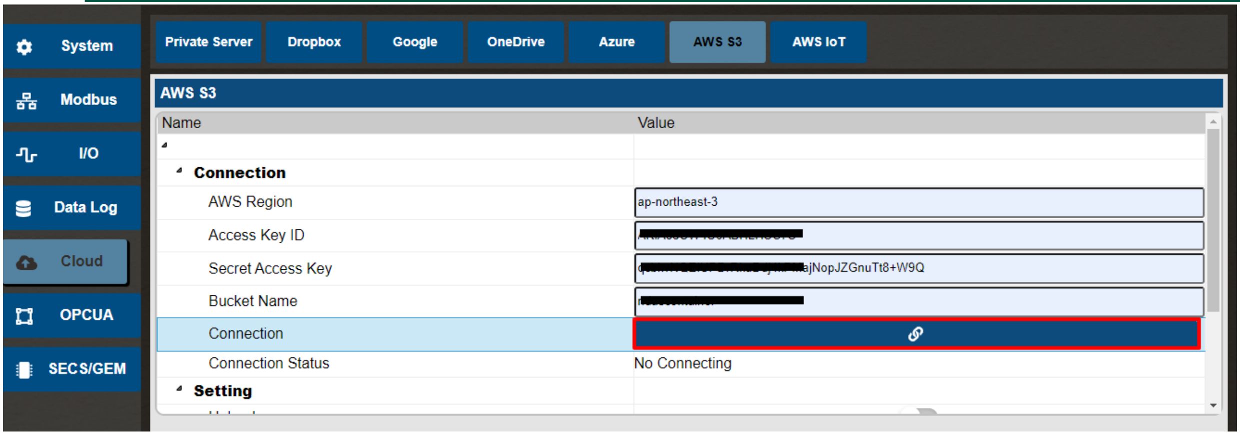


Figure 4-125 Enter the AWS S3 connection information and click the connect button

After the verification is completed, a dialogue window will appear confirming the successful authorization. The connection status will also be updated to "Connected" (as shown in Figure 4-126 and Figure 4-127).

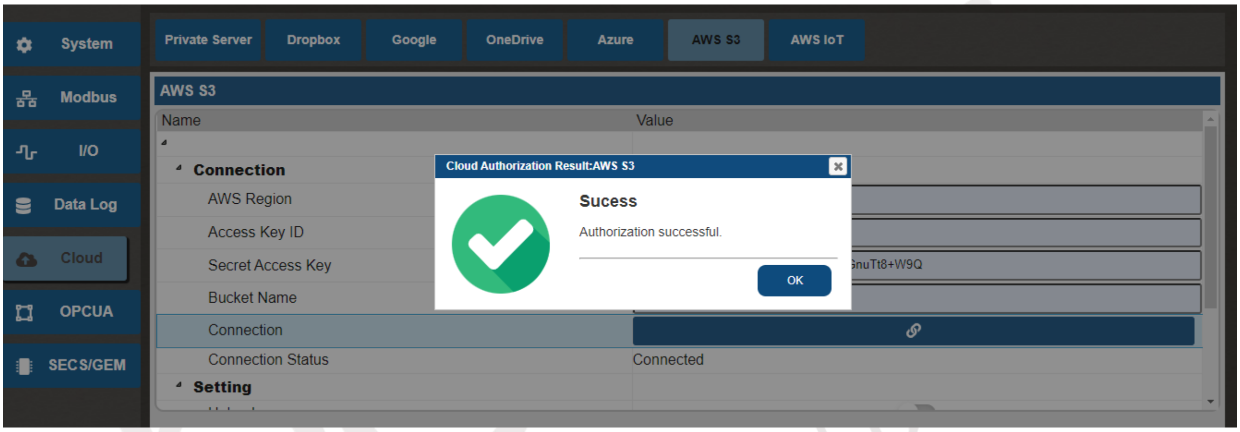


Figure 4-126 AWS S3 connection success dialog window

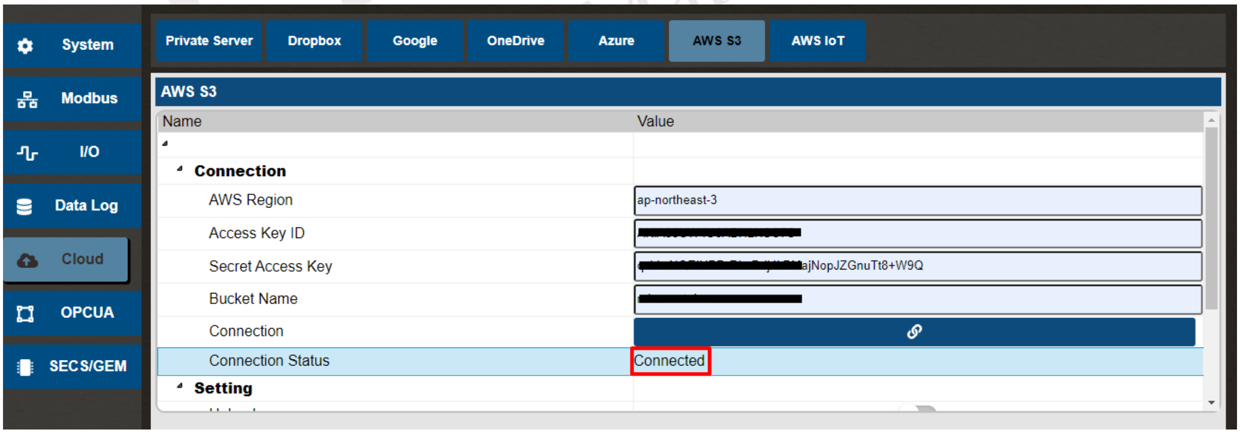


Figure 4-127 AWS S3 connection status

4.1.5.7 AWS IoT

4.1.5.7.1 AWS IoT Account Setup Process

After logging in to the AWS management console, refer to the following operation process.

4.1.5.7.1.1 Create Thing, Certificate of Thing, Policy of Thing

1. Click Services to enter the IoT Core entrance (as shown in Figure 4-128).

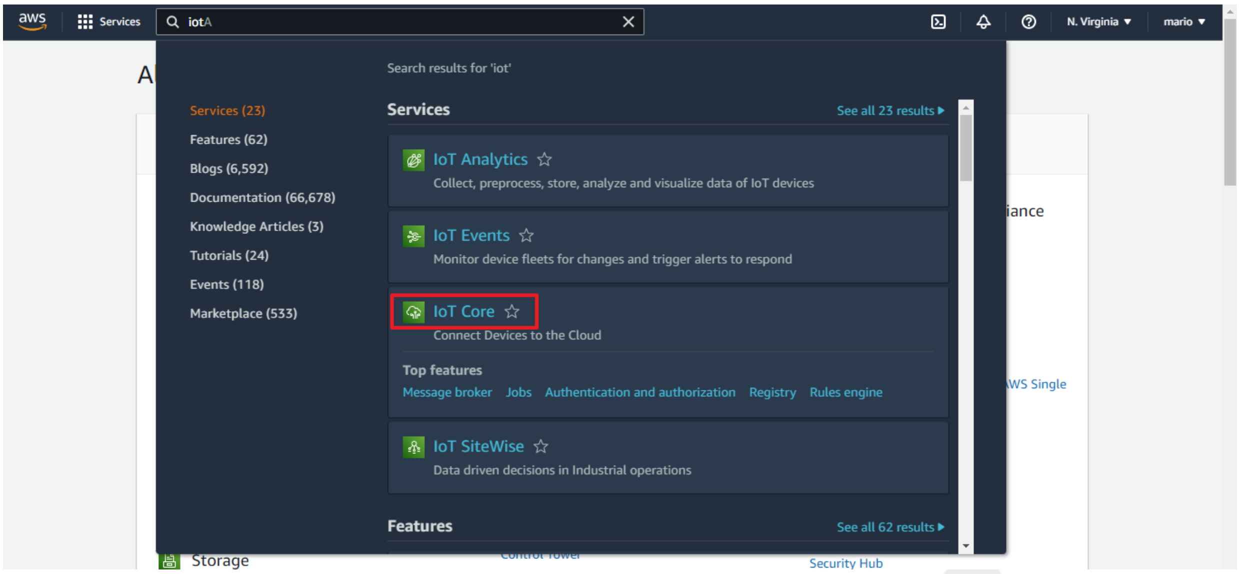


Figure 4-128 IoT Core Endpoint

2. Next, expand the "Manage" section on the left-hand side of the dashboard and click on "Things" to proceed with creating a Thing as instructed (as shown in Figure 4-129).

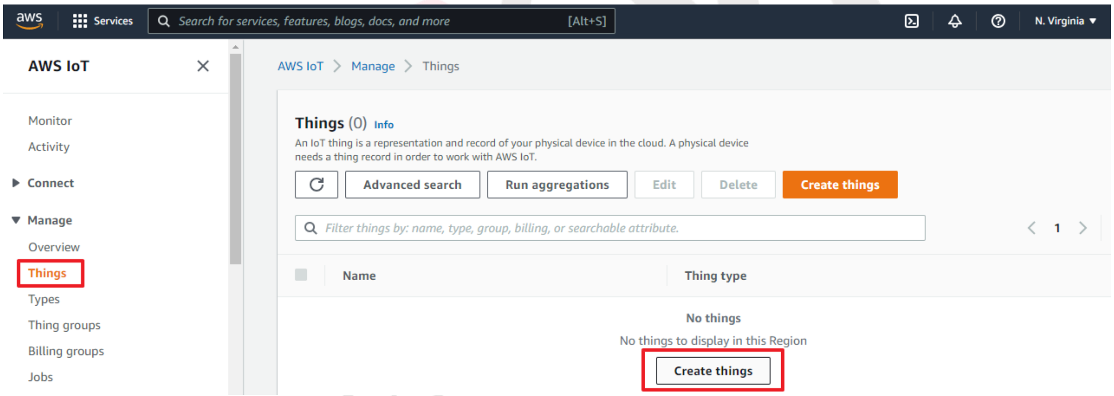


Figure 4-129 Ready to Create Thing

3. When reaching step 3 of creating a Thing, you will need to choose a Policy. If you don't have a Policy available, please follow the instructions and navigate to "Policies" first to create a Policy (as shown in Figure 4-130).

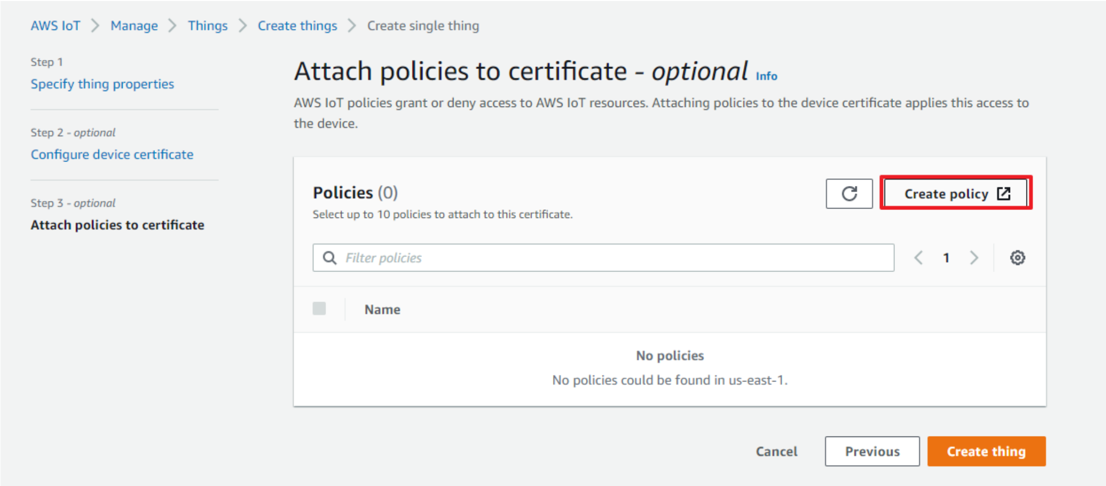


Figure 4-130 Ready to Create Policy

4. Based on your specific requirements, please fill in the options for the IoT Core actions and Policy resources you need. In this example, we will demonstrate granting full permissions, and please ensure to select "Allow" as the Policy effect. Once you have made your selections, click on "Create" (as shown in Figure 4-131).

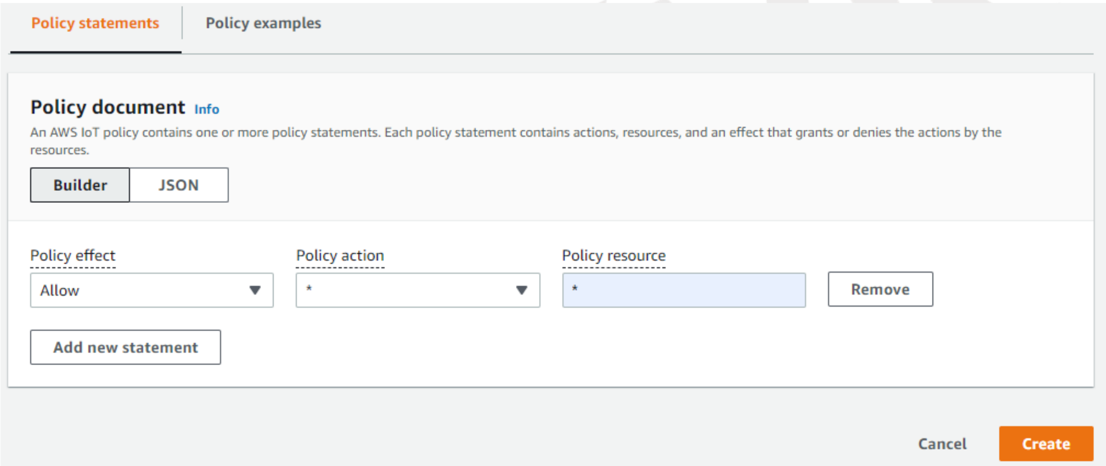


Figure 4-131 Create Policy

5. Returning to the incomplete Thing screen, you will now see the newly created Policy. Select the desired Policy and proceed to create the Thing (as shown in Figure 4-132).

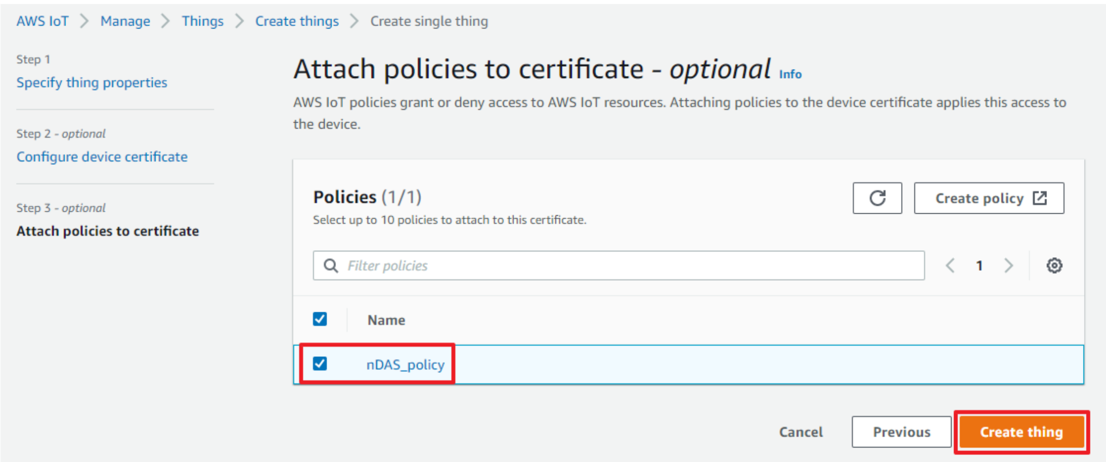


Figure 4-132 Complete the creation of Thing

6. Next, an automatic prompt will appear to download the certificates. Download all the certificates and click "Done" to complete the Thing creation (as shown in Figure 4-133). You can then verify the certificates by expanding the "Secure" section on the left-hand side of the dashboard, selecting "Certificates," and confirming the presence of the newly created certificates. By clicking on a certificate, you can view the associated Thing and Policy. **It is recommended to delete and recreate the Thing, Policy, and certificates if any modifications to the Policy or Thing are required.**

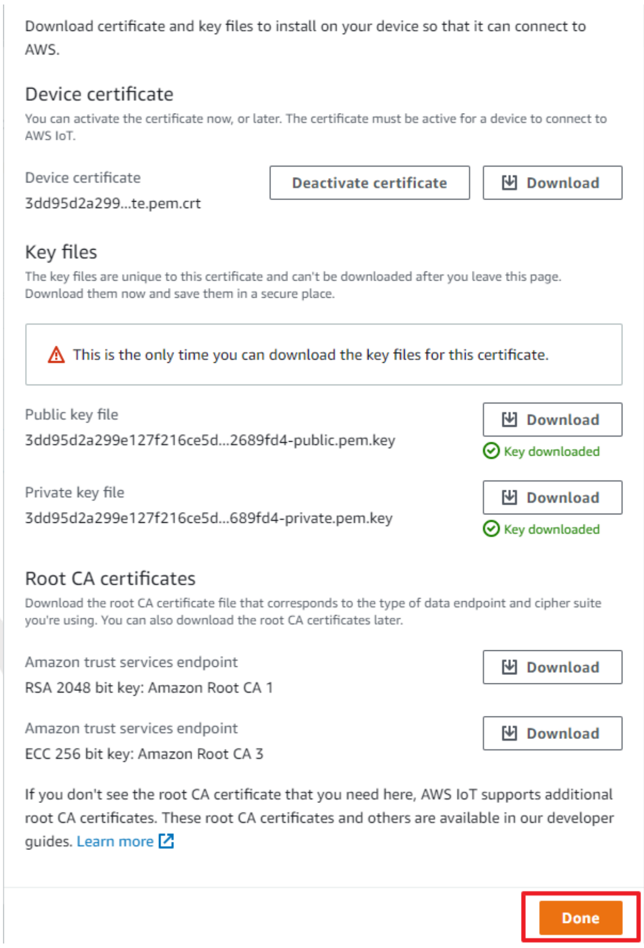


Figure 4-133 Download the Certificate of Thing

4.1.5.7.1.2 Establish Rules for IoT Core to Communicate with S3

1. Within the IoT Core portal, expand the "Act" section on the left-hand side of the dashboard, and click on "Rules" to proceed with creating a rule as instructed (as shown in Figure 4-134).

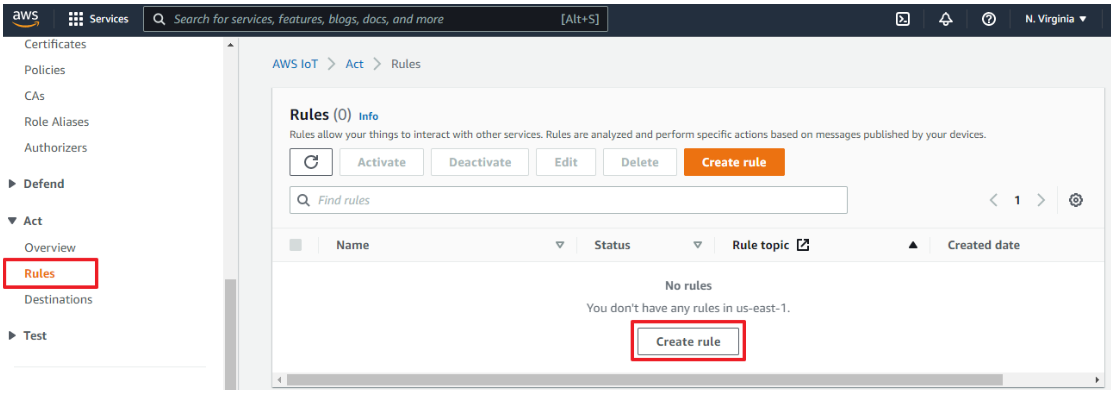


Figure 4-134 Ready to create Rule

2. During step 2 of creating a Rule, you need to enter an SQL statement. This statement determines which messages from the subscribed Topic in IoT Core will be affected and whether to add filters to extract specific information as needed (as shown in Figure 4-135).

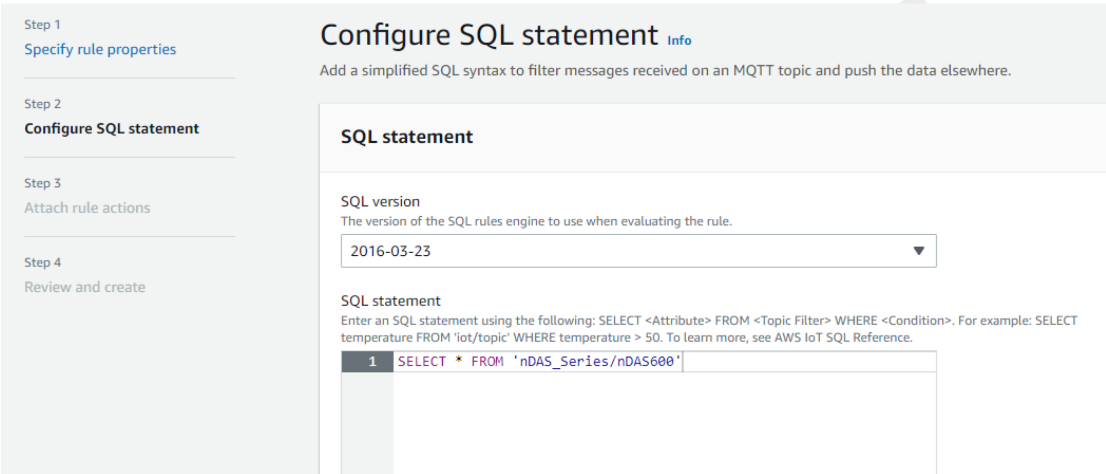


Figure 4-135 Create Rule-Configure SQL statement

3. During step 3 of creating a Rule, you need to choose an action. In this case, you must select the "S3 bucket" option (as shown in Figure 4-136).

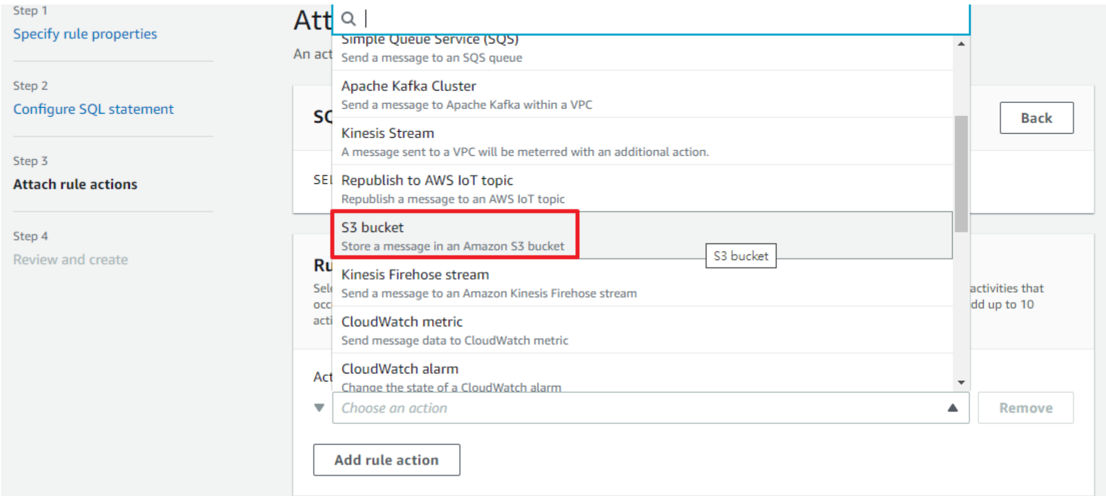


Figure 4-136 Create Rule –Setup Rule Action

4. During step 3 of creating a Rule, after selecting the "S3 bucket" action, you will need to choose the specific S3 Bucket where you want to store the data (as shown in Figure 4-137). If you haven't created the Bucket beforehand, you will need to navigate to the entry point of the S3 service and create the Bucket there (as shown in Figure 4-138).

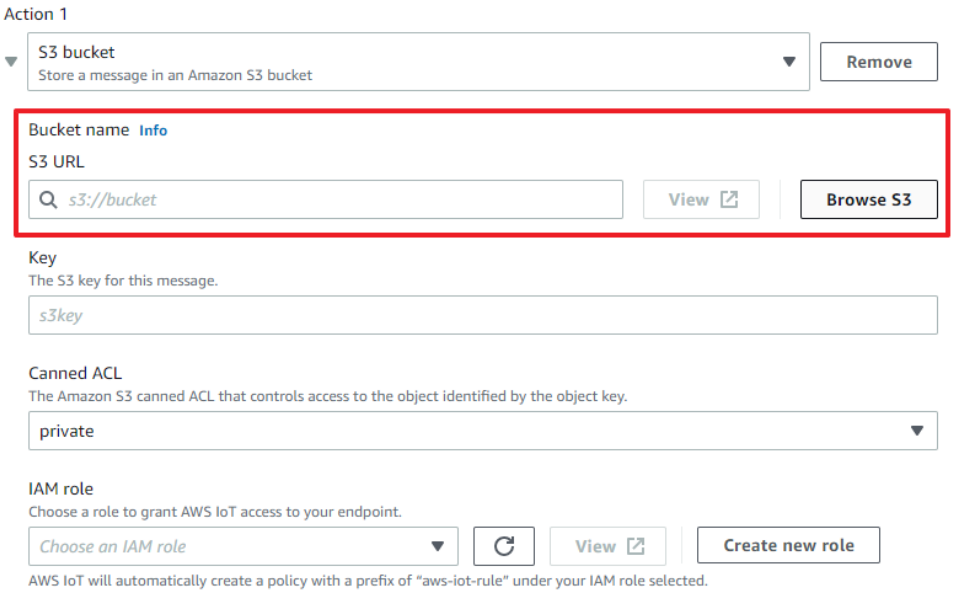


Figure 4-137 Create Rule - Select S3 Bucket in action

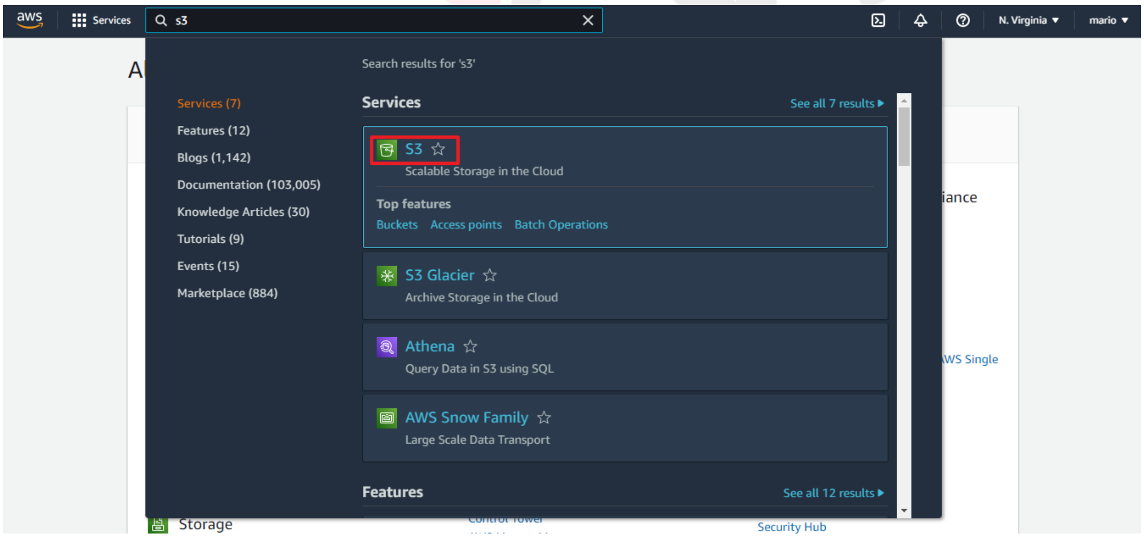


Figure 4-138 AWS S3 Endpoint

5. Upon accessing the AWS S3 service endpoint, click on "Create bucket" and proceed to create the Bucket as instructed (as shown in Figure 4-139).

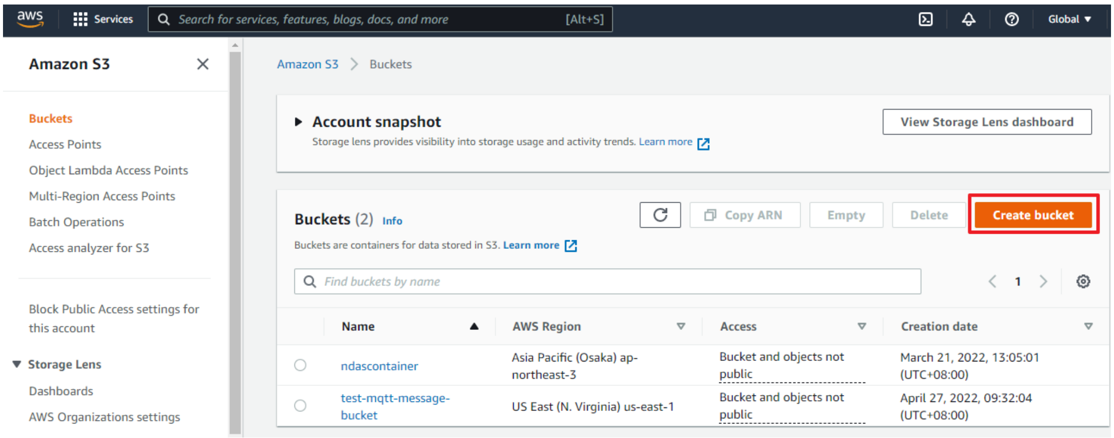


Figure 4-139 create storage

6. Please note that it is important to ensure that the AWS Region of the Bucket matches the Region of IoT Core (as shown in Figure 4-140 and Figure 4-141).

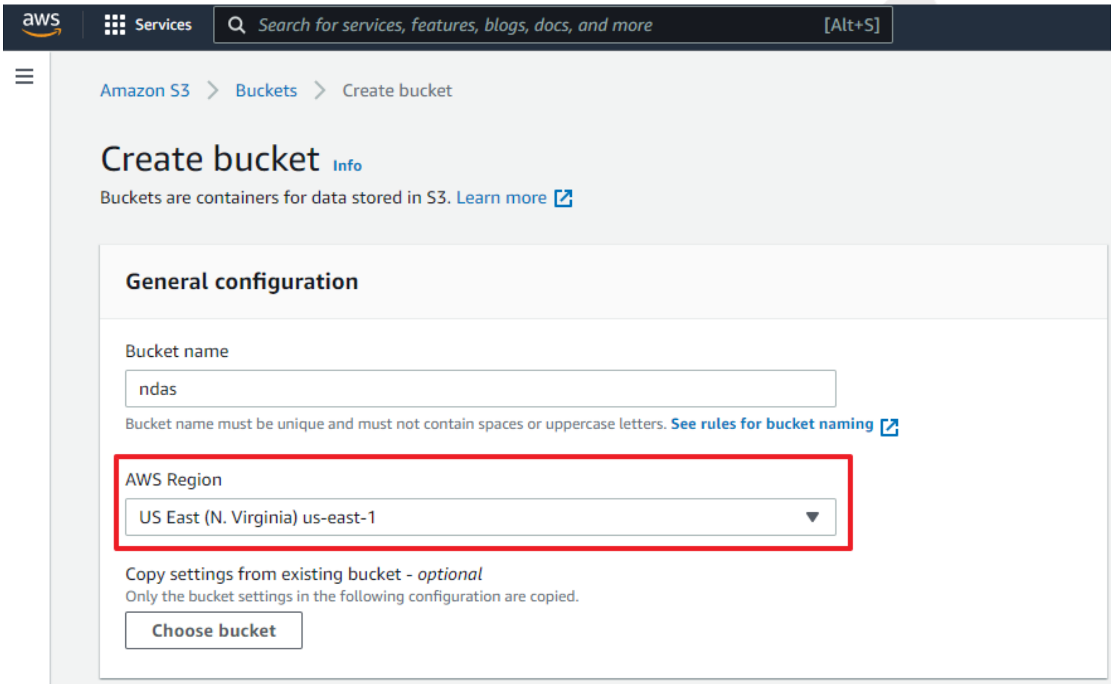


Figure 4-140 AWS Region in Bucket

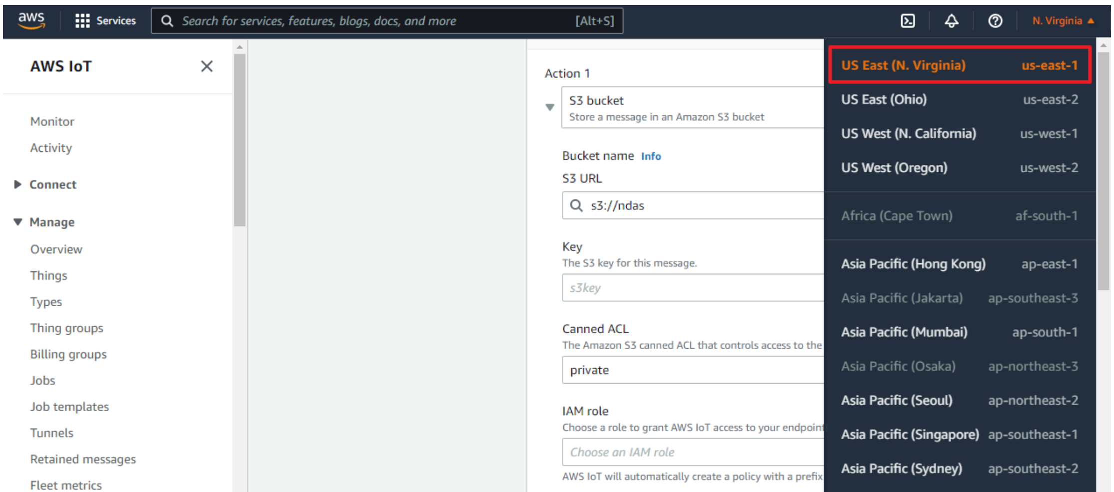


Figure 4-141 AWS IoT Core Region

7. Continuing with step 3 of creating the Rule, after selecting the completed bucket, you need to enter a Key (as shown in Figure 4-142). The Key represents the file name of the data stored in S3. Here is an example code for the Key:

${topic()}/${parse_time("yyyy", timestamp())}${parse_time("MM", timestamp())}${parse_time("dd", timestamp())}/${timestamp()}.csv

The path of the object seen in S3 will be shown as follows:

nDAS_Series/nDAS600/20220101/1641041999000.csv

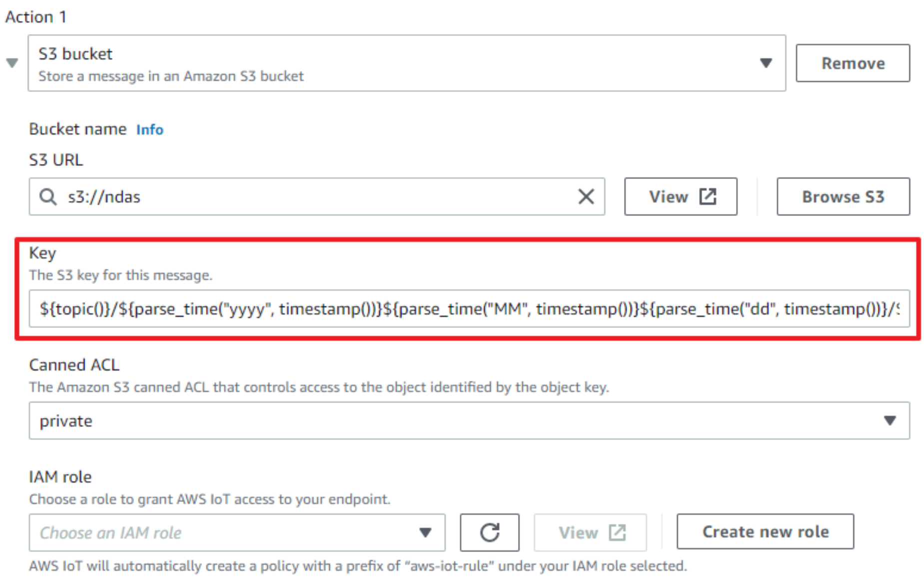


Figure 4-142 Enter Action Key

8. During step 3 of creating the Rule, you need to choose an IAM role responsible for writing data to the S3 Bucket. If you haven't created the role beforehand, you can select "Create new role" and enter a name for the role (as shown in Figure 4-143 and Figure 4-144).

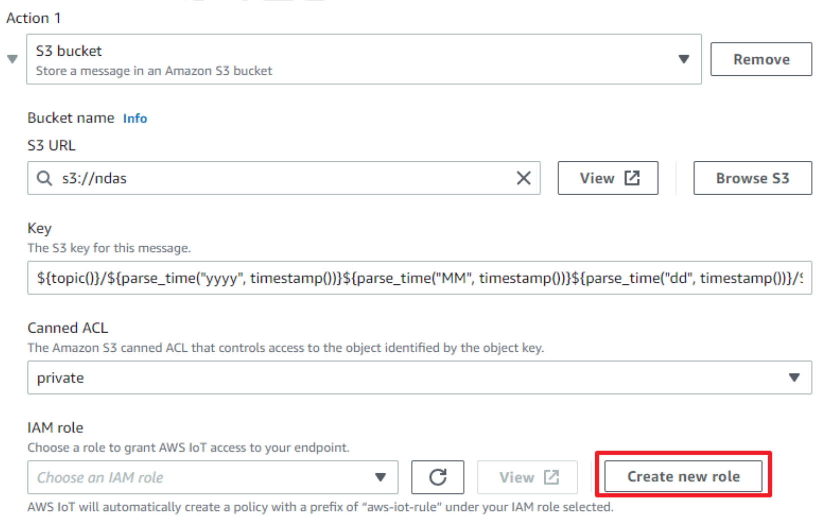


Figure 4-143 Ready to create IAM role

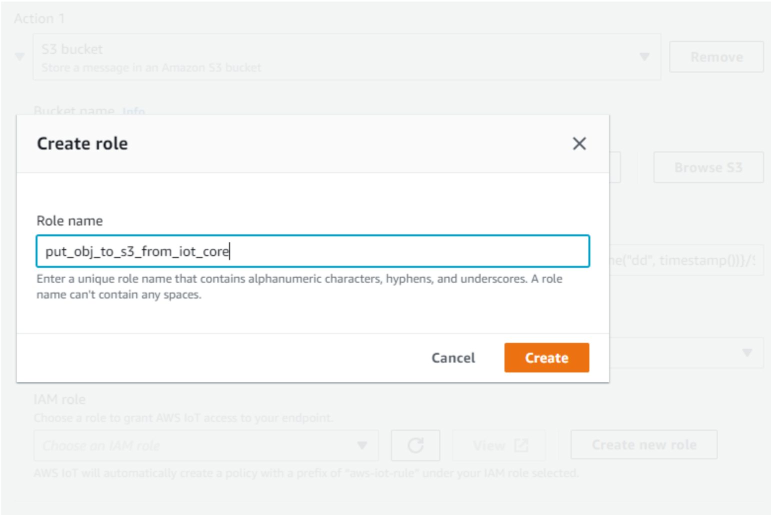


Figure 4-144 Enter name of IAM Role

Please note that the Role requires a Policy in order to have functionality. You will need to navigate to the IAM entry point separately to create the required Policy for the Role.

9. After completing the Action, click on "Next" to proceed. The next step is to review the Rule, ensure everything is correct, and then click on "Create" to finalize the creation (as shown in Figure 4-145).

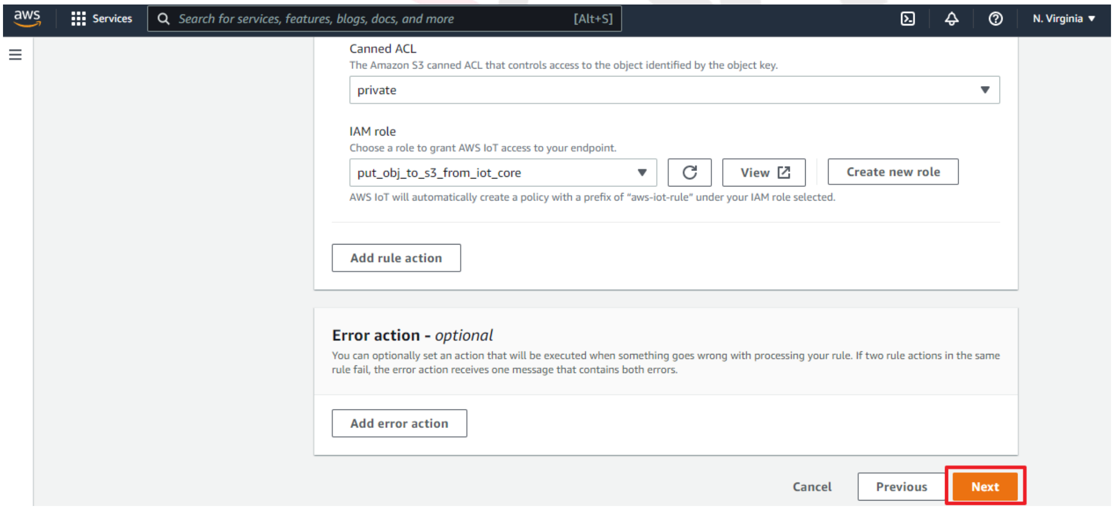


Figure 4-145 Check and finish creating Action

7. Click Services to enter the IAM entry (as shown in Figure 4-146).

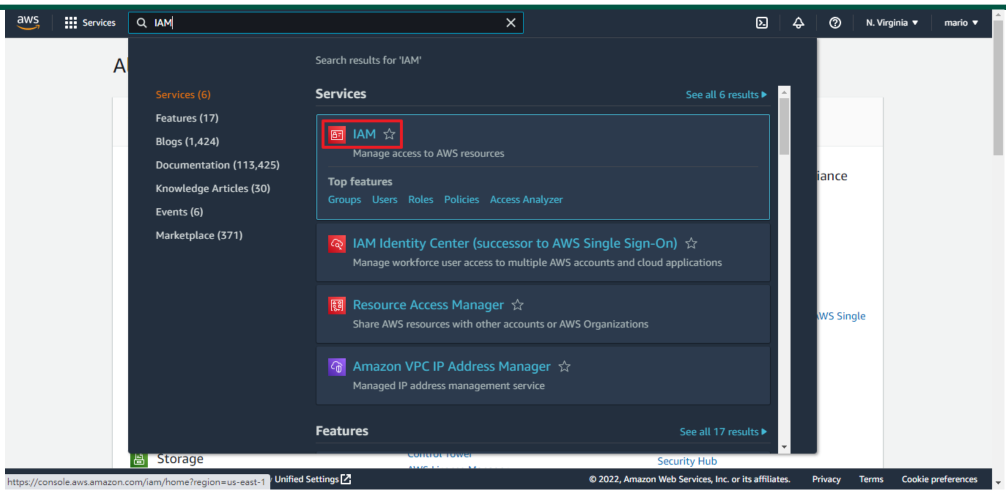


Figure 4-146 IAM Endpoint

10. Upon accessing the IAM entry point, expand the "Access management" section on the left-hand side of the dashboard and click on "Roles." Here, you will be able to see the Role that was created earlier (as shown in Figure 4-147).

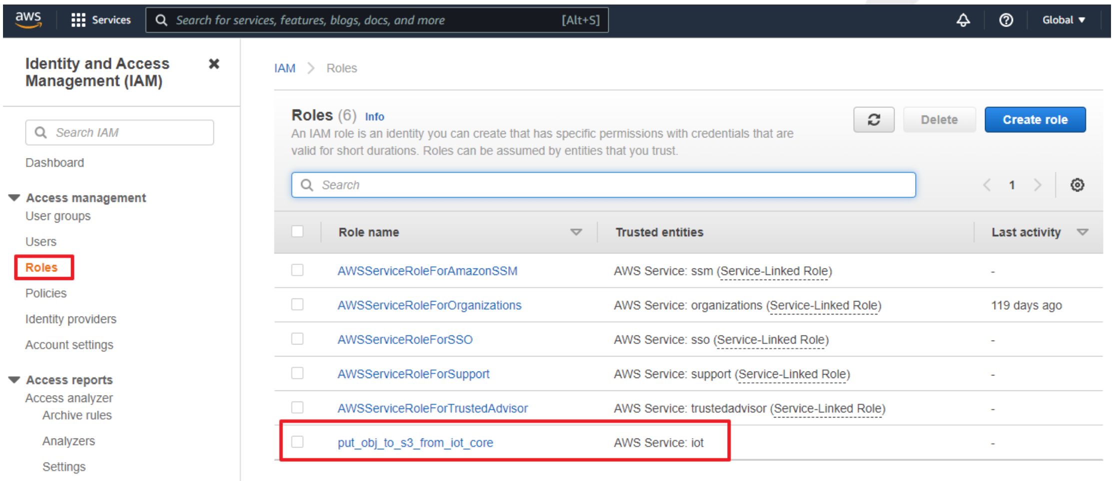


Figure 4-147 Check IAM Role

11. Click on the "Policies" option on the left-hand side of the dashboard and proceed to create a Policy for "Put Object" operation to the S3 Bucket (as shown in Figure 4-148).

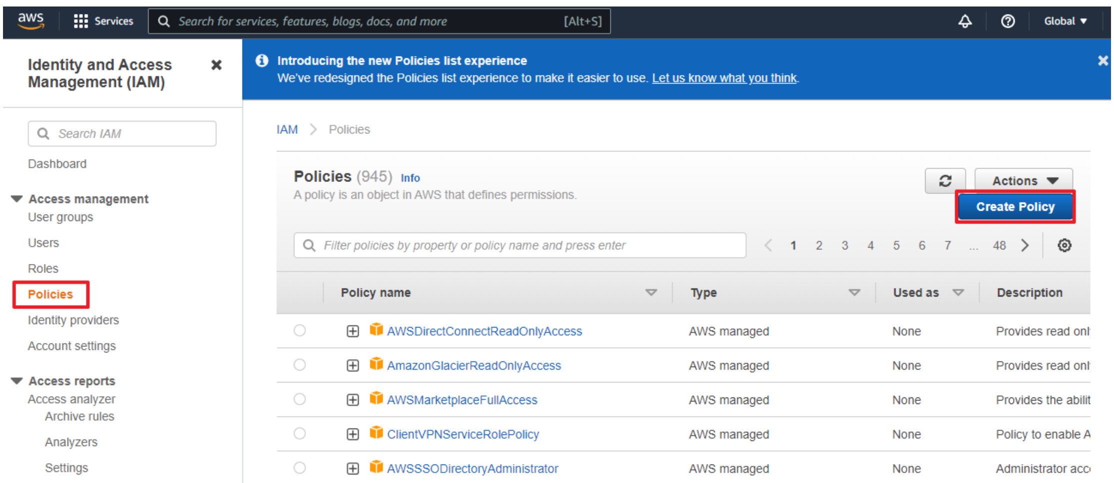


Figure 4-148 Ready to create IAM Role Policy

12. You can choose to follow the wizard to automatically fill in the details according to your requirements, or you can manually enter the JSON parameters. Here is an example as shown in Figure 4-149.

Figure 4-149 Input the policy content of Put Object to S3 Bucket (JSON format)

13. Following the process, after entering the Policy name in the final step, click on "Create Policy" to complete the creation. You will be able to view the newly created Policy in the list of Policies (as shown in Figure 4-150).

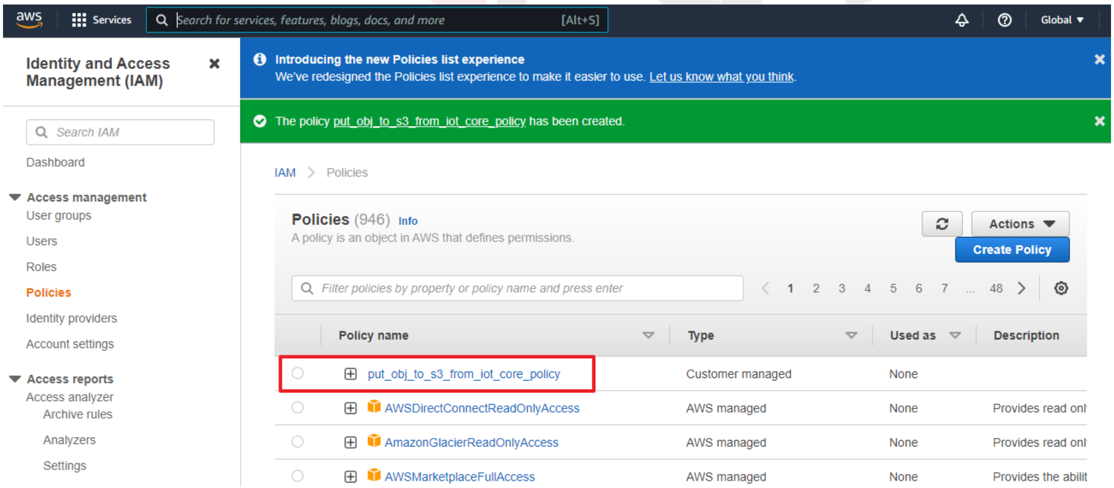


Figure 4-150 Complete the establishment of Policy

14. The Role requires a Policy to have functionality. Click on "Roles" on the left-hand side of the dashboard, then click on the recently created Role. Expand the "Add permissions" section and select "Attach policies" (as shown in Figure 4-151).

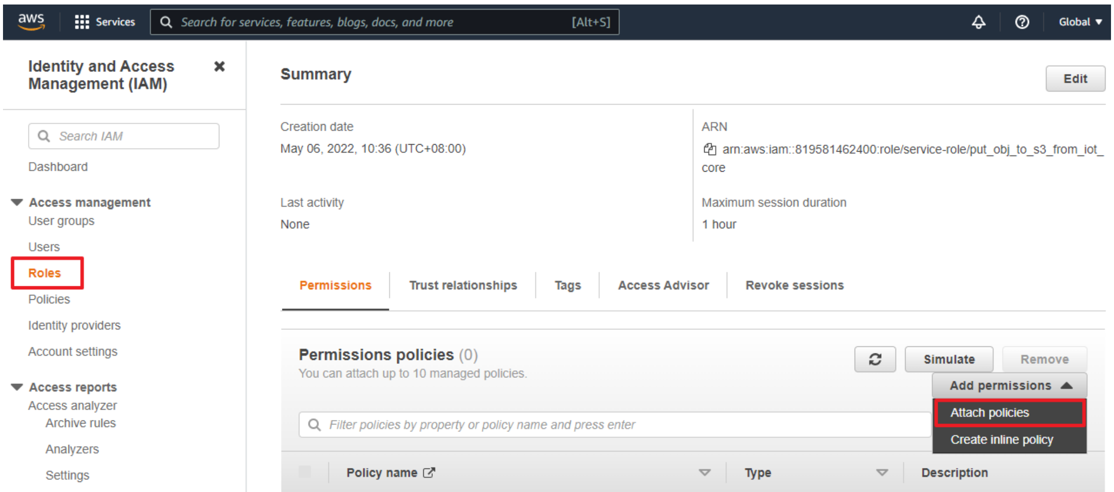


Figure 4-151 Prepare to attach Policy to Role

15. Upon entering the screen to choose the Policy, select the desired Policy to attach, and then click on "Attach policies" located at the bottom right corner of the webpage (as shown in Figure 4-152).

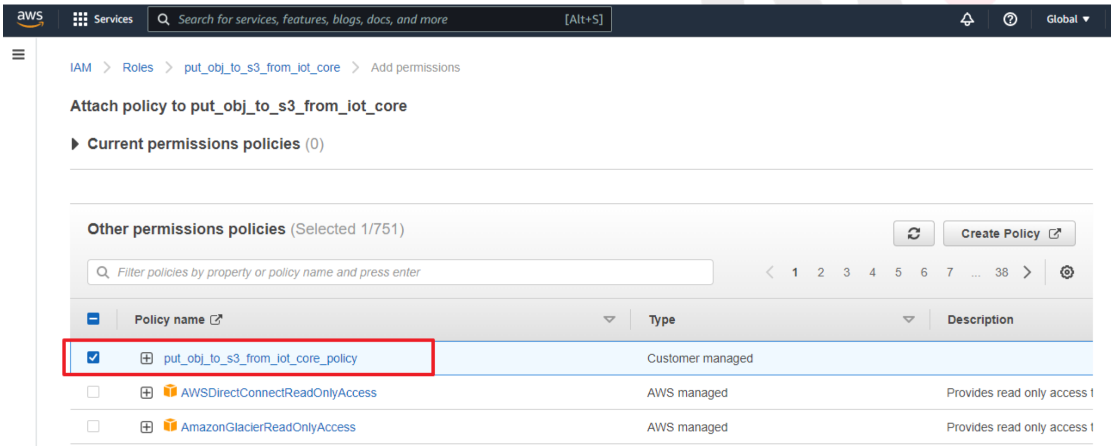


Figure 4-152 Select Policy to attach to Role

4.1.5.7.2 Basic Setting

Click on the icon to access the AWS IoT cloud service connection settings (as shown in Figure 4-153). Table 4-29 provides the configuration options for AWS IoT cloud connection and upload methods.

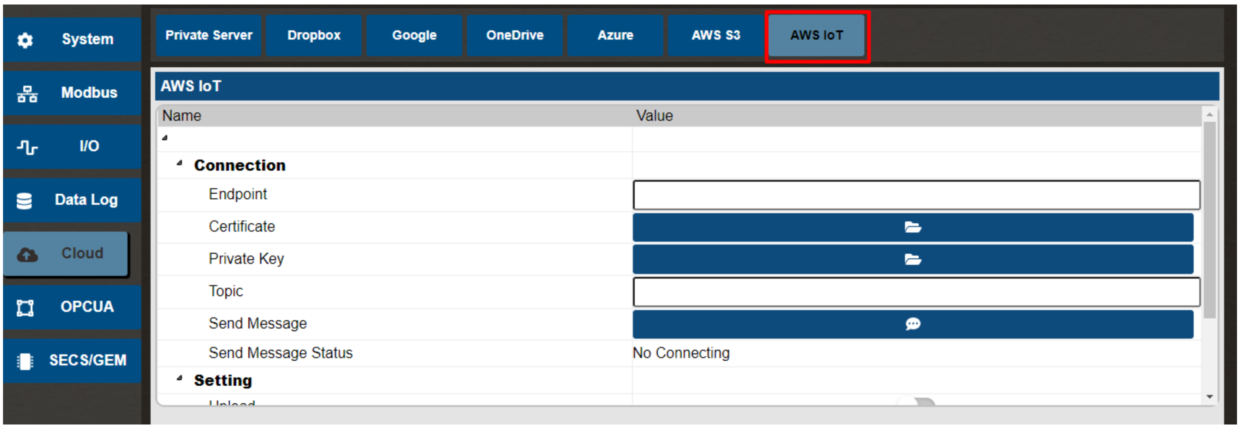


Figure 4-153 AWS IoT cloud service connection settings

Table 4-29 AWS IoT Cloud Connection Setting Parameters

Parameter name		Content
Connection	Endpoint	Enter the endpoint.
	Certificate	Click the button to import the certificate file.
	Private key	Click the button to import the private key file.
	Topic	Enter the topic.
	Send message	Click the button to send a message to the AWS IoT Core test client.
	Message sending status	Display the message sending status.
Settings	File upload	Enable/Disable file upload.
	Upload method	Time interval/Item cycle interval.
	Time interval (ms)	Enter the time based on the upload method time interval mode.
	Item cycle interval (quantity)	Enter the quantity based on the upload method item cycle interval mode.
	File name format	YYYYMMDDHHMMSS / YYYYMMDD.
	Timestamp format	UTC / Local Time.

4.1.5.7.3 Connection Process

Upon accessing the AWS IoT Core entry point, navigate to "Settings" located at the bottom of the left-hand side dashboard. From there, copy the Endpoint (as shown in Figure 4-154).

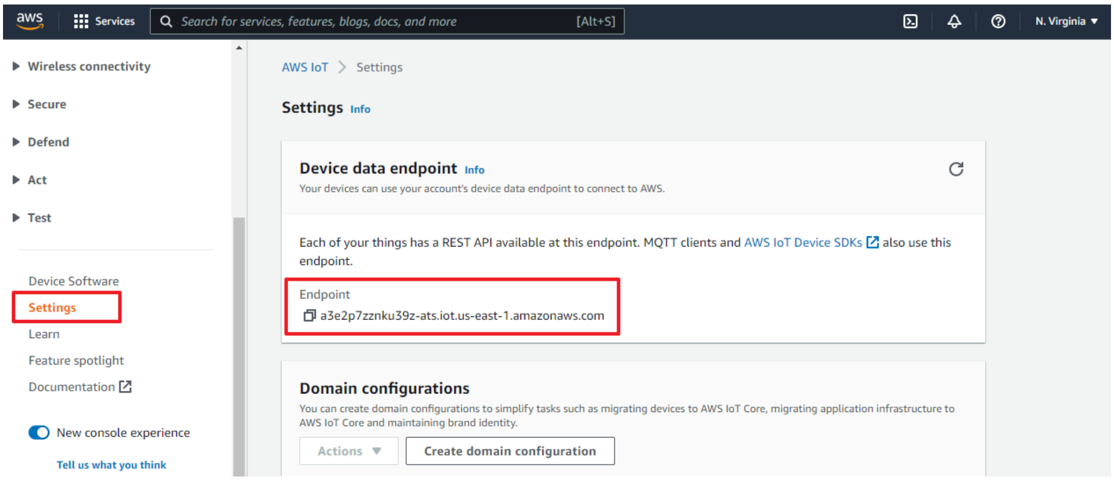


Figure 4-154 Copy the Endpoint

Paste the Endpoint (as shown in Figure 4-154), load the certificate and private key (as shown in

Figure 4-133), enter the subscribed topic (as shown in Figure 4-135). The completed input screen should resemble Figure 4-155.

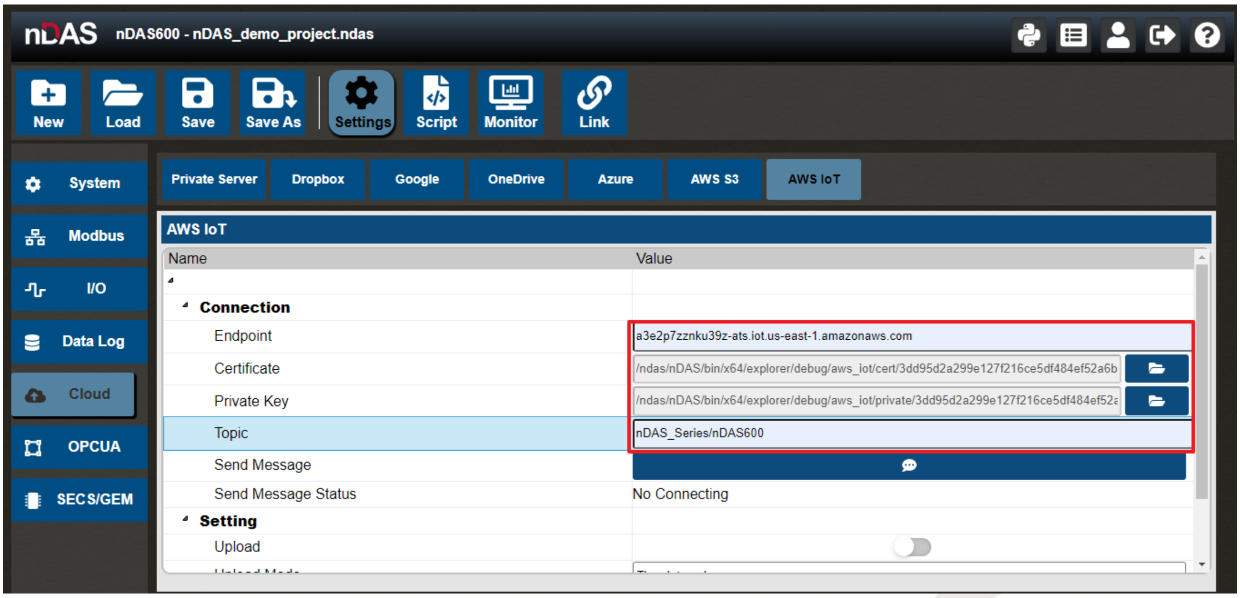


Figure 4-155 Ready to test the connection of AWS IoT

Access the IoT Core entry point and navigate to the MQTT test client from the bottom of the left-hand side dashboard. In the testing interface, enter the topic you have configured in the "Topic" field. Click on the "Subscribe" button. The subscribed topic will appear in the table below (as shown in Figure 4-156). Finally, go back to the nDAS interface and click on the "Send Message" button (as shown in Figure 4-157).

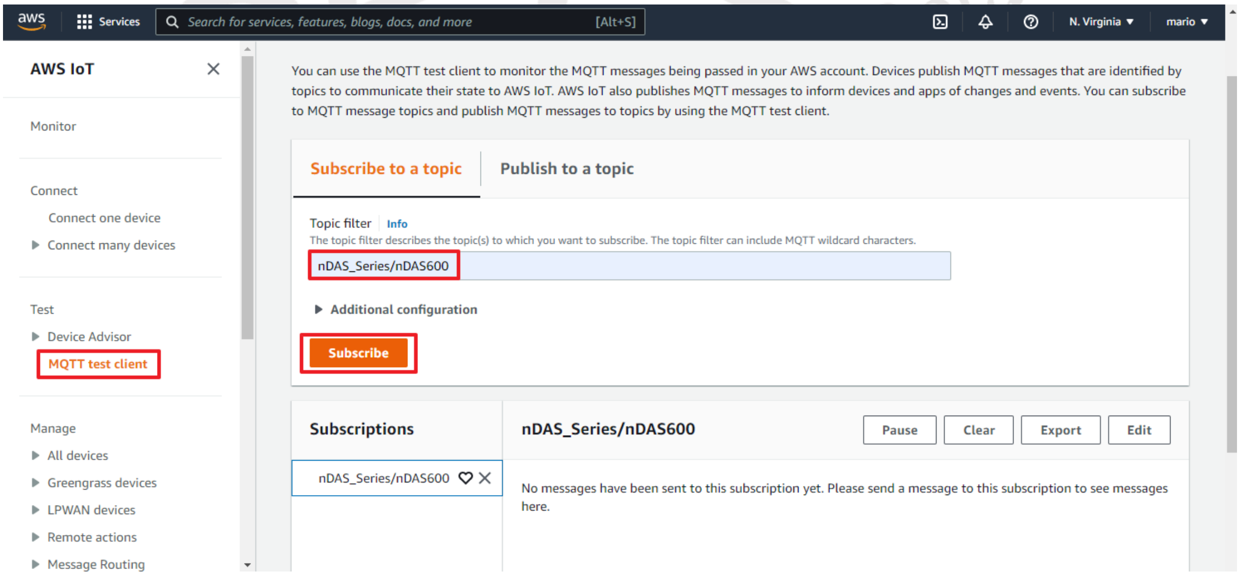


Figure 4-156 AWS IoT connection test screen

From the subscribed topic menu, you can see the test result indicating a successful connection, as shown in Figure 4-157.

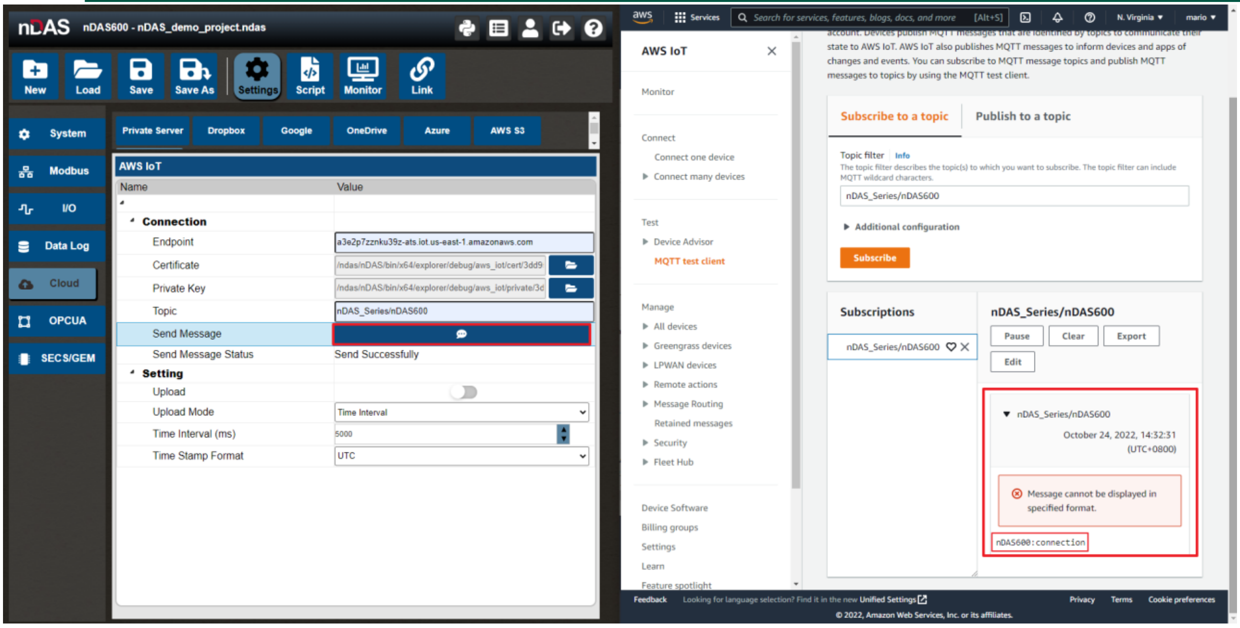


Figure 4-157 Click "Send Message" to view the AWS IoT connection test results

4.1.6 SECS/GEM

By clicking on the icon located in the left-hand menu, you gain access to the interface dedicated to configuring parameters for the SECS/GEM equipment side (as depicted in Figure 4-158). Reference Table 4-30 for the SECS/GEM equipment side parameter settings.

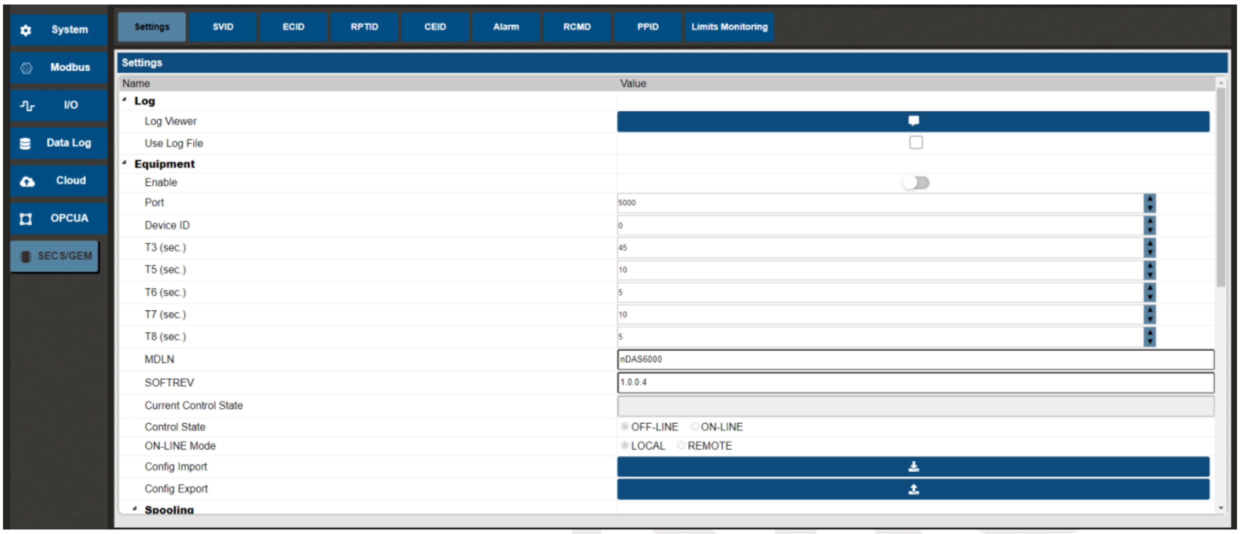


Figure 4-158 SECS/GEM Equipment parameter setting screen

Table 4-30 SECS/GEM Equipment parameter setting

Parameter name		Content
Log	Log Viewer	Upon clicking, the log window for the SECS Equipment becomes visible.
	Host Log Viewer	Upon clicking, the log window for the SECS HOST becomes visible.
	Use Log File	When selected, logs will be stored as files. The saved data will be located in the directory path: C:\VIC7000\bin\log.
Equipment	Enable	Is the SECS equipment side operational?
	Port	Set the communication port number for the equipment side, with a default of 5000.
	Device ID	Specify the station number for the equipment side, with a default of 0.
	T3 (sec.)	Configure the T3 parameter for the equipment side, measured in seconds, with a default of 45 seconds.
	T5 (sec.)	Configure the T5 parameter for the equipment

		side, measured in seconds, with a default of 10 seconds.
	T6 (sec.)	Configure the T6 parameter for the equipment side, measured in seconds, with a default of 5 seconds.
	T7 (sec.)	Configure the T7 parameter for the equipment side, measured in seconds, with a default of 10 seconds.
	T8 (sec.)	Configure the T8 parameter for the equipment side, measured in seconds, with a default of 5 seconds.
	MDLN	Set the MDLN parameter for the equipment side, with a default of the VIC product model.
	SOFTREV	Set the SOFTREV parameter for the equipment side, with a default of the software version.
	Current Control State	Display the current control status of the equipment side.
	Control State	Configure the control status of the equipment side.
	ON-LINE Mode	Configure the ON-LINE mode of the equipment side. Note: Only effective when the control status is ON-LINE.
	Config Import	Upon clicking, you can select a configuration for import.
	Config Export	Upon clicking, you initiate the export of the configuration.
Spooling	Current Spooling State	Display the current simulated offline status of the equipment side.
	Actual Count	Display the number of messages actually stored in the simulated offline area of the equipment side.
	Actual Count SVID	Configure the SVID for the actual quantity parameter of simulated offline messages for the equipment side.

	Total Count	Display the total number of messages ever stored in the simulated offline area of the equipment side.
	Total Count SVID	Configure the SVID for the total quantity parameter of simulated offline messages for the equipment side.
	Start Time	Display the time when the equipment side started storing messages in the simulated offline area.
	Start Time SVID	Configure the SVID for the start time parameter of simulated offline messages for the equipment side.
	Full Time	Display the time when the simulated offline area became full on the equipment side. The default maximum storage for the simulated offline area is 100 messages.
	Full Time SVID	Configure the SVID for the full time parameter of simulated offline messages for the equipment side.
	Max Transmit	Set the maximum number of messages to transmit from the simulated offline area when responding to an offline message request (S6F23, Transmit Spooled Messages).
	Max Transmit ECID	Configure the ECID for the maximum send quantity parameter of simulated offline messages.
	Overwrite	Configure whether to enable overwrite functionality when the simulated offline area is full.
CEID	Change ECV	Configure the CEID triggered when ECV is changed.
	Equipment OFF-LINE	Configure the CEID triggered when the equipment side control status changes to OFF-LINE.
	ON-LINE LOCAL	Configure the CEID triggered when the ON-LINE mode of the equipment side changes to

		LOCAL.
	ON-LINE REMOTE	Configure the CEID triggered when the ON-LINE mode of the equipment side changes to REMOTE.
	Add PPID	Configure the CEID triggered when a PPID is added.
	Delete PPID	Configure the CEID triggered when a PPID is deleted.
	Message Recognition	Configure the CEID triggered when S10F03 receives an acknowledge message.
	Spooling Deactivated	Configure the CEID triggered when the messages in the simulated offline area are cleared.

4.1.6.1 SVID

The software provides the ability to configure Status Variable IDs (SVIDs) for the equipment side, which can be set as I/O, extended bits/characters for Modbus, or as script calculations (CALC). To add a new SVID, follow these steps:

1. Click on the "+" icon button, which will trigger the display of the SVID window.

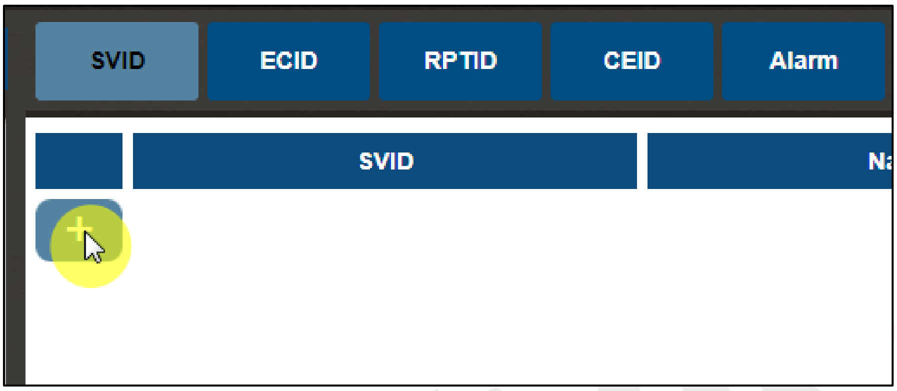


Figure 4-159 Click on the "+" icon button

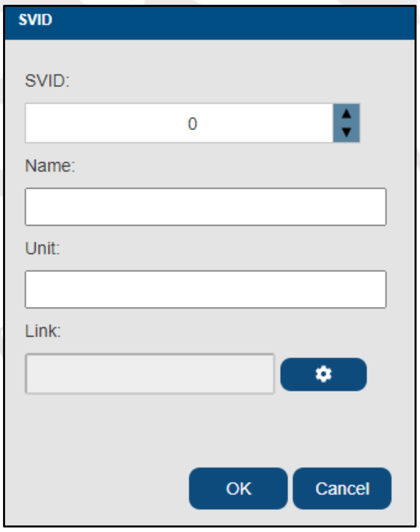


Figure 4-160 SVID window

Note: It is imperative that SVIDs and ECIDs remain distinct and non-zero.

2. Subsequently, by clicking on the configuration symbol link, the SV Link window will emerge. Within this window, you can configure the desired link to I/O, extended bits/characters for Modbus, or script calculations (CALC).

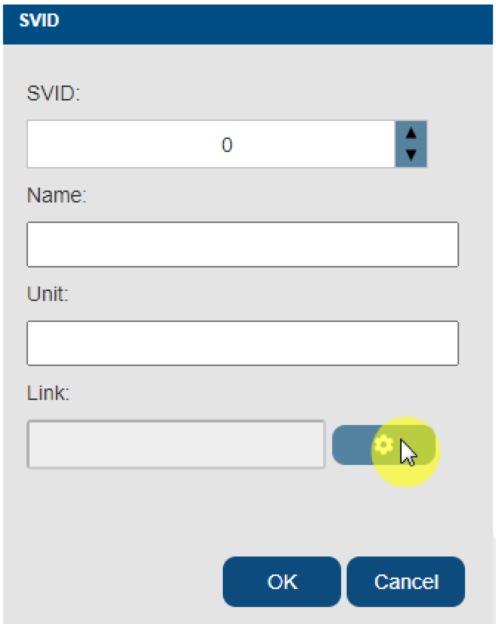


Figure 4-161 Clicking on the configuration symbol link

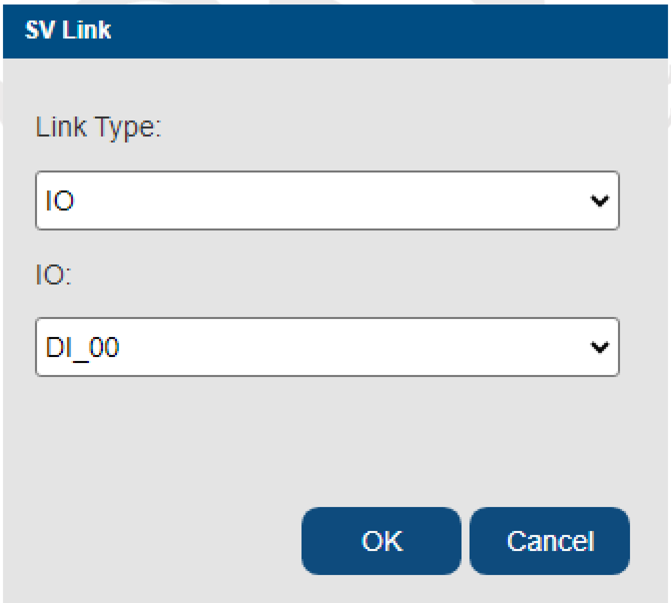


Figure 4-162 SV Link window

3. Finally, upon clicking "Confirm," the configuration of the SVID parameters will be successfully concluded.

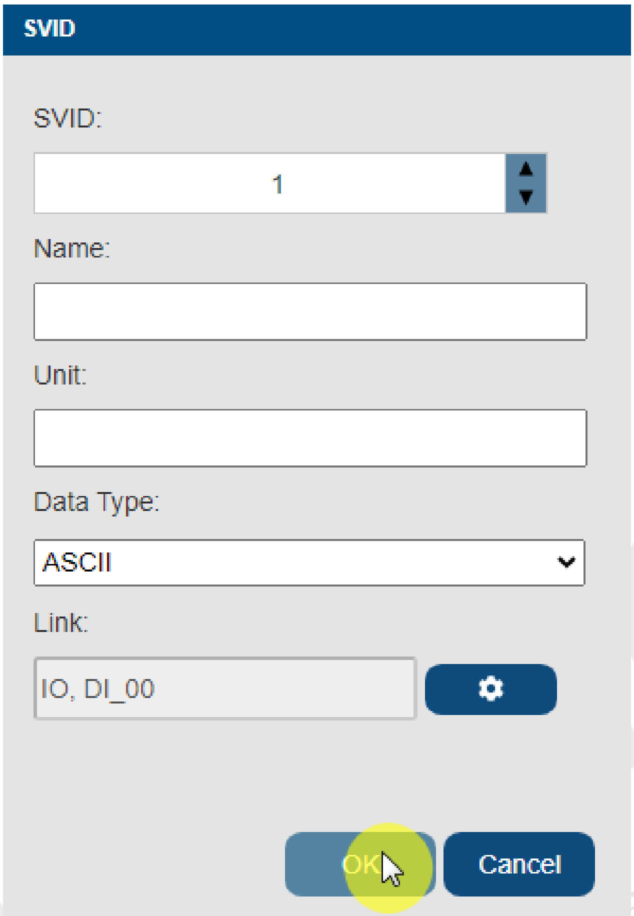


Figure 4-163 clicking "Confirm"

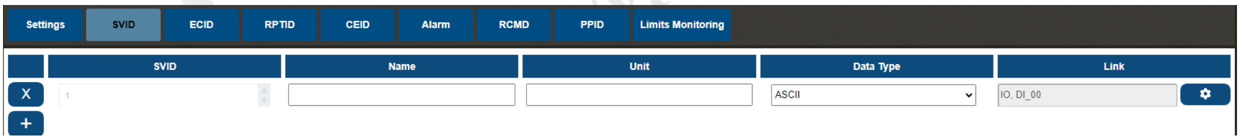


Figure 4-164 Setting SVID successfully

4.1.6.2 ECID

To add a new Equipment Constants ID (ECID) in the software's equipment side, follow these steps:

1. Click on the "+" icon button, which will prompt the display of the ECID window.

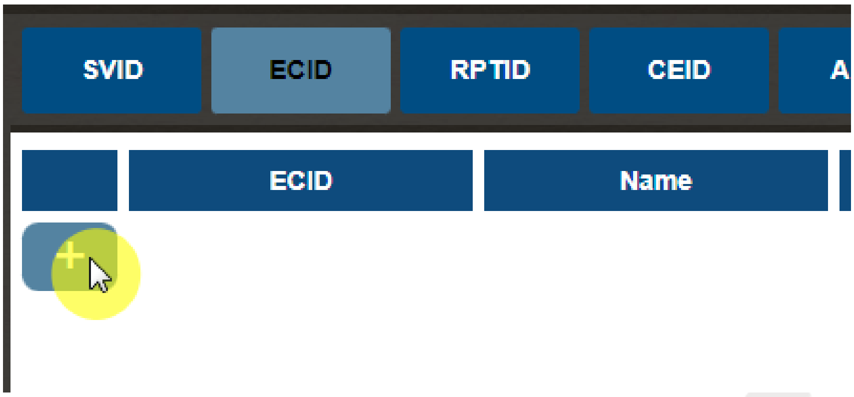


Figure 4-165 Click on the "+" icon button

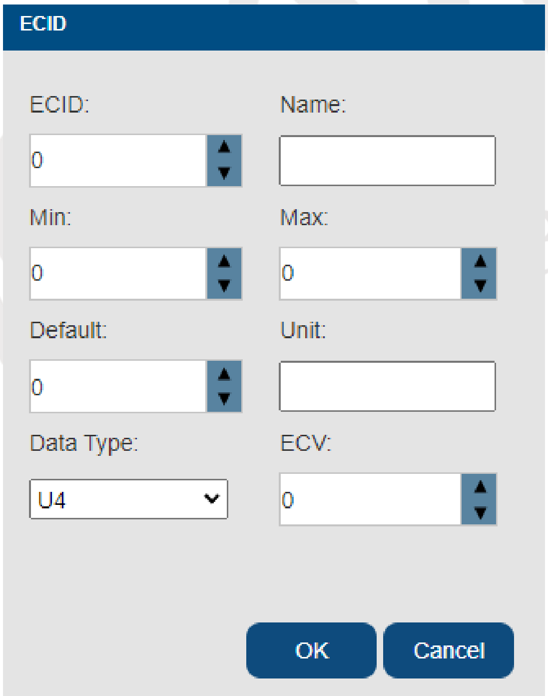


Figure 4-166 ECID window

Note: Please be aware that SVIDs and ECIDs must remain distinct and non-zero.

2. Click "Confirm" to finalize the ECID parameter configuration.

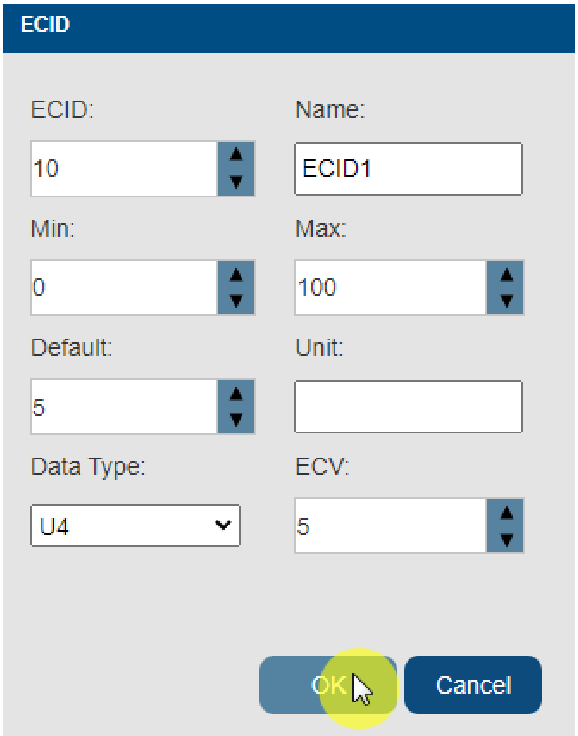


Figure 4-167 clicking "Confirm"

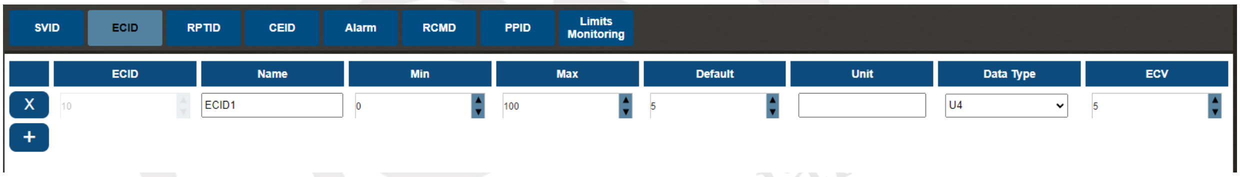


Figure 4-168 Setting ECID successfully

4.1.6.3 RPTID

To add a new Report ID (RPTID) in the software's equipment side, follow these steps:

1. Click on the "+" icon button, which will bring up the RPTID window.

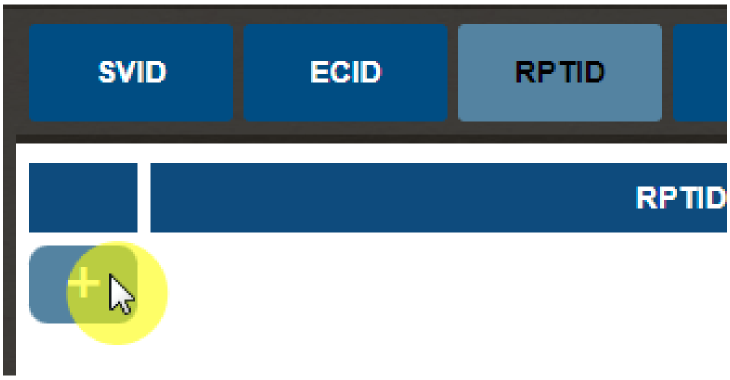


Figure 4-169 Click on the "+" icon button

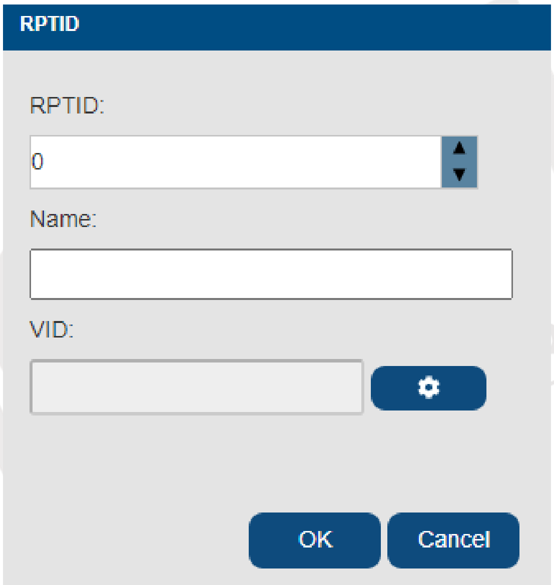


Figure 4-170 RPTID window

2. Click on the configuration symbol link for VID, which will lead to the VID configuration window. Here, you can set one or multiple VID(s).

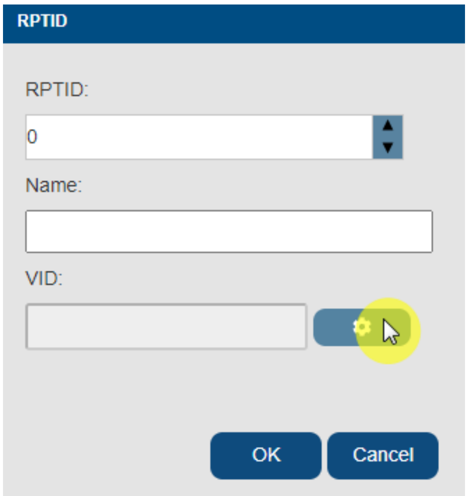


Figure 4-171 Clicking on the VID configuration symbol

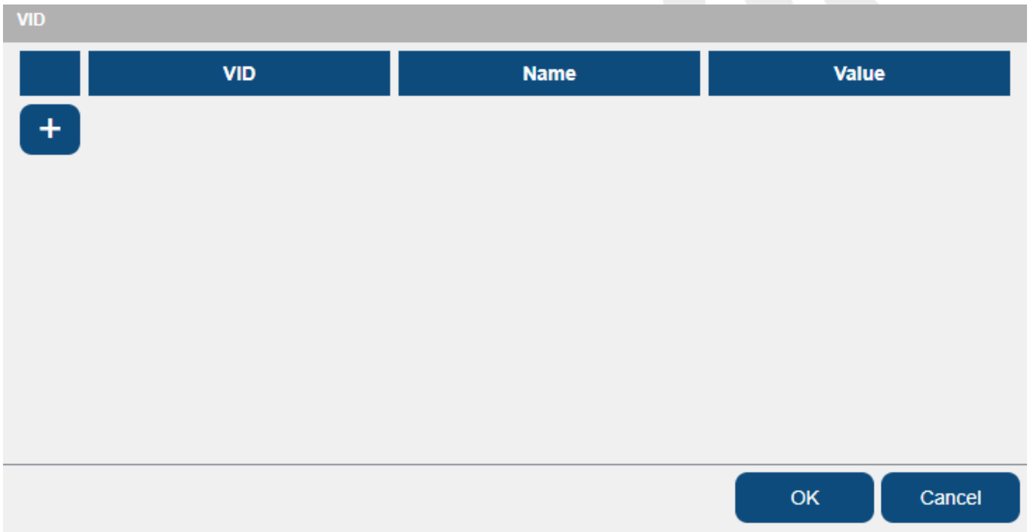


Figure 4-172 VID configuration window

3. Finally, click "Confirm" to complete the RPTID parameter configuration.

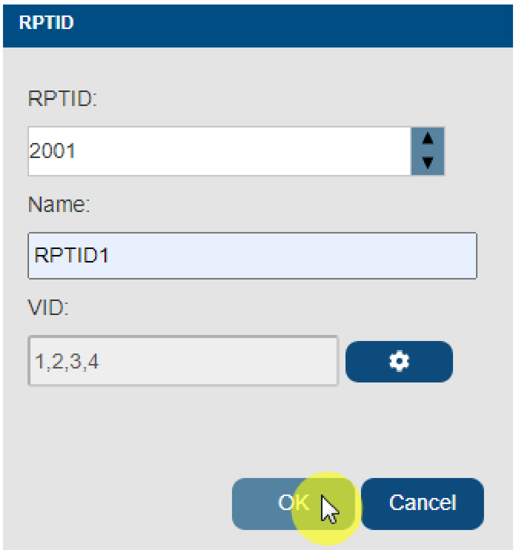


Figure 4-173 clicking "Confirm"

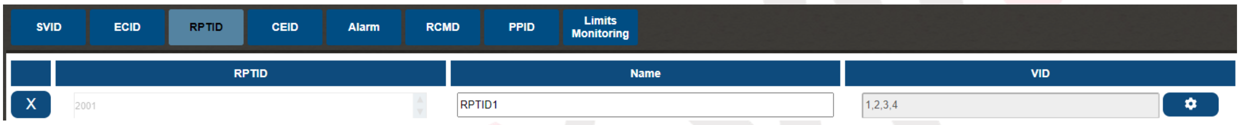


Figure 4-174 Setting RPTID successfully

4.1.6.4 CEID

To add a new Collection Event ID (CEID) in the software's equipment side, follow these steps:

1. Click on the "+" icon button, which will bring up the CEID window.

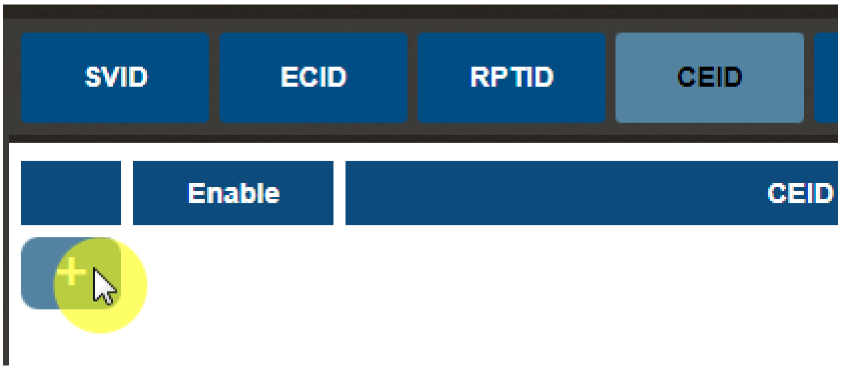


Figure 4-175 Click on the "+" icon button

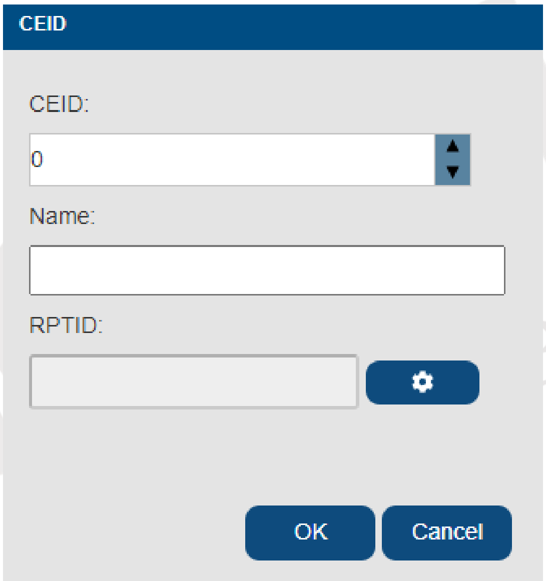


Figure 4-176 CEID window

2. Click on the configuration symbol link for RPTID, which will lead to the RPTID configuration window. Here, you can set one or multiple RPTID(s).

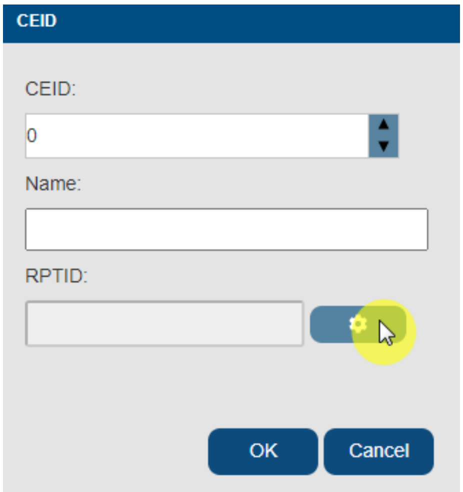


Figure 4-177 Clicking on the RPTID configuration symbol

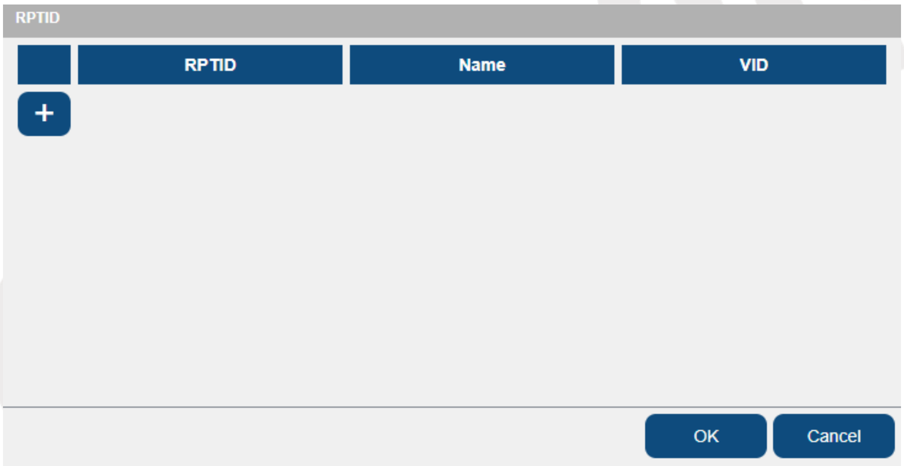


Figure 4-178 RPTID configuration window

3. Finally, click "Confirm" to complete the CEID parameter configuration.

Figure 4-179 clicking "Confirm"

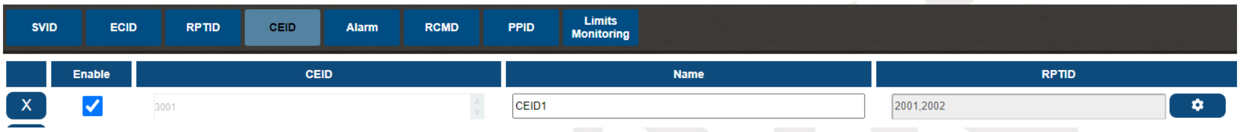


Figure 4-180 Setting CEID successfully

4.1.6.5 Alarm

To add a new Alarm in the software's equipment side, follow these steps:

1. Click on the "+" icon button, which will bring up the Alarm window. The parameters within this window are introduced in the following table.

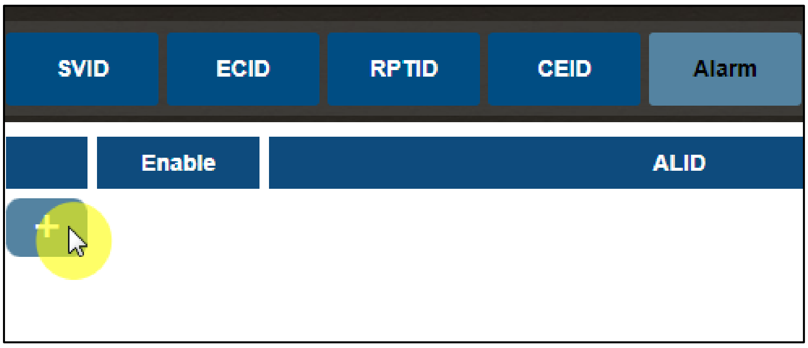


Figure 4-181 Click on the "+" icon button

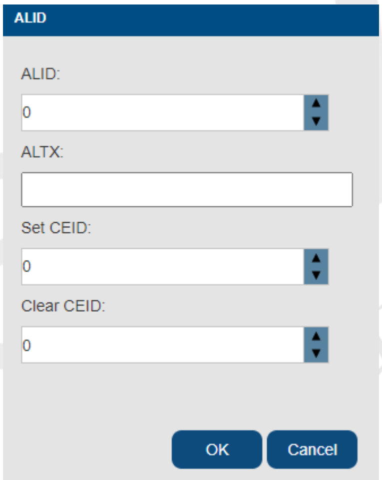


Figure 4-182 Alarm window

Table 4-31 Alarm parameter setting

Parameter name	Content
ALID	Alarm Identification.
ALTX	Alarm Text.
Set CEID	CEID to be triggered when the alarm state is triggered (set).
Clear CEID	CEID to be triggered when the alarm state is cleared.

2. Finally, click "Confirm" to complete the Alarm parameter configuration.

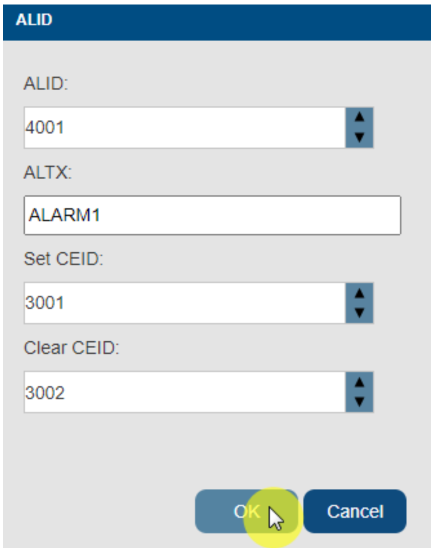


Figure 4-183 clicking "Confirm"

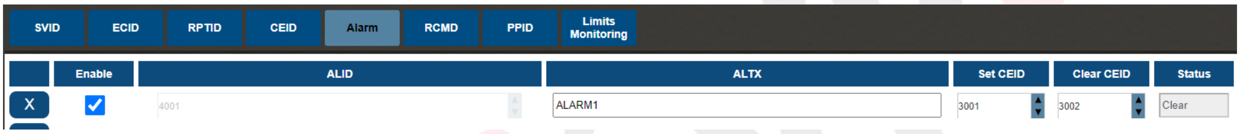


Figure 4-184 Setting Alarm successfully

4.1.6.6 RCMD

To add a new Remote Command String (RCMD) in the software's equipment side, follow these steps:

1. Click on the "+" icon button, which will bring up the RCMD window.

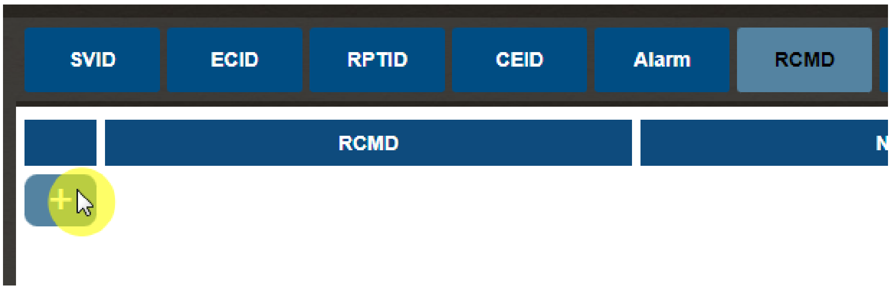


Figure 4-185 Click on the "+" icon button

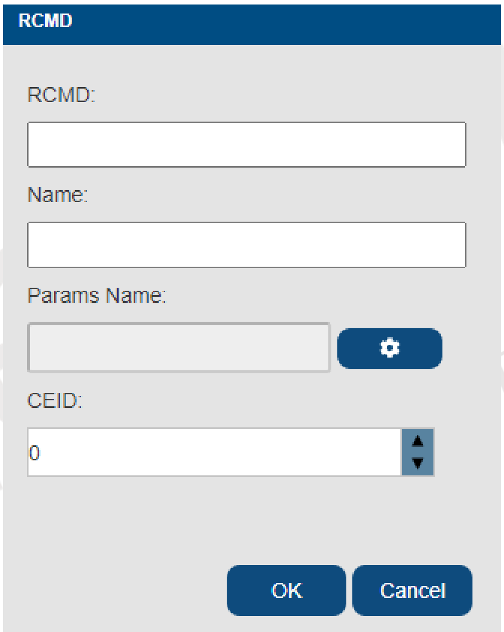


Figure 4-186 RCMD window

2. Click on the configuration symbol link for parameter names, which will lead to the parameter name configuration window. Here, you can set one or multiple parameter names.

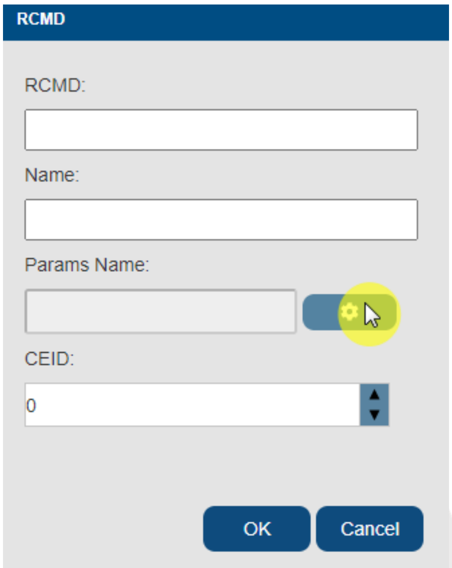


Figure 4-187 Clicking on the parameter name configuration symbol

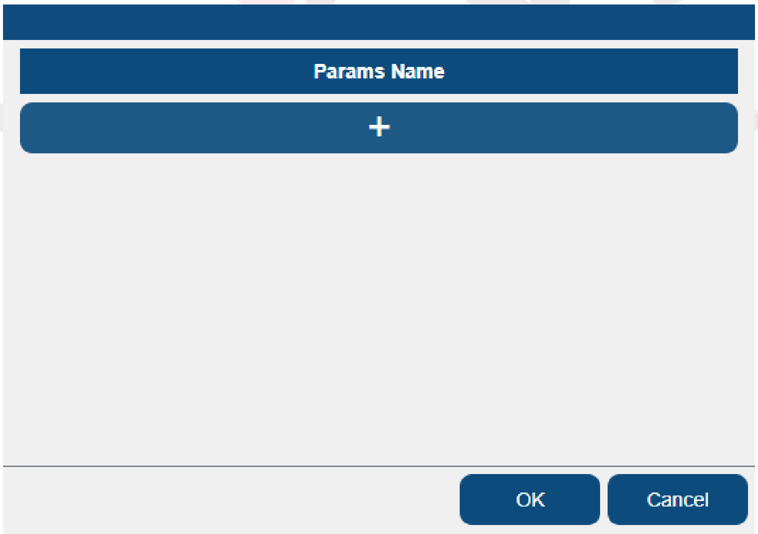


Figure 4-188 parameter name configuration window

3. Finally, click "Confirm" to complete the RCMD parameter configuration.

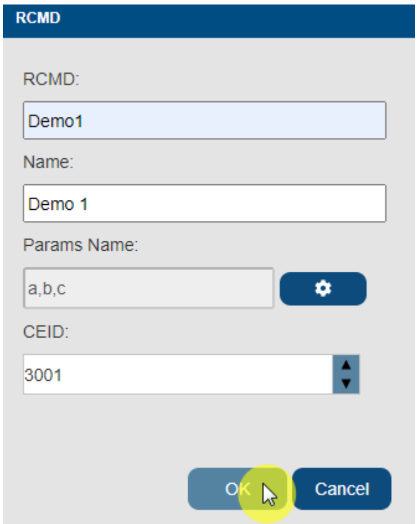


Figure 4-189 clicking "Confirm"

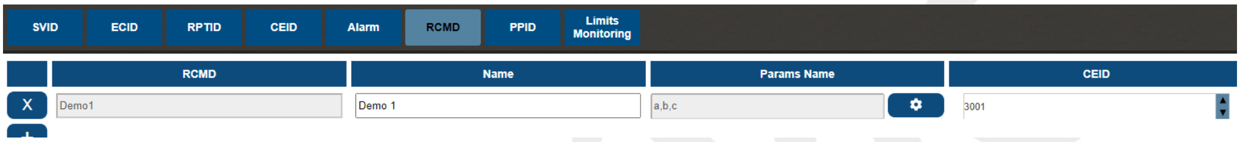


Figure 4-190 Setting RCMD successfully

4.1.6.7 PPID

To add a new Process Program Identity (PPID) in the software's equipment side, follow these steps:

1. Click on the "+" icon button, which will bring up the PPID window.

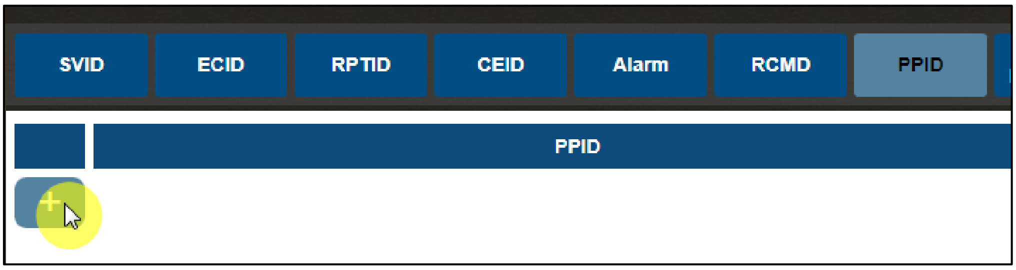


Figure 4-191 Click on the "+" icon button

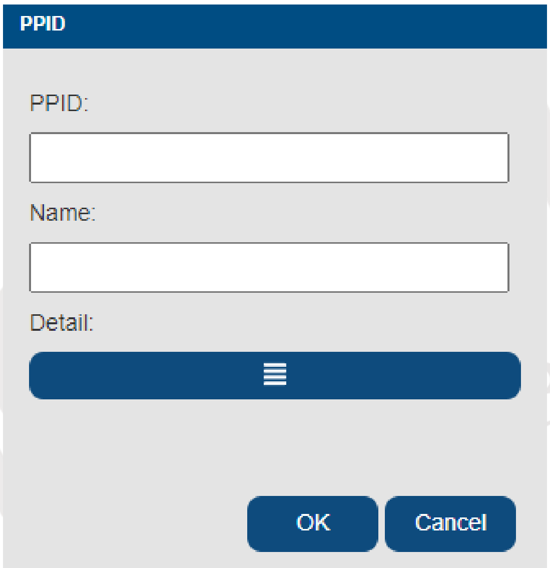


Figure 4-192 PPID window

2. Click on the "Detailed Content" button, which will lead to the detailed content configuration window for the process program. Here, you can set one or multiple Command Codes (CCODE) as well as the parameter names contained within each CCODE.

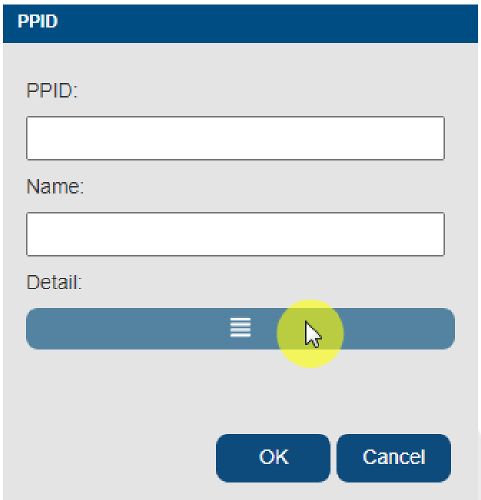


Figure 4-193 Click on the "Detailed Content" button

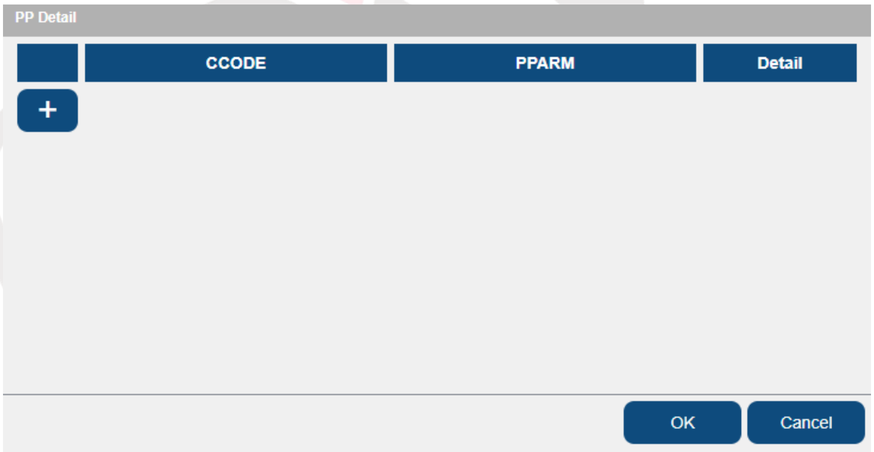


Figure 4-194 detailed content configuration window for the process program

3. Finally, click "Confirm" to complete the PPID parameter configuration.

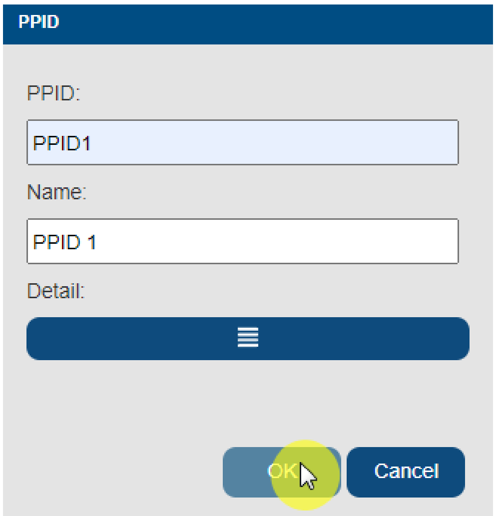


Figure 4-195 clicking "Confirm"

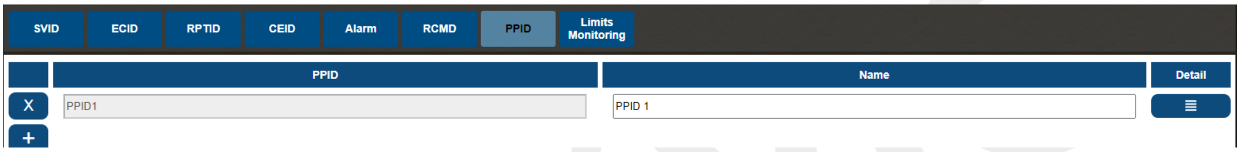
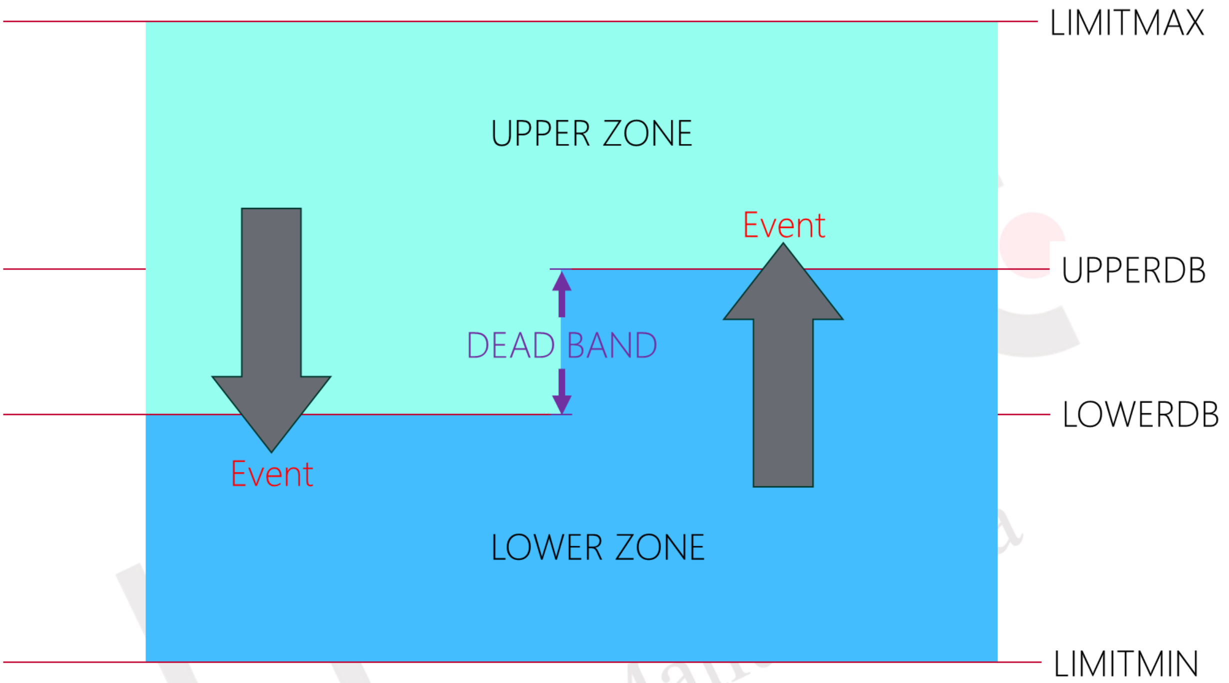


Figure 4-196 Setting PPID successfully

4.1.6.8 Limits Monitoring

This is used to monitor whether a Variable (VID) surpasses or falls below the set boundary values. The triggering principle is illustrated in the diagram below. The region between the Upper Zone and the Lower Zone contains a Dead Band. When a value transitions from the Upper Zone and falls below the Upper Dead Band (UPPERDB), no event is triggered until it drops below the Lower Dead Band (LOWERDB), where triggering occurs. Conversely, if a value transitions from the Lower Zone and exceeds the Upper Dead Band, triggering occurs.



To add a new Limit Monitoring in the software's equipment side, follow these steps:

1. Click on the "+" icon button, which will bring up the VID window. Here, you can set the VID to be monitored.

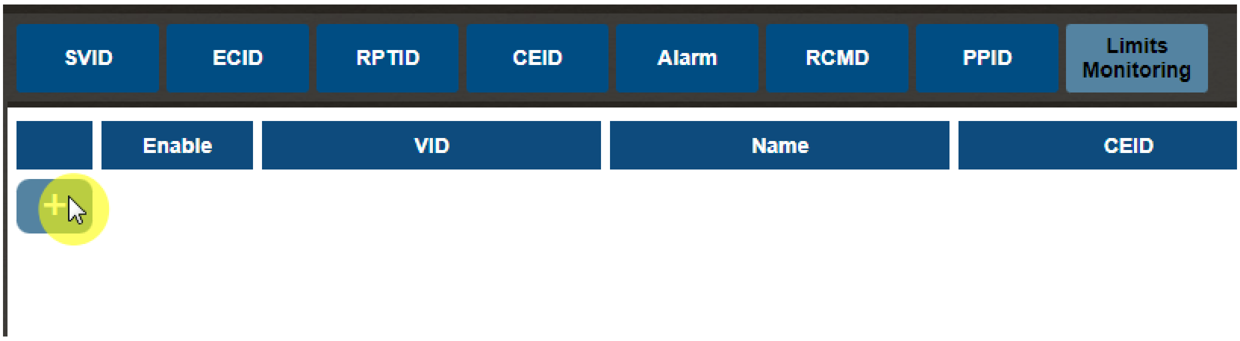


Figure 4-197 Click on the "+" icon button

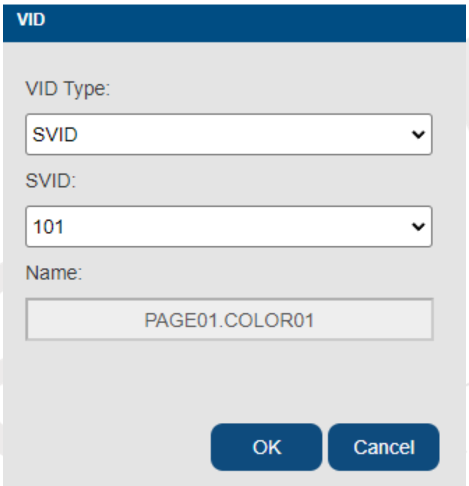


Figure 4-198 VID window

2. Set the maximum and minimum limits for the restriction, and specify the CEID to be triggered.

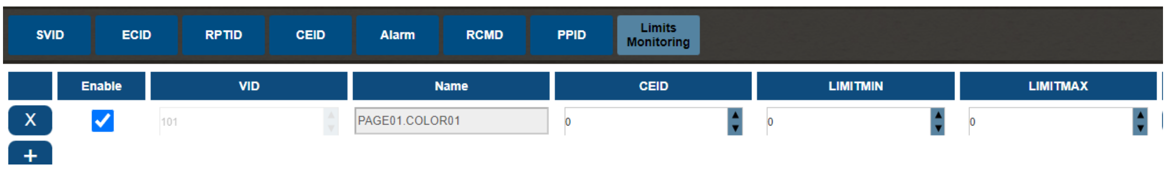


Figure 4-199 Set the maximum and minimum limits for the restriction, and specify the CEID to be triggered

3. Click on "Detailed Content," which will prompt the setting of upper and lower limits. In this window, you can configure individual or multiple Limit IDs (LIMITID), along with their corresponding Upper Dead Band (UPPERDB) and Lower Dead Band (LOWERDB) values. Upon completion, click "Confirm" to finalize the limit monitoring parameter configuration.

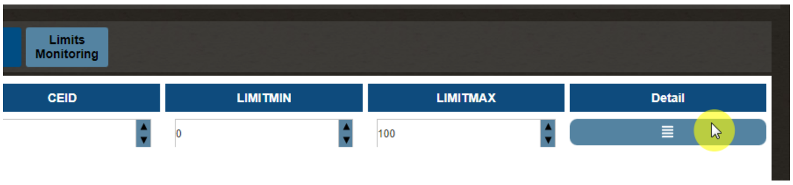


Figure 4-200 Click on "Detailed Content"

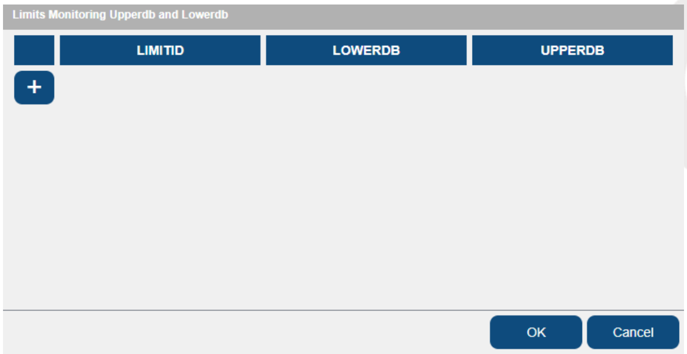


Figure 4-201 setting window of upper and lower limits

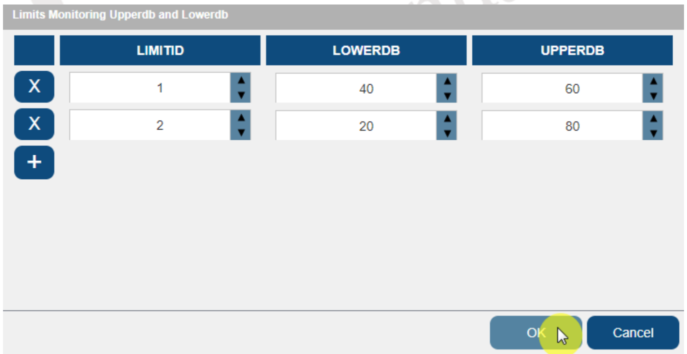


Figure 4-202 clicking "Confirm"

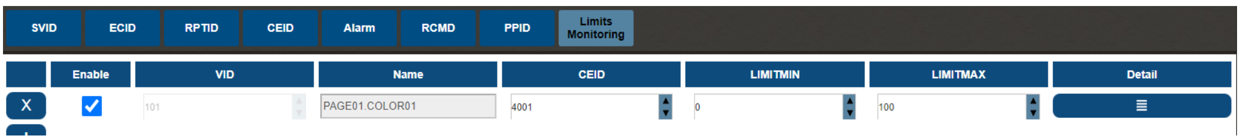


Figure 4-203 Setting limit monitoring parameter configuration successfully

4.1.7 OPCUA

Click on the icon in the left-side menu to access the OPC UA Server parameter settings screen (as shown in Figure 4-204). Table 4-32 provides the OPC UA Server parameter configuration details.

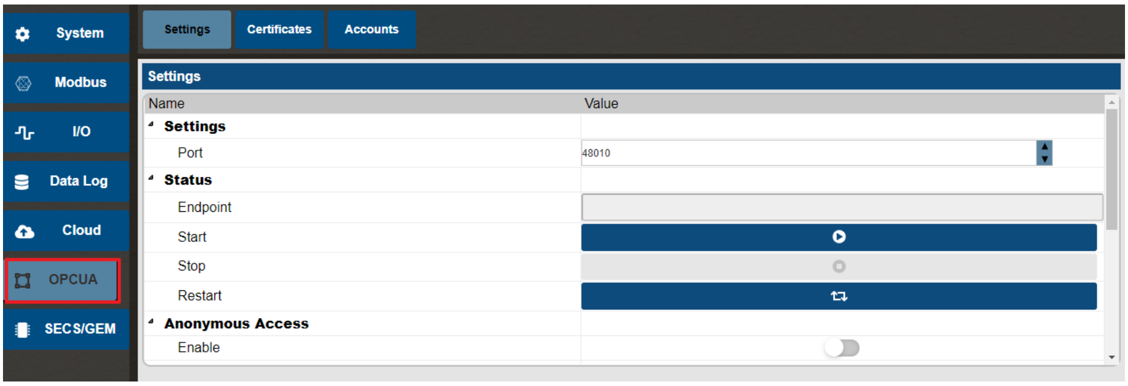


Figure 4-204 OPCUA Server parameter setting screen

Table 4-32 OPCUA Server parameter setting

Parameter name		Content
Settings	Port	Set the port number
Status	Endpoint	Display DI signal status
	Start	Start OPC UA Server
	Stop	Stop OPC UA Server
	Restart	Restart OPC UA Server
Anonymous Access	Enable	Enable/disable anonymous access mode
Security Policies	None	Enable/disable None security policy
	Basic128Rsa15	Enable/disable Basic128Rsa15 security policy
	Basic256	Enable/disable Basic256 security policy
	Basic256Sha256	Enable/disable Basic256Sha256 security policy
	Aes128Sha256RsaOaep	Enable/disable Aes128Sha256RsaOaep security policy
	Aes256Sha256RsaPss	Enable/disable Aes256Sha256RsaPss security policy
Security Modes	Sign	Enable/disable Sign security mode
	Sign & Encrypt	Enable/disable Sign & Encrypt security mode

Note: If you make changes to the configuration parameters of the OPCUA Server while it is running, you need to manually restart the OPCUA Server.

4.1.7.1 Certificates

Clicking on the icon in the top menu will display the OPCUA Server certificate list (as shown in Figure 4-205). When an OPCUA Client accesses the Server, the certificate information of that Client will be displayed. If the inspection of the certificate is satisfactory, you can right-click the mouse to choose whether to trust or deny the OPCUA Client (as shown in Figure 4-206).

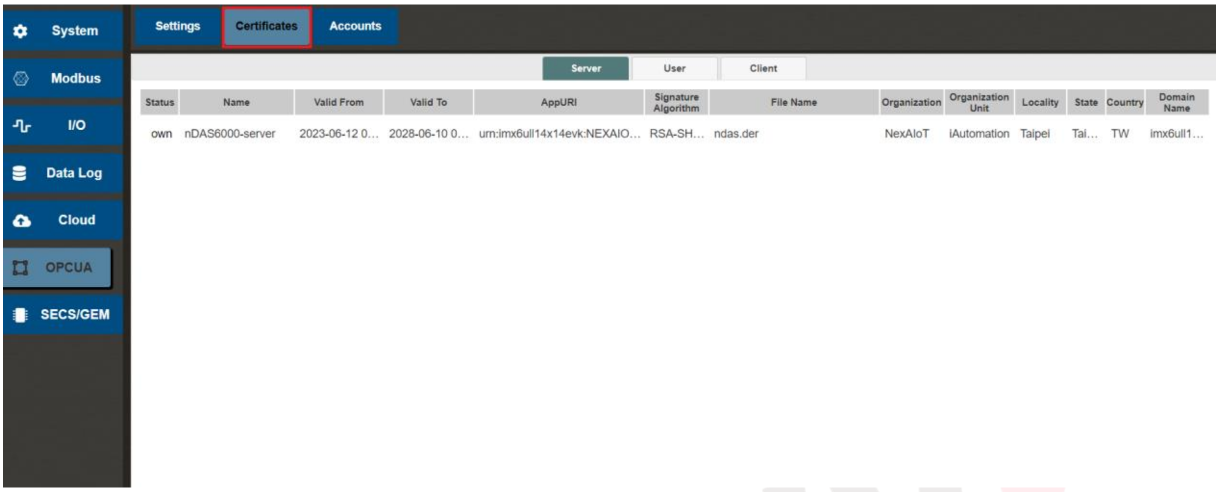


Figure 4-205 OPCUA server certification list

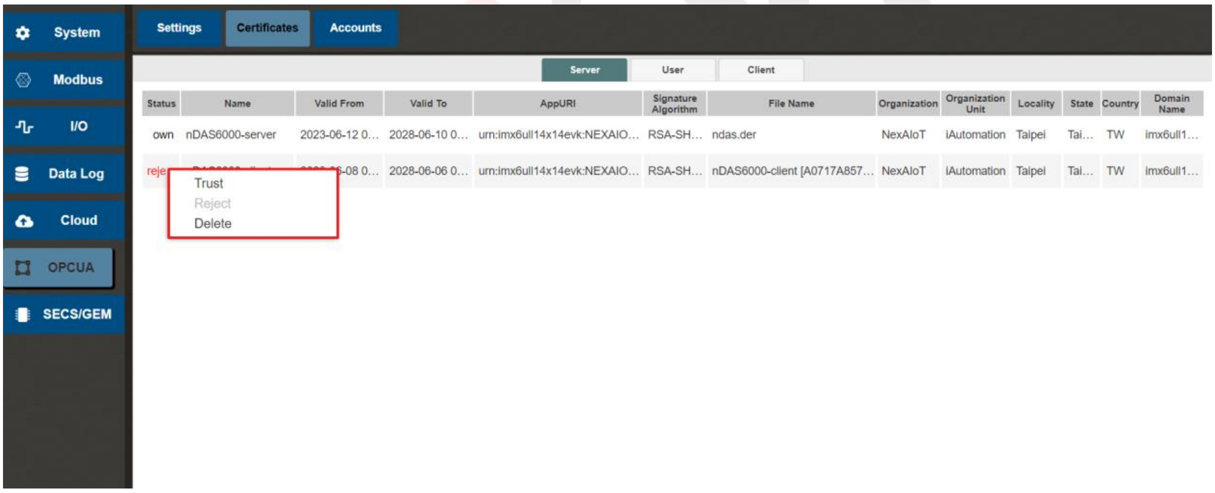


Figure 4-206 Select Trust / Reject for OPCUA client in OPCUA server certification list

If the OPCUA Client authentication method involves certificate validation, the OPCUA Server will perform a second verification. By clicking on the "User" tab in the OPCUA Server certificate list, the certificate information of that Client will be displayed. If the inspection of the certificate is satisfactory, you can right-click the mouse to choose whether to trust or deny the certificate (as shown in Figure 4-207).

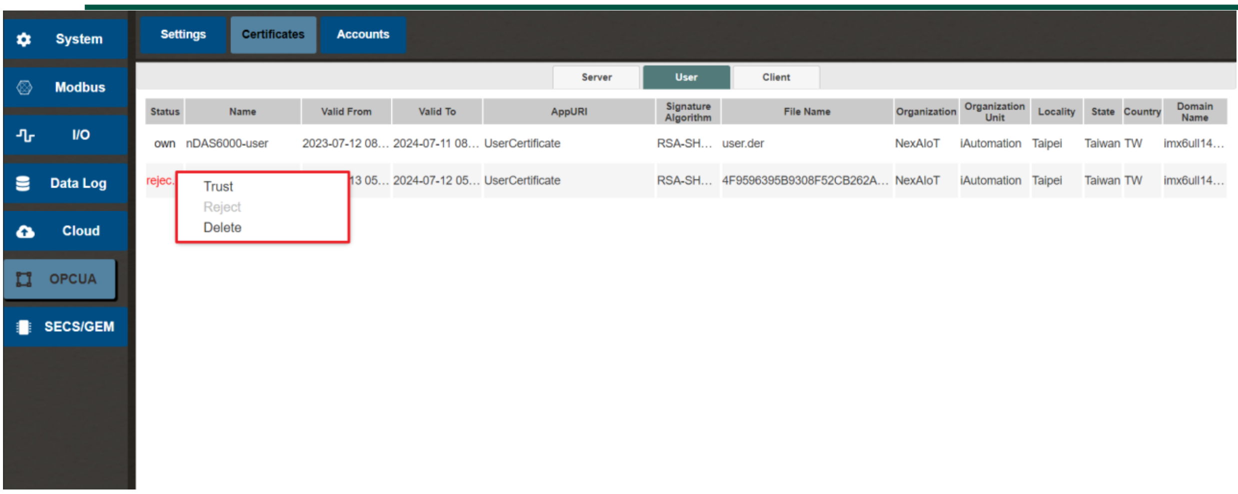


Figure 4-207 Select Trust / Reject for OPCUA user in OPCUA user certification list

By clicking on the "Client" tab in the OPCUA Server certificate list, you can view the certificates of the local machine when connected as a Client to another OPCUA Server. By default, these certificates are trusted (as shown in Figure 4-208). For more detailed information about OPCUA Client and its operation, please refer to the OPCUA Client operator's description in the VIC Flow manual.

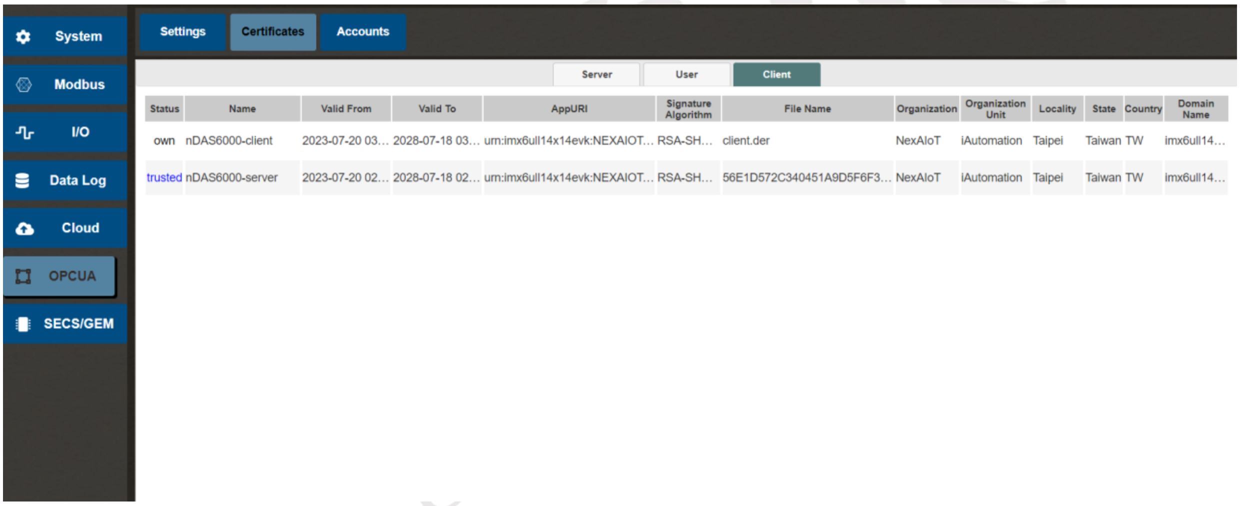


Figure 4-208 Select Trust / Reject for OPCUA server in OPCUA client certification list

4.1.7.2 Account

Kindly click on the icon located at the top menu to reveal the roster of OPCUA Server accounts (as illustrated in Figure 4-209). By selecting the "+" symbol, you will be able to create a new user. Enter the desired username and password, followed by the selection of user privileges (as depicted in Figure 4-210). To finalize the configuration, kindly click on "OK" (as shown in Figure 4-211).

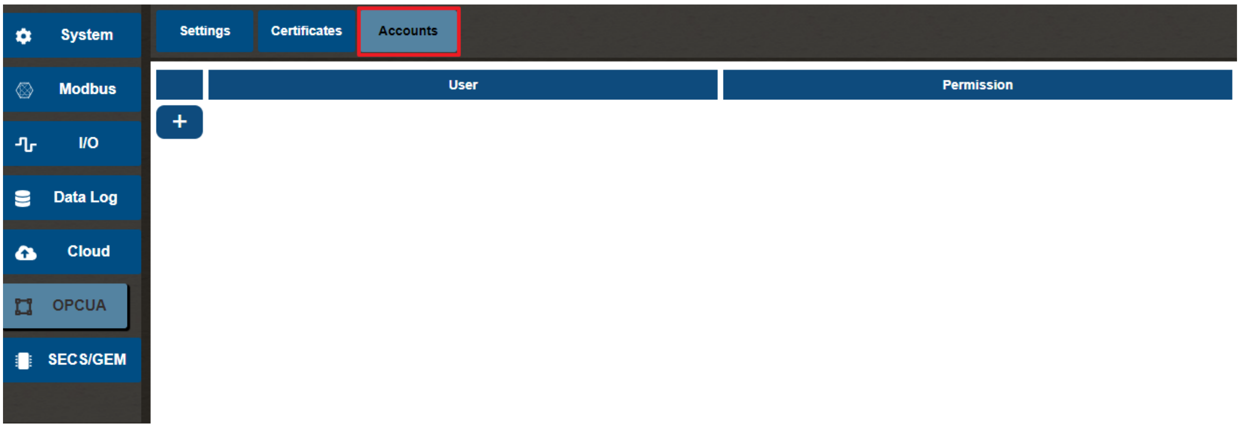


Figure 4-209 OPCUA Server Account List

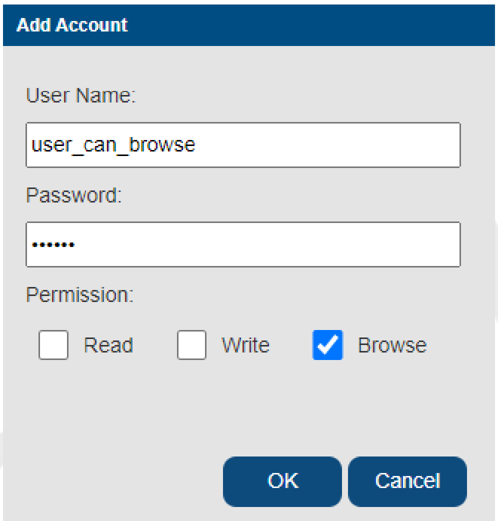


Figure 4-210 Add OPCUA Server User

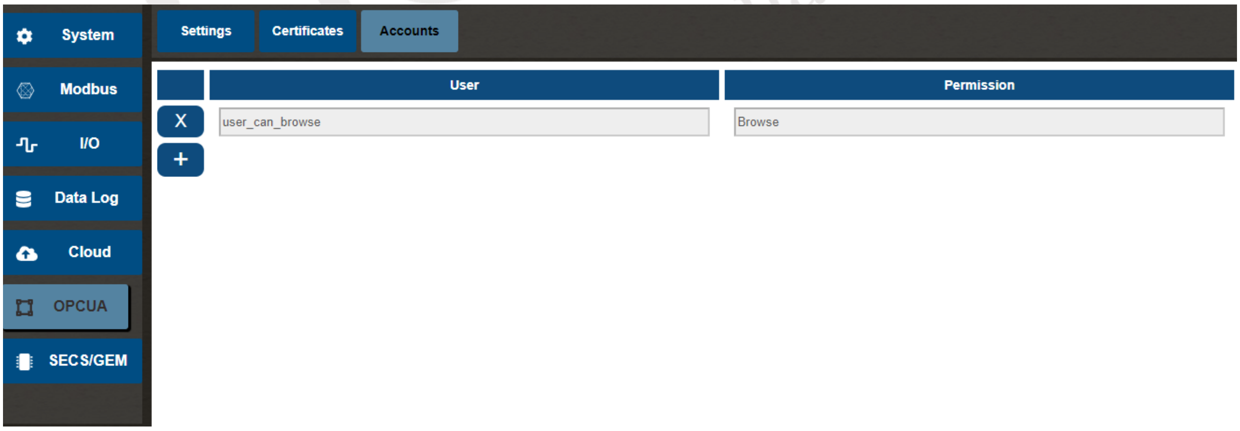


Figure 4-211 Finished to Adding OPCUA Server Users

Note: If the OPCUA Server is running, you need to restart the OPCUA Server manually when you finish adding users.

4.2 Script

By clicking on the script icon (as shown in Figure 4-212), you can enter the script, as illustrated in Figure 4-213. The scripting language used is Python.

Figure 4-212 Click on Script icon

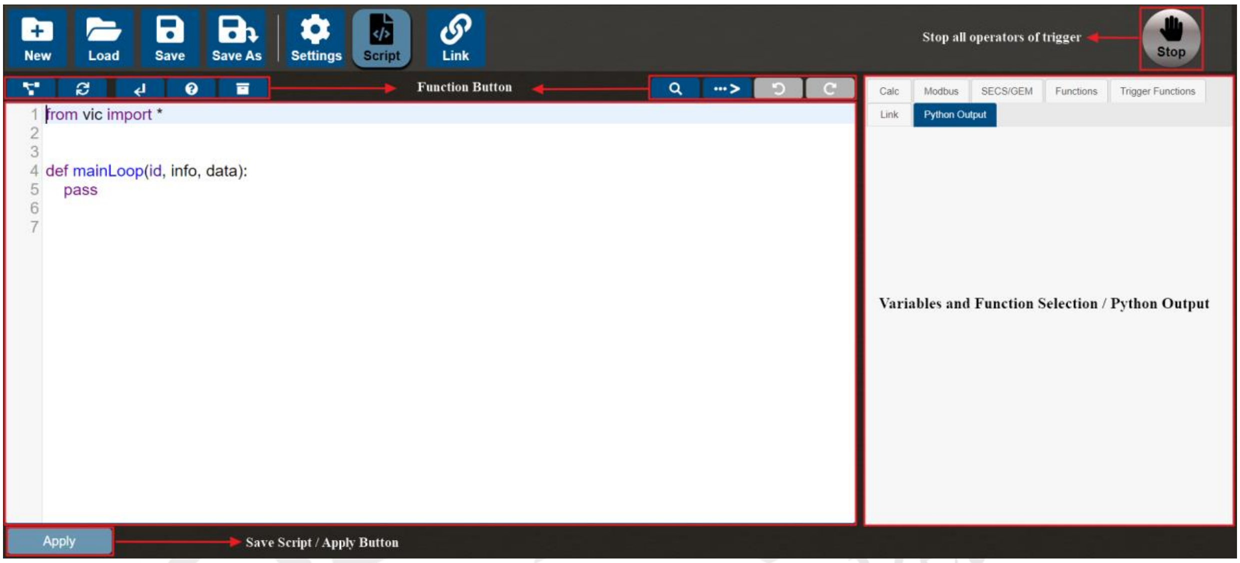


Figure 4-213 Script

4.2.1 Script Editing Area

Users can write scripts in the script editing area, and when they finish writing, they must click the "Apply" button, with the shortcut key being "Ctrl + S". If the syntax in the script is correct, the Python output field will display "Apply successfully", as shown in Figure 4-214. If there is a syntax error, the error message will show the line number, and the error may be on the displayed line number or the next line. In addition to the error information displayed in the Python output field, there will also be error markers on the line number, as shown in Figure 4-215.

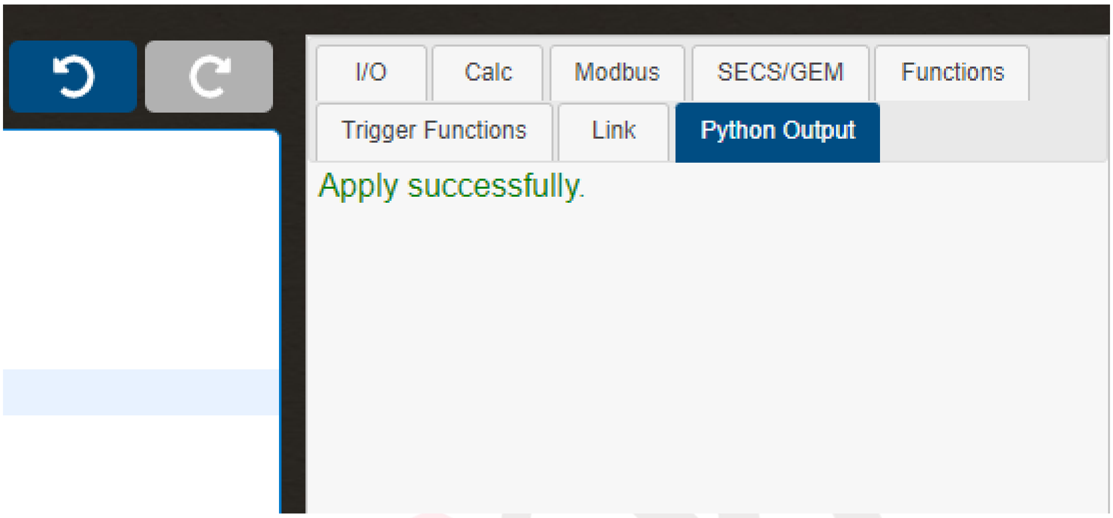


Figure 4-214 The syntax of the script is correct

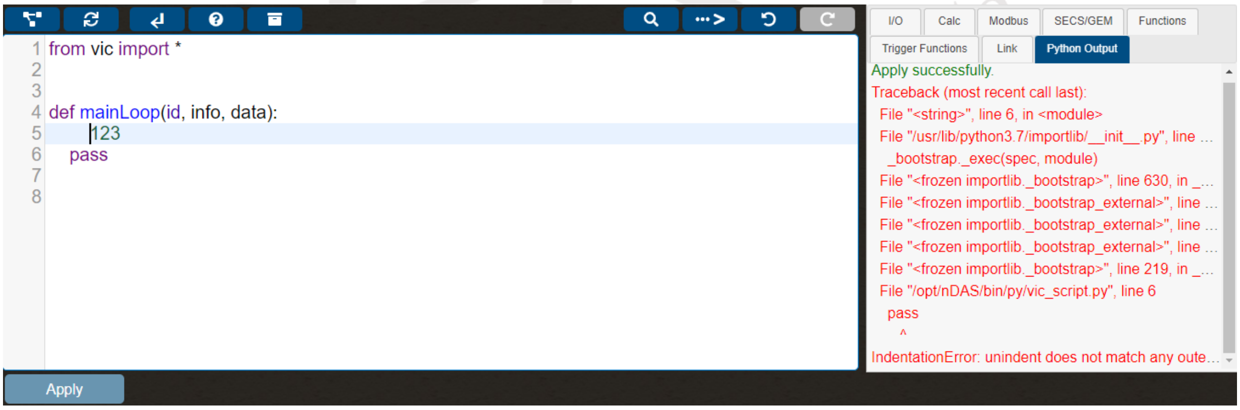


Figure 4-215 The syntax of the script is error

4.2.2 Input Area for Variables and Functions

The script page of this software has eight categories of variable and function input areas,namely I/O, CALC(Calculation operator), Modbus, SECS/GEM, function, trigger function, link, and Python output, as shown in Table 4-33.

Table 4-33 Script Variable Table

n(channeln), please refer to section 4.2.2.1.	
I/O	
DI_n	Status of DI channels
DO_n	Status of DO channels
AI_n	Status of AI channels
Calculations, please refer to section 4.2.2.2.	
CALC[0]	calculation variable 00
CALC[1]	calculation variable 01
CALC[2]	calculation variable 02
. .	. .
CALC[99]	calculation variable 99
Functions, please refer to section 4.2.2.3.	
PYTHON_MAIN_LOOP	Please refer to section 4.2.2.3.16 to generate default content for Python scripts.
LOG	Please refer to section 4.2.2.3.1 to log messages to a log.
LED	For controlling the machine's LED, please refer to section 4.2.2.3.2 for detailed instructions.
SEND.EMAIL	Please refer to section 4.2.2.3.5 to send an email.
SEND.LINE	Please refer to section 4.2.2.3.6 to send a Line message.
SEND.WECHAT	Please refer to section 4.2.2.3.7 to send a WeChat group message.
SEND.WECHAT_P	Please refer to section 4.2.2.3.8 to send a WeChat personal message: .

SEND.TEAMS	Please refer to section 4.2.2.3.9 to send a Microsoft Teams message.
OPCUA_ITEM	Retrieve the opcua_item and utilize it in other OPCUA methods as instructed in section 4.2.2.3.10.
OPCUA.READ	Obtain the OPCUA node status following the guidance in section 4.2.2.3.11.
OPCUA.WRITE	Write values to an OPCUA node by referring to section 4.2.2.3.12.
OPCUA.CALL_METHOD	Invoke an OPCUA Call Method using the instructions in section 4.2.2.3.13.
OPCUA.SUBSCRIBE	Subscribe to an OPCUA node by consulting section 4.2.2.3.14.
OPCUA.UNSUBSCRIBE	Cancel a subscription to an OPCUA node as outlined in section 4.2.2.3.15.
CURRENT_TIME	Please refer to section 4.2.2.3.3 for current time.
DB_IN	For writing values to the database, please refer to section 4.2.2.3.4 for detailed guidance.
WRITE_TEXT	Please refer to section 4.2.2.3.17 to write the contents into a text file.
WRITE_TEXT_LN	Please refer to section 4.2.2.3.18 to write the contents into a text file and perform a line break after writing.
SET_PREFERENCE	Please refer to section 4.2.2.3.19 to set a system variable that can still exist even after nDAS is restarted.
GET_PREFERENCE	Please refer to section 4.2.2.3.20 to obtain the value of a system variable, which will return a default value if the variable does not exist.
SHOW_DIALOG	Please refer to section 4.2.2.3.21 to display a popup dialog.

TRIGGER_OPERATOR	Please refer to section 4.2.2.3.22 to trigger the specified VIC flow trigger operator.
SECS/GEM, please refer to section 4.2.2.4.	
SET_ALARM	Please refer to section 4.2.2.4.1.1
CURRENT_CONTROL_ONLINE_MODE	Please refer to section 4.2.2.4.1.2
SET_CONTROL_OFFLINE	Please refer to section 4.2.2.4.1.3
SET_CONTROL_ONLINE	Please refer to section 4.2.2.4.1.4
SET_CONTROL_ONLINE_LOCAL	Please refer to section 4.2.2.4.1.5
SET_CONTROL_ONLINE_REMOTE	Please refer to section 4.2.2.4.1.6
TRIGGER_CEID	Please refer to section 4.2.2.4.1.7
SEND_S10F01	Please refer to section 4.2.2.4.1.8
OnSecsGemEquipS02F18	Please refer to section 4.2.2.4.1.9
OnSecsGemEquipS02F31	Please refer to section 4.2.2.4.1.10
OnSecsGemEquipS02F41	Please refer to section 4.2.2.4.1.11
OnSecsGemEquipS02F49	Please refer to section 4.2.2.4.1.12
OnSecsGemEquipS10F03	Please refer to section 4.2.2.4.1.13
ARE_YOU_THERE	Please refer to section 4.2.2.4.2.1
SEND_S02F17	Please refer to section 4.2.2.4.2.2
Trigger Functions, please refer to section 4.2.2.5	
OnInitScript	Please refer to section 4.2.2.5.1 to initialization script is triggered upon loading the project.
OnReloadFunction	Please refer to section 4.2.2.5.2 to click the reload button or press the shortcut key (F3) to reload.
OnSystemInit	Please refer to section 4.2.2.5.3 to the program opens and initialization is complete (automatically loading the project).
OnSystemExit	Please refer to section 4.2.2.5.4 to the program is closed.
OnBeforeTrigger	Trigger function is triggered when the trigger function is activated.
OnAfterTrigger	Trigger function is triggered after the

	trigger function is activated.
OnOpcuaDataChange	When the subscribed OPCUA node value changes within the script, trigger this function.
OnCalcWrite	When the RESTful API writes values to the CALC operator, trigger this function.
Modbus-Bit Channel, please refer to section 4.2.2.6.1	
BIT_CHANNEL[0][0]	Status of Modbus COM0 Bit channel 0.
BIT_ERR_CHANNEL[0][0]	Error status of Modbus COM0 Bit channel 0.
BIT_CHANNEL[0][1]	Status of Modbus COM0 Bit channel 1.
BIT_ERR_CHANNEL[0][1]	Error status of Modbus COM0 Bit channel 1.
. .	. .
BIT_CHANNEL[0][127]	Status of Modbus COM0 Bit channel 127.
BIT_ERR_CHANNEL[0][127]	Error status of Modbus COM0 Bit channel 127.
Modbus- Word Channel, please refer to section 4.2.2.6.2	
WORD_CHANNEL[0][0]	Status of Modbus COM0 Word channel 0.
WORD_ERR_CHANNEL[0][0]	Error status of Modbus COM0 Word channel 0.
WORD_CHANNEL[0][1]	Status of Modbus COM0 Word channel 1.
WORD_ERR_CHANNEL[0][1]	Error status of Modbus COM0 Word channel 1.
. .	. .
WORD_CHANNEL[0][127]	Status of Modbus COM0 Word channel 127
WORD_ERR_CHANNEL[0][127]	Error status of Modbus COM0 Word channel 127
Modbus-Modbus Bit, please refer to section 4.2.2.6.3	

MODBUSBIT[1]	Modbus Bit 01
MODBUSBIT[2]	Modbus Bit 02
MODBUSBIT[3]	ModbusBit 03
. .	. .
MODBUSBIT[65535]	ModbusBit 65535
Modbus- Modbus String, please refer to section 4.2.2.6.4.	
MODBUSSTR[1]	Modbus String 01
MODBUSSTR[2]	Modbus String 02
MODBUSSTR[3]	Modbus String 03
. .	. .
MODBUSSTR[65535]	Modbus String 65535
Modbus-Modbus Int, please refer to section 4.2.2.6.5.	
MODBUSINT[1]	Modbus Int 01
MODBUSINT[2]	Modbus Int 02
MODBUSINT[3]	Modbus Int 03
. .	. .
MODBUSINT[65535]	Modbus Int 65535
Modbus-Modbus Uint, please refer to section 4.2.2.6.6.	
MODBUSUINT[1]	Modbus Uint 01
MODBUSUINT[2]	Modbus Uint 02
MODBUSUINT[3]	Modbus Uint 03
. .	. .
MODBUSUINT[65535]	Modbus Uint 65535
Modbus-Modbus Uint16, please refer to section 4.2.2.6.7.	
MODBUSUINT16[1]	Modbus Uint16 01
MODBUSUINT16[2]	Modbus Uint16 02
MODBUSUINT16[3]	Modbus Uint16 03

.	.
.	.
MODBUSUINT16[65535]	Modbus Uint16 65535
Modbus-ModbusH Bit, please refer to section 4.2.2.6.8	
MODBUSHBIT[1]	Modbus Coils Bit 01
MODBUSHBIT [2]	Modbus Coils Bit 02
MODBUSHBIT [3]	Modbus Coils Bit 03
.	.
.	.
MODBUSHBIT [65535]	Modbus Coils Bit 65535
Modbus-ModbusH String, please refer to section 4.2.2.6.9	
MODBUSHSTR[1]	Modbus Holding String 01
MODBUSHSTR[2]	Modbus Holding String 02
MODBUSHSTR[3]	Modbus Holding String 03
.	.
.	.
MODBUSHSTR[65535]	Modbus Holding String 65535
Modbus-ModbusH Int, please refer to section 4.2.2.6.10	
MODBUSHINT[1]	Modbus Holding Int 01
MODBUSHINT[2]	Modbus Holding Int 02
MODBUSHINT[3]	Modbus Holding Int 03
.	.
.	.
MODBUSHINT[65535]	Modbus Holding Int 65535
Modbus-ModbusH Uint, please refer to section 4.2.2.6.11.	
MODBUSHUINT[1]	Modbus Holding Uint 01
MODBUSHUINT[2]	Modbus Holding Uint 02
MODBUSHUINT[3]	Modbus Holding Uint 03
.	.
.	.
MODBUSHUINT[65535]	Modbus Holding Uint 65535
Modbus-ModbusH Uint16，please refer to section 4.2.2.6.12。	

MODBUSHUINT16[1]	Modbus Holding Uint16 01
MODBUSHUINT16[2]	Modbus Holding Uint16 02
MODBUSHUINT16[3]	Modbus Holding Uint16 03
. .	. .
MODBUSHUINT16[65535]	Modbus Holding Uint16 65535
Modbus- Read System Modbus Bit, please refer to section 4.2.2.6.13.	
MODBUSSYSBIT[1]	Read Modbus System Bit 01
MODBUSSYSBIT[2]	Read Modbus System Bit 02
MODBUSSYSBIT[3]	Read Modbus System Bit 03
ModbusModbus Functions, please refer to section 4.2.2.6.14.	
MODBUS_MASTER_READ_BIT	Read Modbus Server Bit.
MODBUS_MASTER_WRITE_BIT	Write Modbus Server Bit.
MODBUS_MASTER_READ_STR	Rea dModbus Server String.
MODBUS_MASTER_WRITE_STR	Write Modbus Server String.
MODBUS_MASTER_READ_UINT	Read Modbus Server unsigned integers.
MODBUS_MASTER_WRITE_UINT	Write Modbus Server unsigned integers.
Link, please refer to section 4.2.2.7.	
TCP/IP Var Name	The TCP/IP variable name on the Link
Modbus Var Name	The Modbus variable name on the Link
SMTP Var Name	The SMTP variable name on the Link
LINE Var Name	The LINE variable name on the Link
Wechat Var Name	The Wechat variable name on the Link
Teams Var Name	The Teams variable name on the Link
OPCUA Client Var Name	The OPCUA variable name on the Link
Python Output	
Output of a Python script includes the content printed by “print” statements and any error messages.	

The aforementioned variables and functions can be selected not only in the input area of variables and functions, but also in the script editing area by right-clicking the mouse, as shown in Figure 4-216.

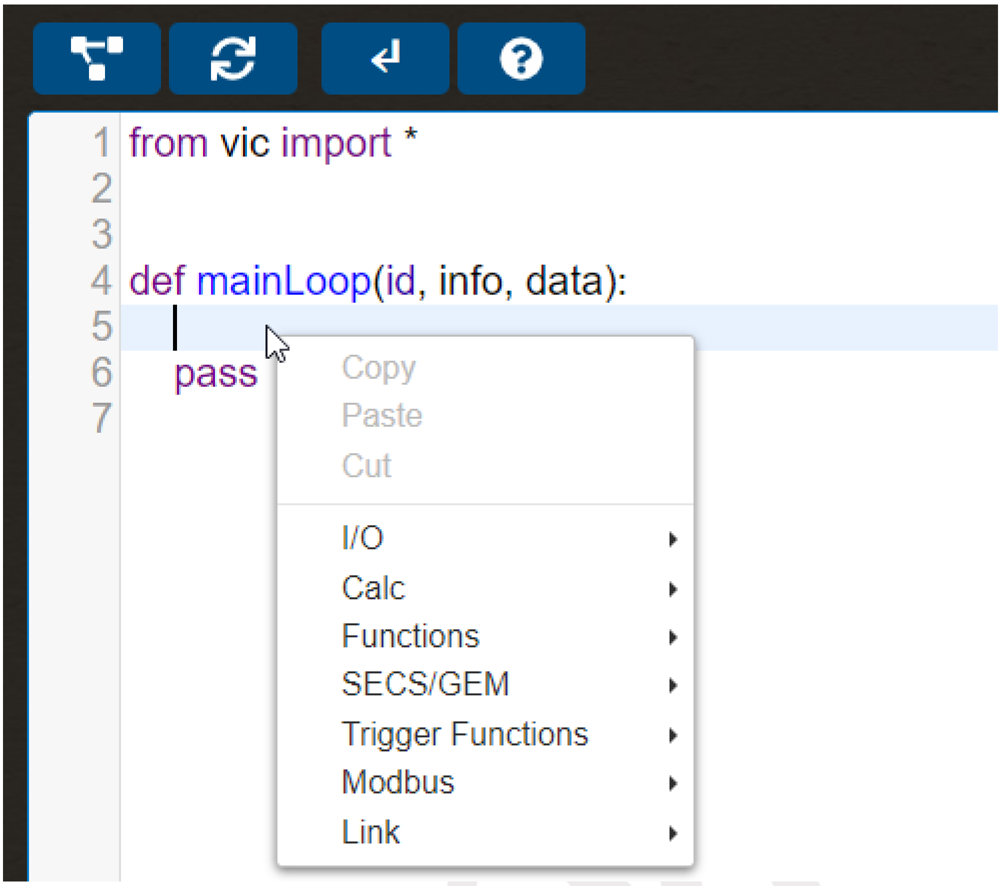


Figure 4-216 Selecting Variables and Functions by Right-Clicking the Mouse

4.2.2.1 I/O

To perform logical checks on I/O in the script, select I/O in the variables and function input area to display the list of I/O contained in nDAS (as shown in Figure 4-217).

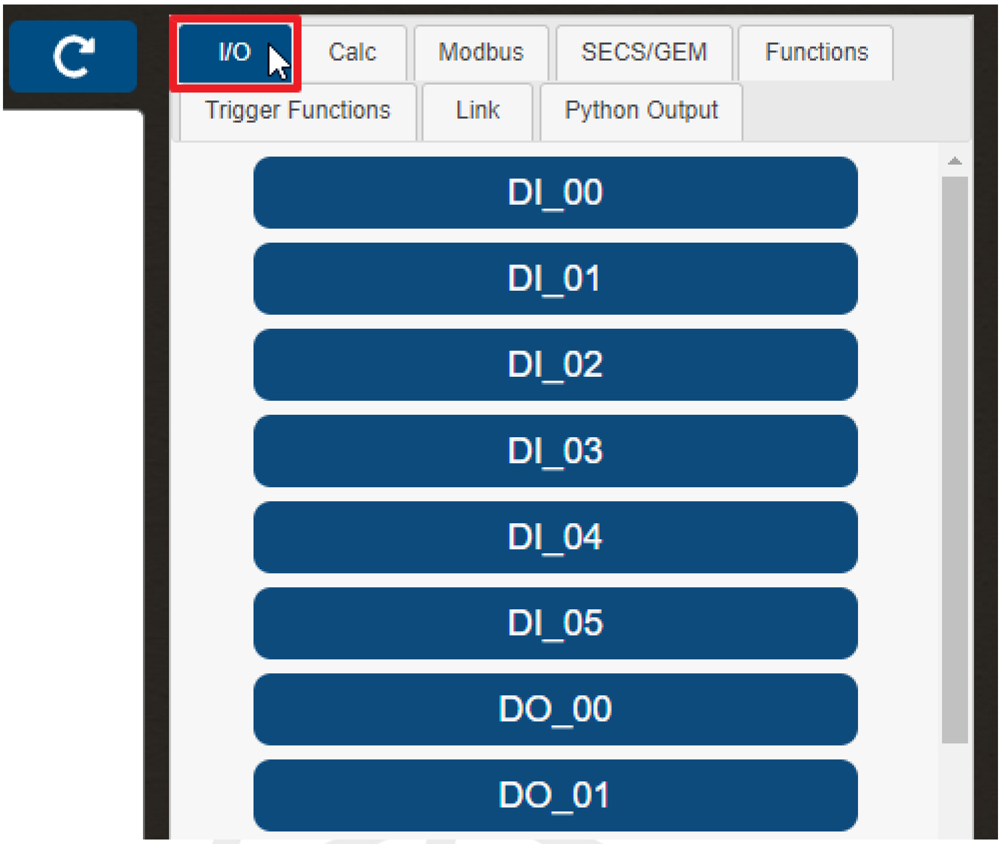


Figure 4-217 Selection of I/O for variables and function inputs

For example, if you want to print the status of DI channel 0 in the script execution result, you need to write a script in the script editing area, and the execution result is shown in Figure 4-218.

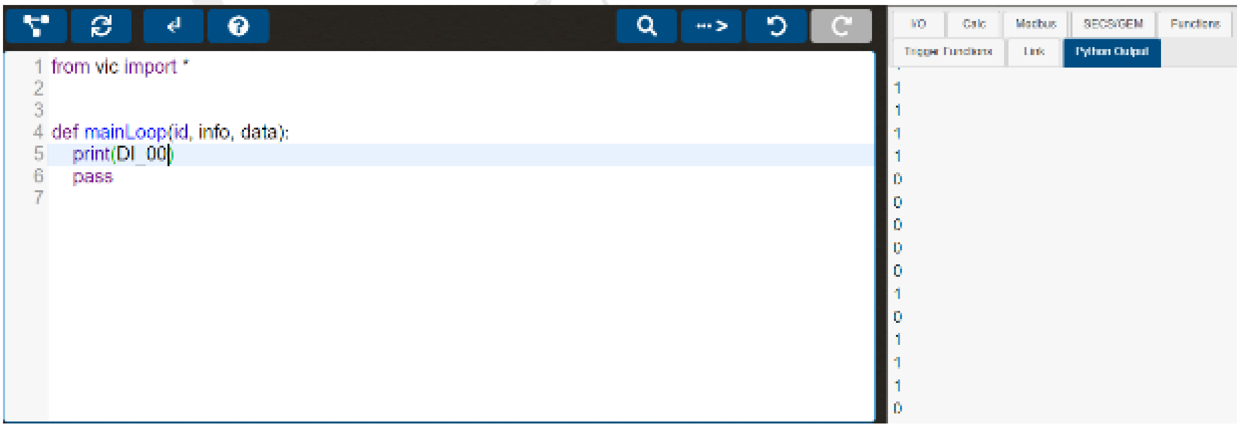


Figure 4-218 Print the status of DI channel 0

4.2.2.2 Calculation

To display script execution results in the interface, record them in a database, or transmit them through other means of transmission, it is necessary to use calculation variables. Select "Calculation Variables" in the Variables and Function Inputs section, which displays the calculation variables (as shown in Figure 4-219). There are a total of 100 calculation variables in this software, but only 20 (0-19) are displayed in the Variables and Function Inputs section.If the user needs to use calculation variables beyond the number displayed in the interface, they must input them manually by entering "CALC[n]", where n is a number between 0 and 99.

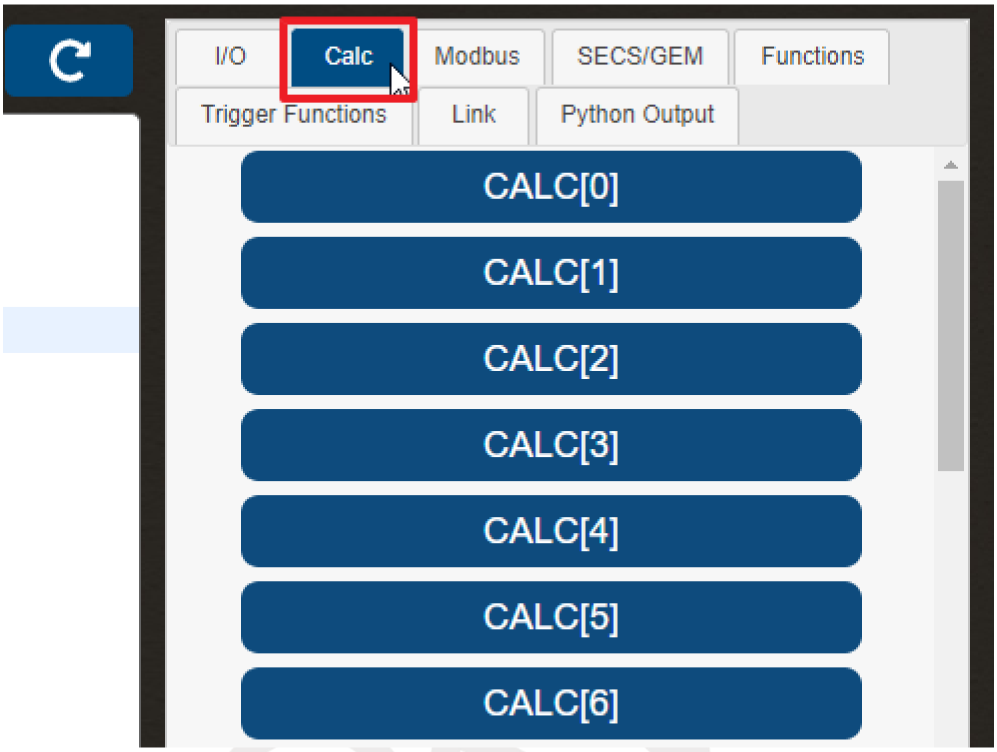


Figure 4-219 Selection of Calculation for variables and function inputs

For instance, if you want to display the status of analog input channel 0 in the script execution result, you need to write a script in the script page (as shown in Figure 4-220), and the execution result will be displayed as shown in Figure 4-221.

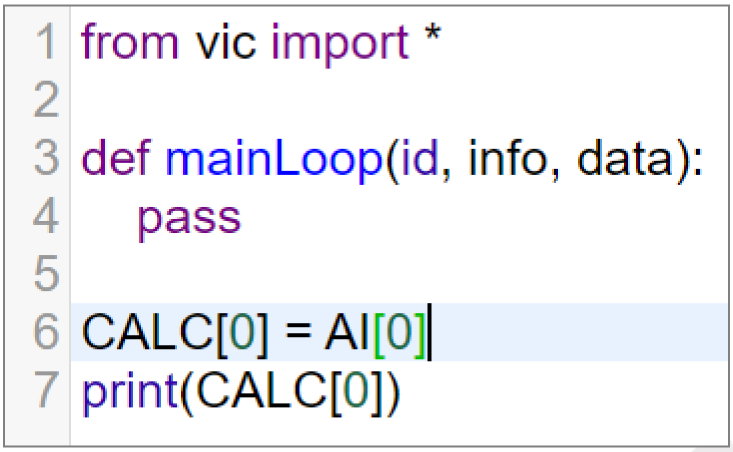


Figure 4-220 Example script writing

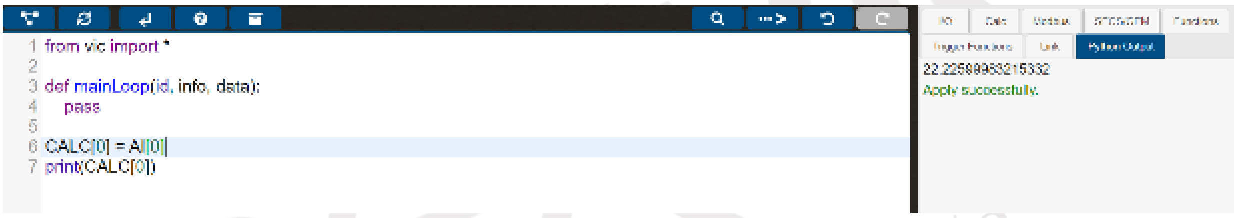


Figure 4-221 Result of script execution

4.2.2.3 Functions

Selecting a function from the variable and function input area will display the system script functions, as shown in Figure 4-222. These functions include SEND.EMAIL、LOG、SEND.LINE, and many others.

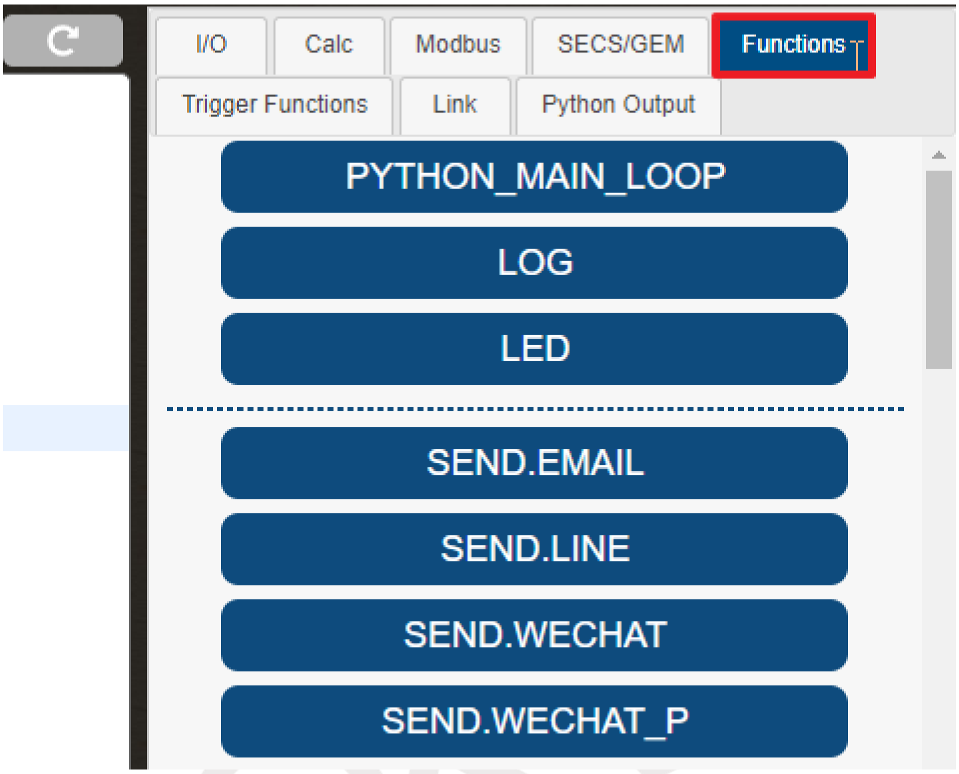


Figure 4-222 Selection of Functions for variables and function inputs

4.2.2.3.1 LOG

This function enables logging messages in the system log. When the LOG button is clicked in the variable and function input area (as shown in Figure 4-223), a LOG function is added to the script editing area (as shown in Figure 4-224).

LOG(type, message, sec) function needs to be configured with three parameters, representing the following meanings：

type：sets the color of the message recorded in the system log (0: black, 1: red, 2: blue).

message：sets the content of the message to be recorded in the system log.

sec：sets the time interval during which this function should not be executed. If set to 0, there is no time interval.

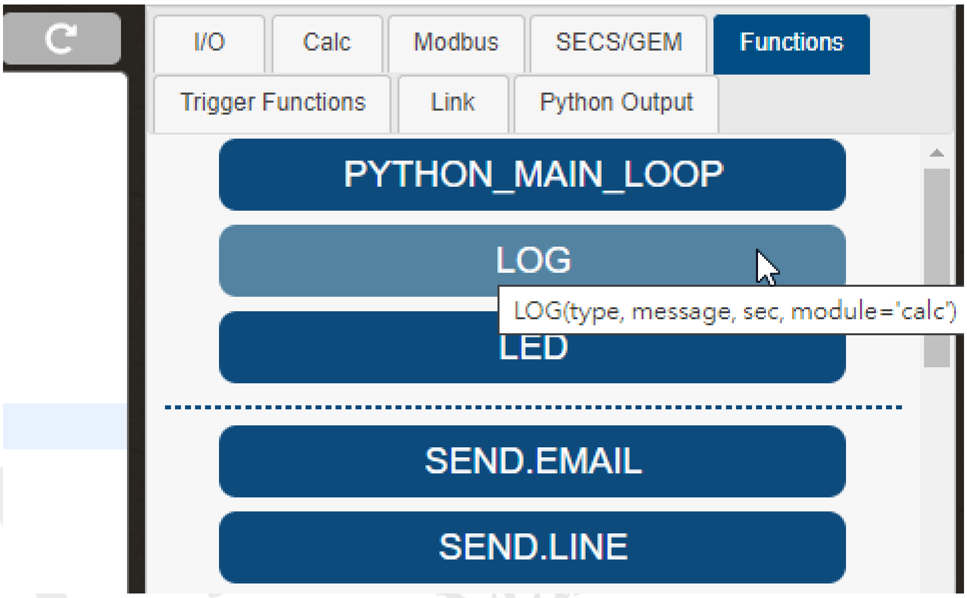


Figure 4-223 Click on “LOG”

```
def mainLoop(id, info, data):
    LOG( , , )
    pass

```

Figure 4-224 Add a LOG function in the script editing area

4.2.2.3.2 LED

This function is used to control the LED lights on the machine. By clicking the "LED" function in the Variables and Functions input section (as shown in Figure 4-225), an LED function will be added to the script writing area (as illustrated in Figure 4-226).

The LED(id, color, value) function requires three parameters, each representing the following:
- id: Parameter of type int, specifying the LED number to control.
- color: Parameter of type int, with possible values 0 or 1; 0 indicates green, and 1 indicates red.
- value: Parameter of type int, where 0 means off and 1 means on.

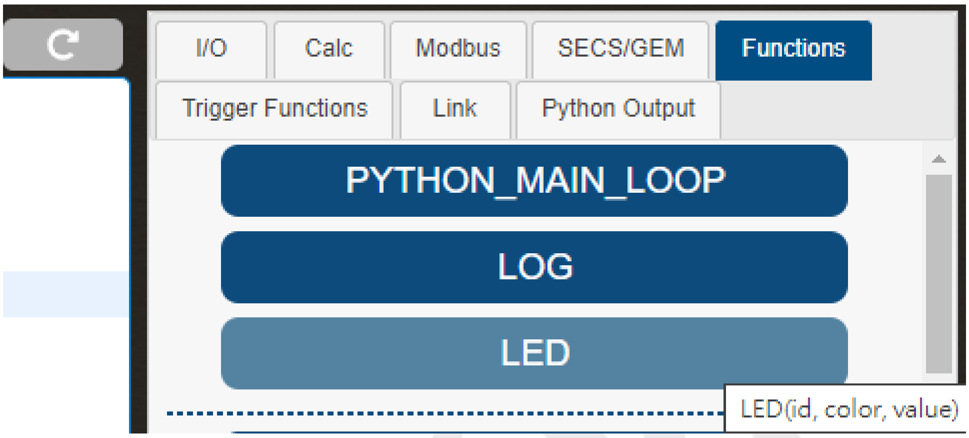


Figure 4-225 Click on “LED”

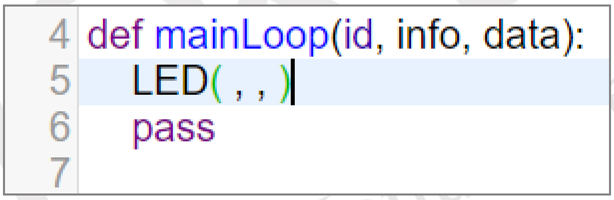


Figure 4-226 Add a LED function in the script editing area

Note: When controlling the LED, if nDAS is equipped with wireless Wi-Fi, it's necessary to disable the Wi-Fi first.

4.2.2.3.3 CURRENT_TIME

This function retrieves the current time.To access it, click on the CURRENT_TIME function in the variable and function input area (as shown in Figure 4-227), and the CURRENT_TIME function will be added to the script editing area (as shown in Figure 4-228). The execution result is shown in Figure 4-229.

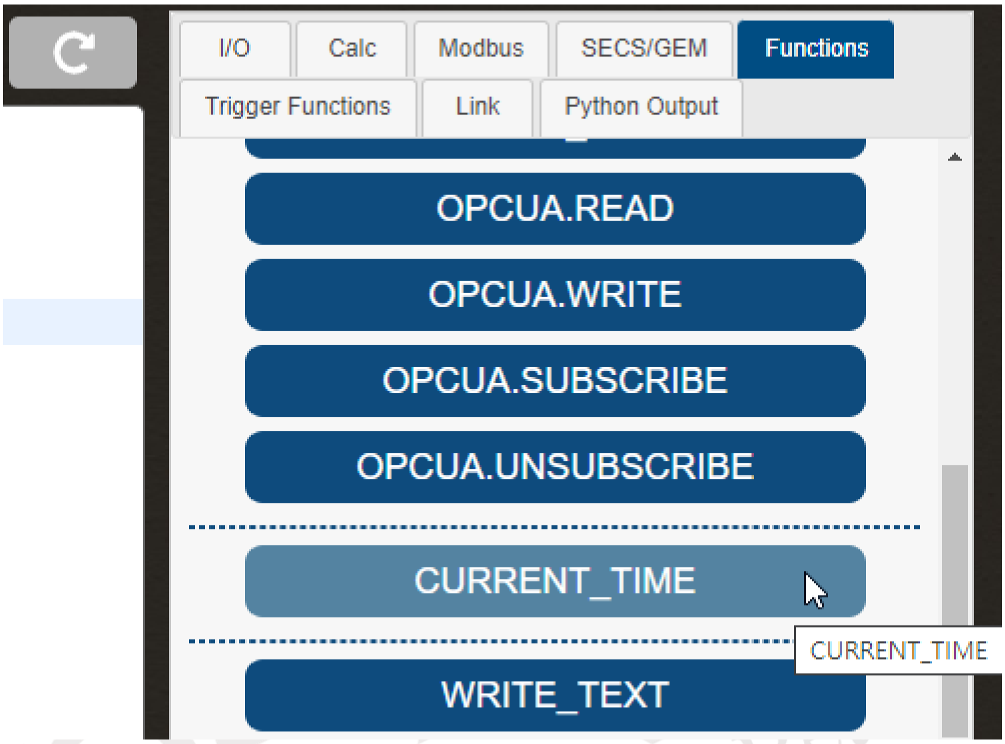


Figure 4-227 Click on “CURRENT_TIME”

Figure 4-228 Add a CURRENT_TIME function in the script editing area

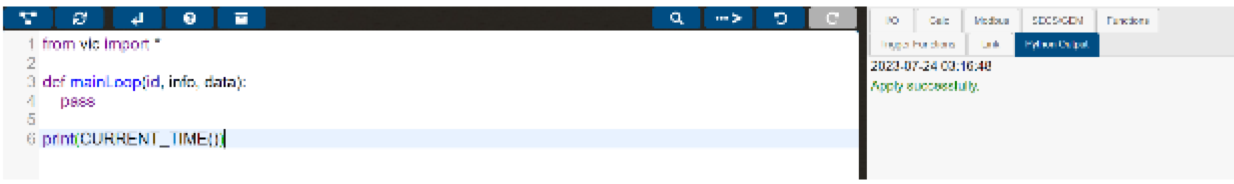


Figure 4-229 Result of CURRENT_TIME execution

4.2.2.3.4 DB_IN

This function is used to write data to the database, and the storage format should be referenced from section 4.1.4.2. By clicking the "DB_IN" function in the Variables and Functions input section (as shown in Figure 4-230), a DB_IN function will be added to the script writing area (as depicted in Figure 4-231).

The DB_IN(value, slot, channel, log_change) function requires four parameters, each representing the following:

- value: The value you intend to write to the database.
- slot: Slot ID, ranging from 0 to 1024.
- channel: Channel ID, ranging from 0 to 1024.
- log_change: Whether to enable recording when the value changes.

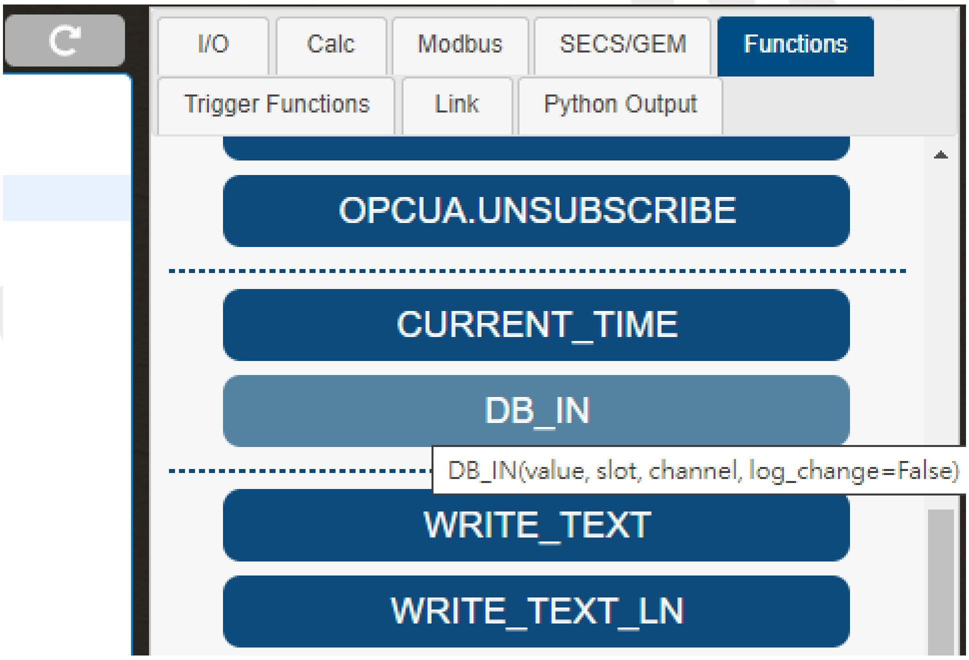


Figure 4-230 Click on "DB_IN"

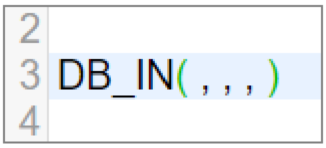


Figure 4-231 Add a DB_IN function in the script editing area

4.2.2.3.5 SEND.EMAIL

This function allows for setting up email messages to be sent to users. Prior to using the function, users must first set up an SMTP connection on the Link (please refer section 4.4.3 for instructions). Once this is complete, clicking on SEND.EMAIL in the function input area will add the SEND.EMAIL function to the script editing area (as shown in Figure 4-232)。

SEND.EMAIL(id, to, subject, body, sec) function requires five parameters, each representing the following：

id：Sets the SMTP connection to be used. Users can select the connection by choosing it from the function input area (as shown in Figure 4-233)and clicking on the desired SMTP connection (as shown in Figure 4-234), which will then be added to the script editing area (as shown in Figure 4-235).

to：The email address of the recipient.

subject：The subject of the email message.

body：The content of the email message.

sec：The time interval in seconds during which this funciton will not be executed again. If set to 0, there will be no time interval.

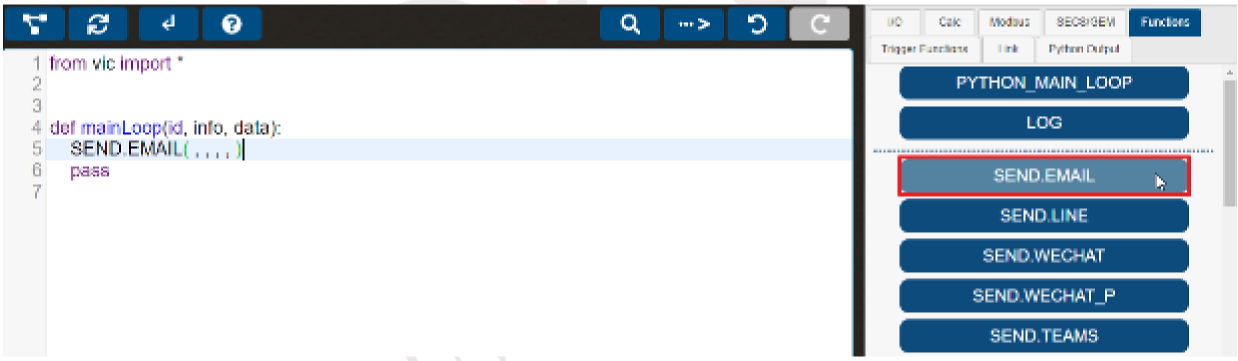


Figure 4-232 Click “SEND.EMAIL” and add the SEND.EMAIL function in the script editing area.

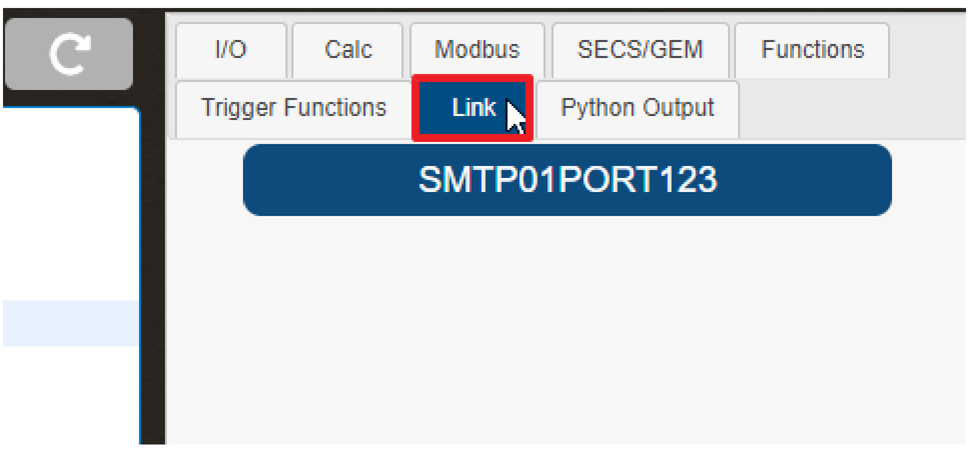


Figure 4-233 Choose the link in the variables and functions input area

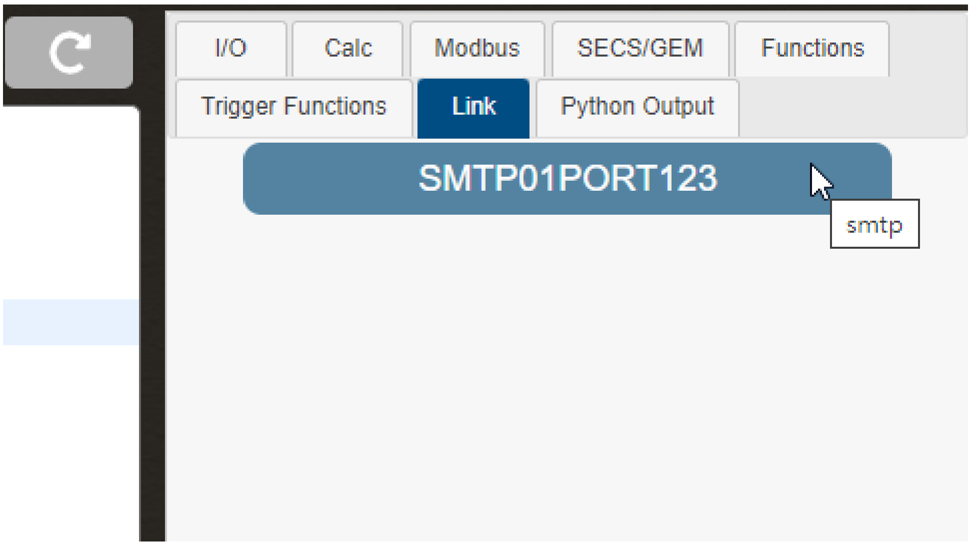


Figure 4-234 Click on the SMTP to be used.

```
def mainLoop(id, info, data):
    SEND.EMAIL("SMTP01PORT123" , , , , )
    pass

```

Figure 4-235 Add the SMTP in the script editing area.

4.2.2.3.6 SEND.LINE

This function enables the sending of messages to users through LINE. Prior to use, one must set up LINE Notify on the Link, please refer to section 4.4.4 for instructions. Once this is done, simply click on SEND.LINE in the variables and functions input area (as shown in Figure 4-236) to add the SEND.LINE function to the script editor (as shown in Figure 4-237)。

SEND.LINE (id, message, sec) function requires three parameters, each with the following meaning：

id：Sets the LINE Notify connection to use. To select a connection, choose from the options available in the varables and funcitons input area (as shown in Figure 4-238), and click on the desired LINE Notify connection (as shown in Figure 4-239) to add it to the script editor (as shown in Figure 4-240).

message：The content of the message to be sent.

sec：The time interval in seconds during which this funciton will not be executed again. If set to 0, there will be no time interval.

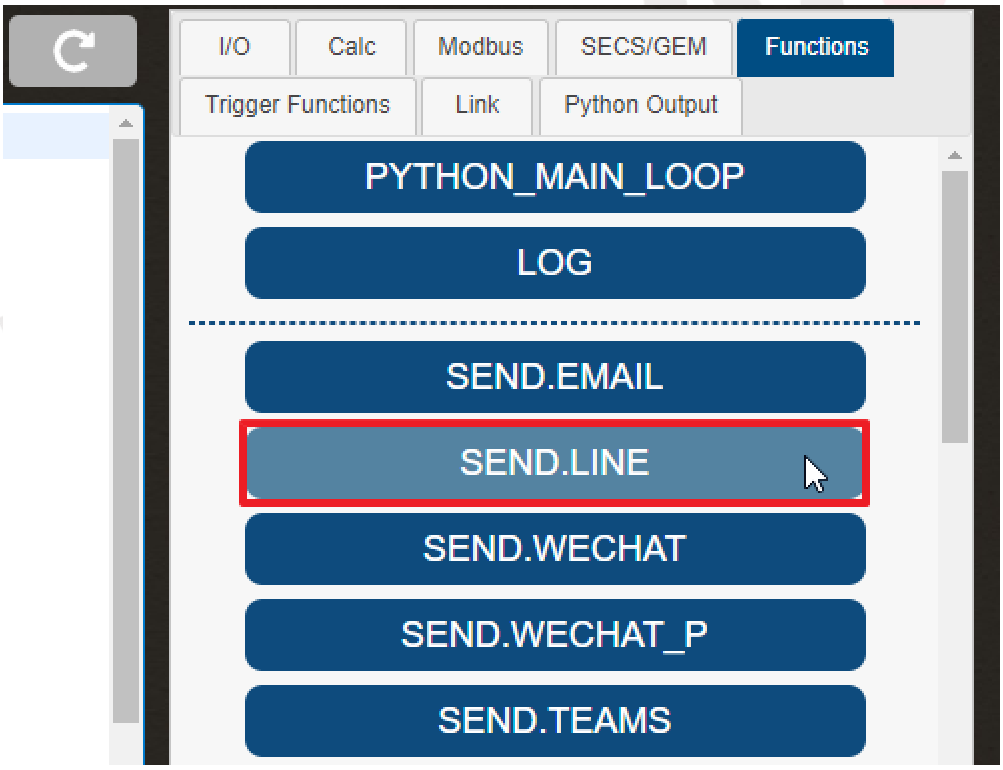


Figure 4-236 Click on “SEND.LINE”.

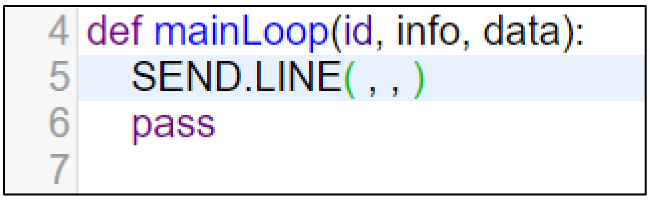


Figure 4-237 Add the SEND.LINE function in the script editing area.

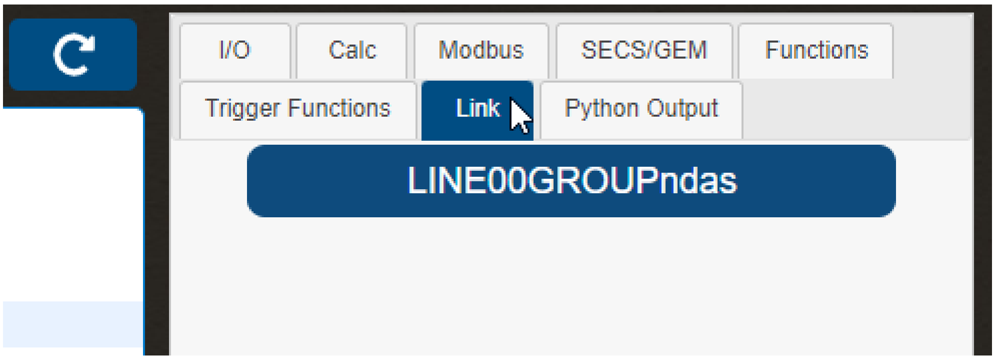


Figure 4-238 Choose the link in the variables and functions input area.

Figure 4-239 Click on the LINE Notify connection to be used.

```
def mainLoop(id, info, data):
    SEND.LINE("LINE00GROUPndas" , , )
    pass
```

Figure 4-240 Add the LINE Notify connection in the script editing area.

The example code transmits the status of DI channel 0 every one second. The script is written as shown in Figure 4-241. The received message at LINE is shown in Figure 4-242.

```
def mainLoop(id, info, data):
    SEND.LINE("LINE00GROUPndas" , "DI_00 = " + str(DI_00), 1)
    pass

```

Figure 4-241 Example script for SEND.LINE

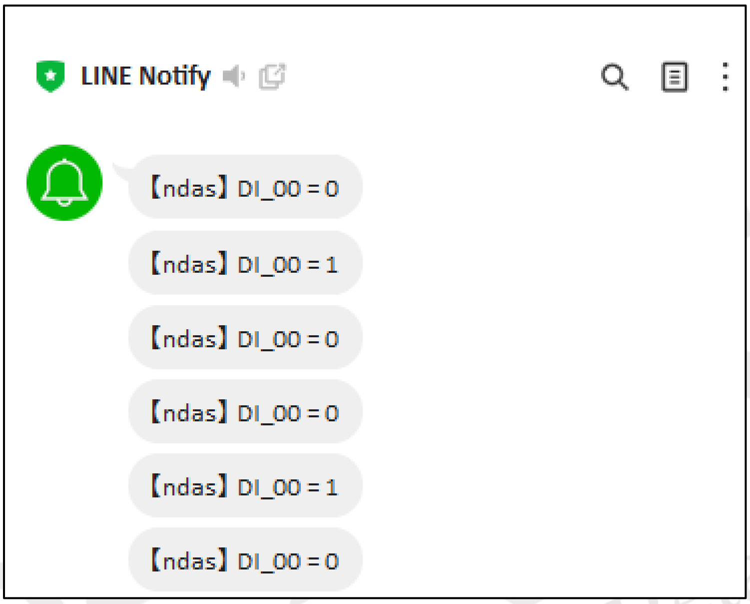


Figure 4-242 Message received by LINE in actuality

4.2.2.3.7 SEND.WECHAT

This function allows sending messages to every user who follows the test WeChat public account, and requires prior setup of Wechat (please refer to sectionn 4.4.5 for instructions). After completing the setup, clicking SEND.WECHAT in the function section of the variable and function input area (as shown in Figure 4-243) will add the SEND.WECHAT function to the script editor (as shown in Figure 4-244).

SEND.WECHAT (id, templateID, keyword1, keyword2, keyword3, sec) function requires six parameters, each representing the following：

id：sets the WeChat connection to use, selected from the connection list in the variable and function input area (as shown in Figure 4-245), and chosen by clicking on the desired connection (as shown in Figure 4-246), which will be added to the script editor (as shown in Figure 4-247).

templateID：sets the ID of the template message to be sent (please refer to section 5.4.1 to obtain instructions).

keyword1：sets the data to be sent as keyword1 content in the template message.

keyword2：sets the data to be sent as keyword2 content in the template message.

keyword3：sets the data to be sent as keyword3 content in the template message.

sec：The time interval in seconds during which this funciton will not be executed again. If set to 0, there will be no time interval.

Note：If keyword1 to keyword3 do not require data to be sent, please' ' (a space between single quotes) in the parameter field.

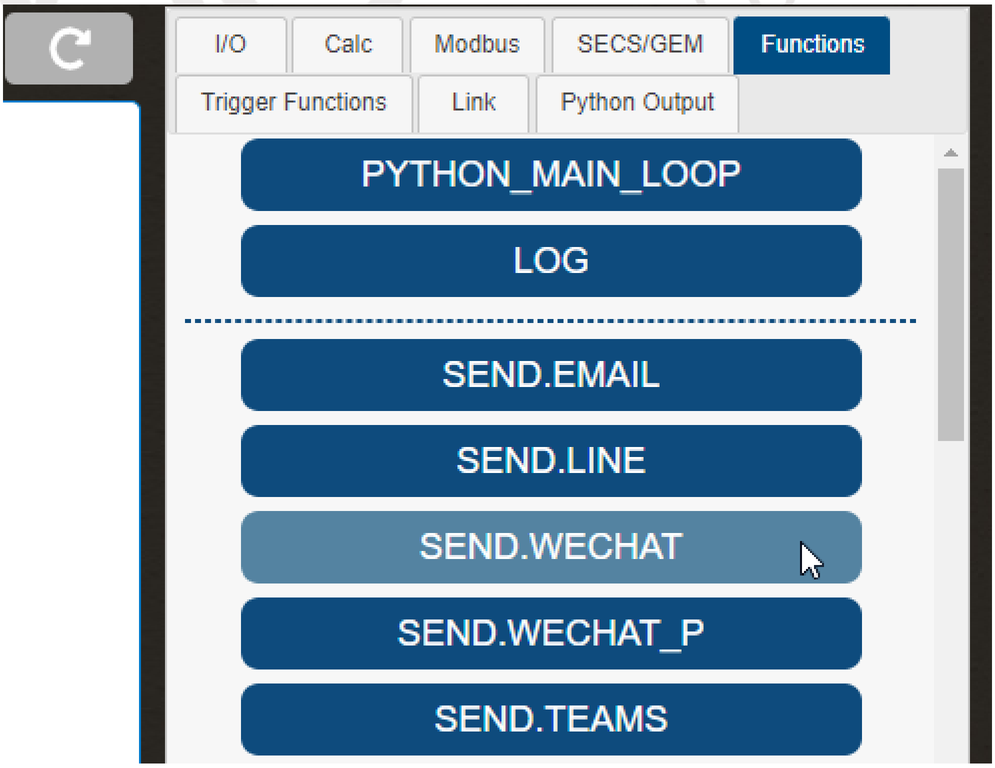


Figure 4-243 Click on “SEND.WECHAT”

```
def mainLoop(id, info, data):
    SEND.WECHAT( , , , , , )
    pass
```

Figure 4-244 Add the SEND.WECHAT function in the script editing area

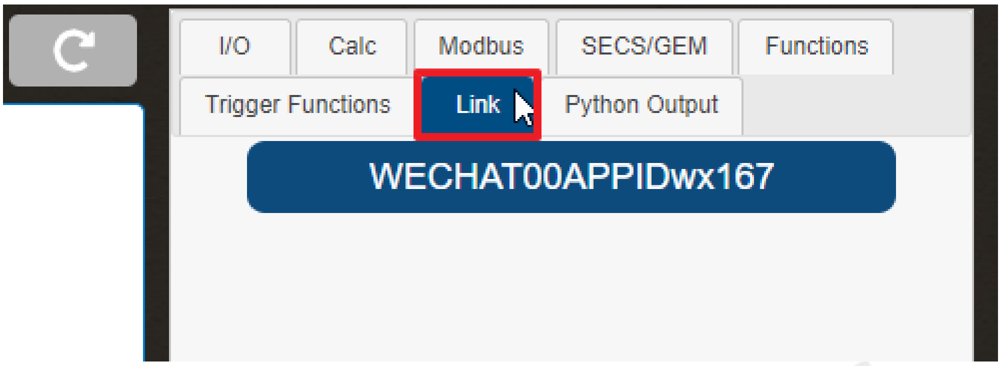


Figure 4-245 Choose the link in the variables and functions input area.

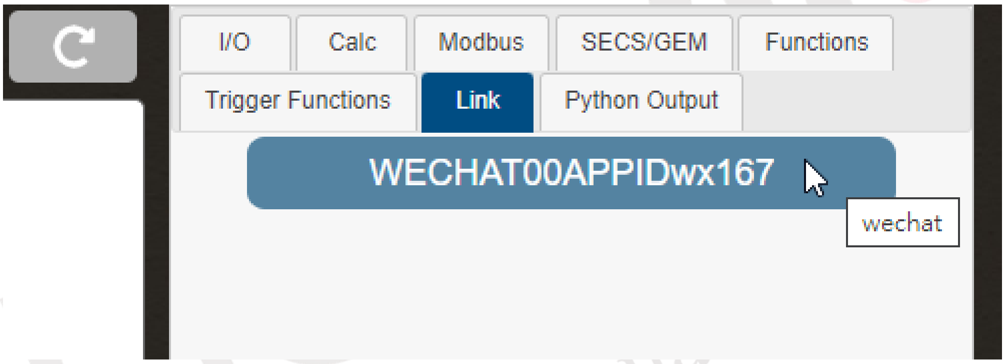


Figure 4-246 Click on the Wechat connection to be used.

```
def mainLoop(id, info, data):
    SEND.WECHAT("WECHAT00APPIDwx167" , , , , , )
    pass

```

Figure 4-247 Add the Wechat connection in the script editing area

The example transmits data with keywords “keyword1” to “keyword3”, representing the status of DI channel 0 to 2, and sends the message every five seconds. The script is written as shown in Figure 4-248, and the actual message received on WeChat is shown in Figure 4-249.

```
ID = "WECHAT00APPIDwx167"
TempID = "Q6MODwbfIIlon9Ik3tps0zQP2ib3Zyp1Xxg8qlINlb4"

sec = 5

def mainLoop(id, info, data):
    keyword01 = "DI_00值為 " + str(DI_00)
    keyword02 = "DI_01值為 " + str(DI_01)
    keyword03 = "DI_02值為 " + str(DI_02)
    SEND.WECHAT(ID , TempID, keyword01, keyword02, keyword03, sec)
    pass

```

Figure 4-248 Example script for SEND.WECHAT

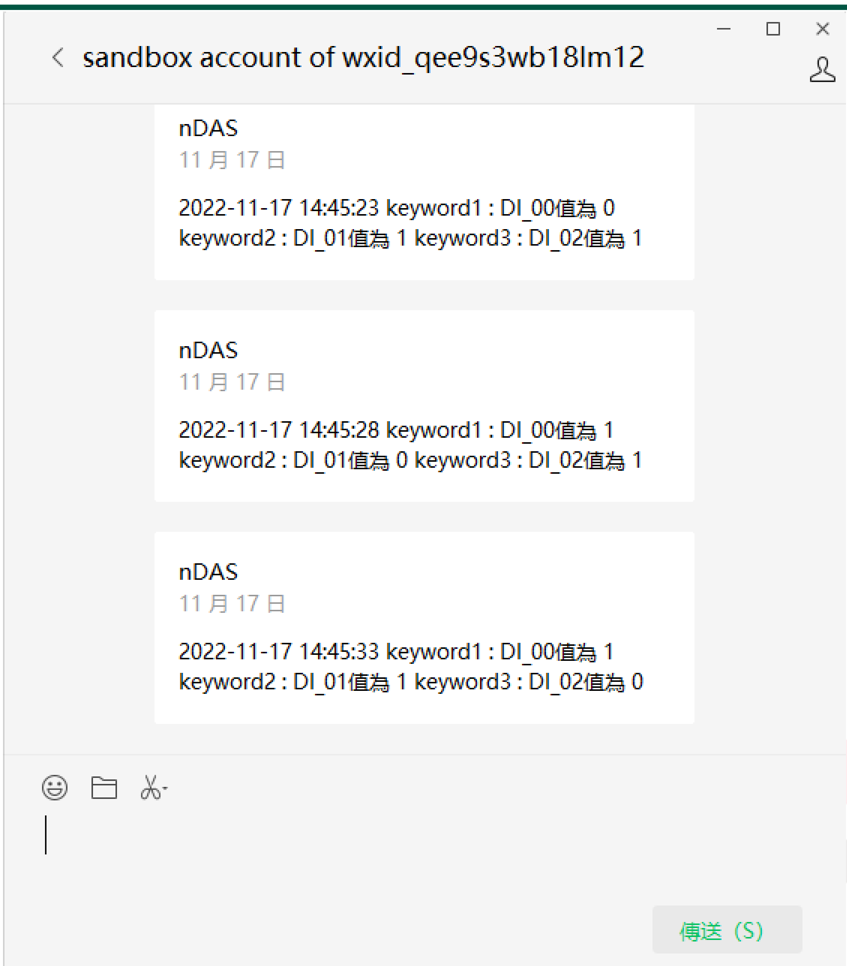


Figure 4-249 Message received by WeChat in actuality

4.2.2.3.8 SEND.WECHAT_P

This function can be used to send messages to specific users who have subscribed to the test public account on WeChat. To use this function, you need to first set up the WeChat connection on the Link (please refer to section 4.4.5 for instructions). Once the setup is complete, click on the SEND.WECHAT_P function in the variable and function input area (as shown in Figure 4-250)to add the SEND.WECHAT_P function to the script editor (as shown in Figure 4-251).

SEND.WECHAT_P (id, templateID, openID, keyword1, keyword2, keyword3, send image, sec) function requires seven parameters to be set, each representing the following：

id：sets the Wechat connection to use, selected from the connection list in the variable and function input area (as shown in Figure 4-252), and chosen by clicking on the desired connection (as shown in Figure 4-253), which will be added to the script editor (as shown in Figure 4-254).

templateID：sets the ID of the template message to be sent (please refer to secntion 5.4.1 to obtain instructions).

openID：To transmit the WeChat ID of a user who follows the public account, please refer to section 5.4.2 for the method to obtain it.

keyword1：sets the data to be sent as keyword1 content in the template message.

keyword2：sets the data to be sent as keyword2 content in the template message.

keyword3：sets the data to be sent as keyword3 content in the template message.

sec：The time interval in seconds during which this funciton will not be executed again. If set to 0, there will be no time interval.

If keyword1 to keyword3 do not require data to be sent, please' ' (a space between single quotes) in the parameter field.

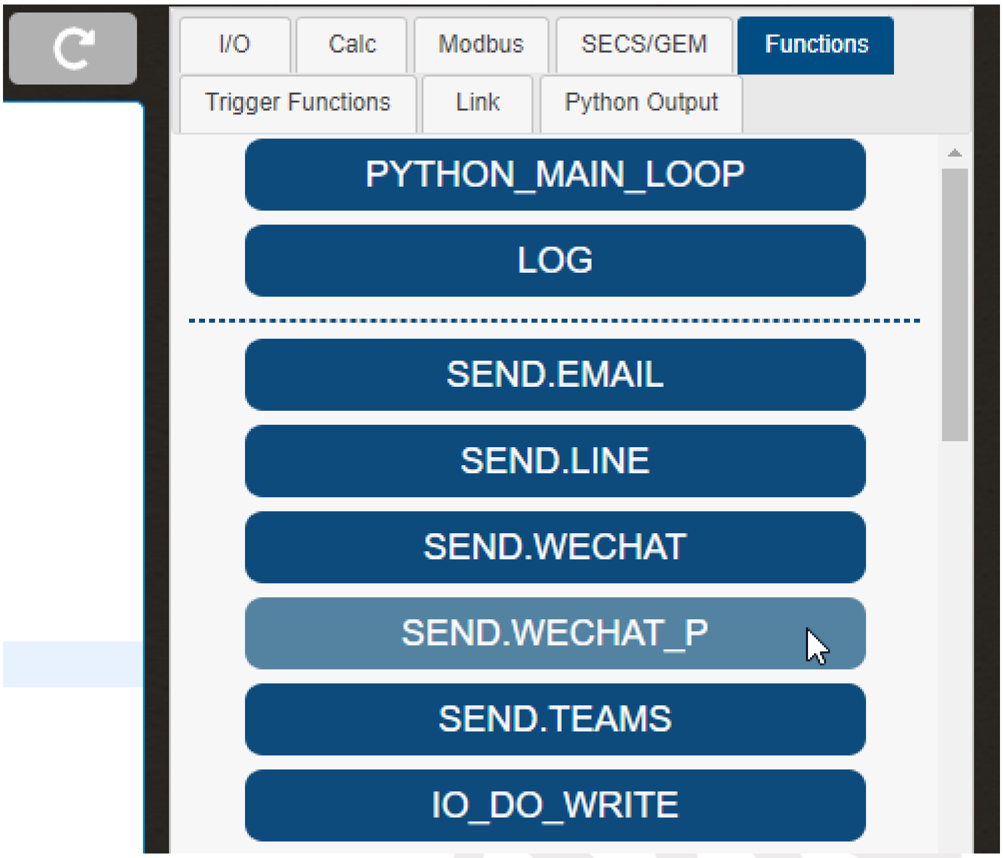


Figure 4-250 Click on “SEND.WECHAT_P”.

```
def mainLoop(id, info, data):
    SEND.WECHAT_P( , , , , , , )
    pass

```

Figure 4-251 Add the SEND.WECHAT_P fucntion in the script editing area.

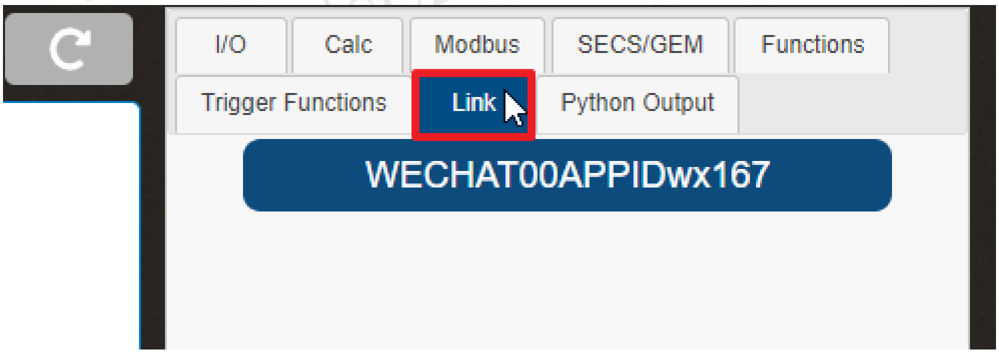


Figure 4-252 Choose the link in the variables and functions input area

Figure 4-253 Click on the Wechat to be used.

```
def mainLoop(id, info, data):
    SEND.WECHAT_P("WECHAT00APPIDwx167" , , , , , , )
    pass
```

Figure 4-254 Add Wechat connection in the script editing area

The example transmits data with keyword1 to keyword3, representing the status of DI channel 0 to 2, and sends the message every five seconds. The script is written as shown in Figure 4-255, and the actual message received on WeChat is shown in Figure 4-256.

```
ID = "WECHAT00APPIDwx167"
TempID = "Q6MODwbfIlIon9Ik3tps0zQP2ib3Zyp1Xxg8qlINlb4"
OpenID = "oBFdB58-X2xfVeKXFEDcyO0pRzMw"
sec = 5

def mainLoop(id, info, data):
    keyword01 = "DI_00值為 " + str(DI_00)
    keyword02 = "DI_01值為 " + str(DI_01)
    keyword03 = "DI_02值為 " + str(DI_02)
    SEND.WECHAT_P(ID , TempID,OpenID , keyword01, keyword02, keyword03, sec)
    pass
```

Figure 4-255 Example script for SEND.WECHAT_P

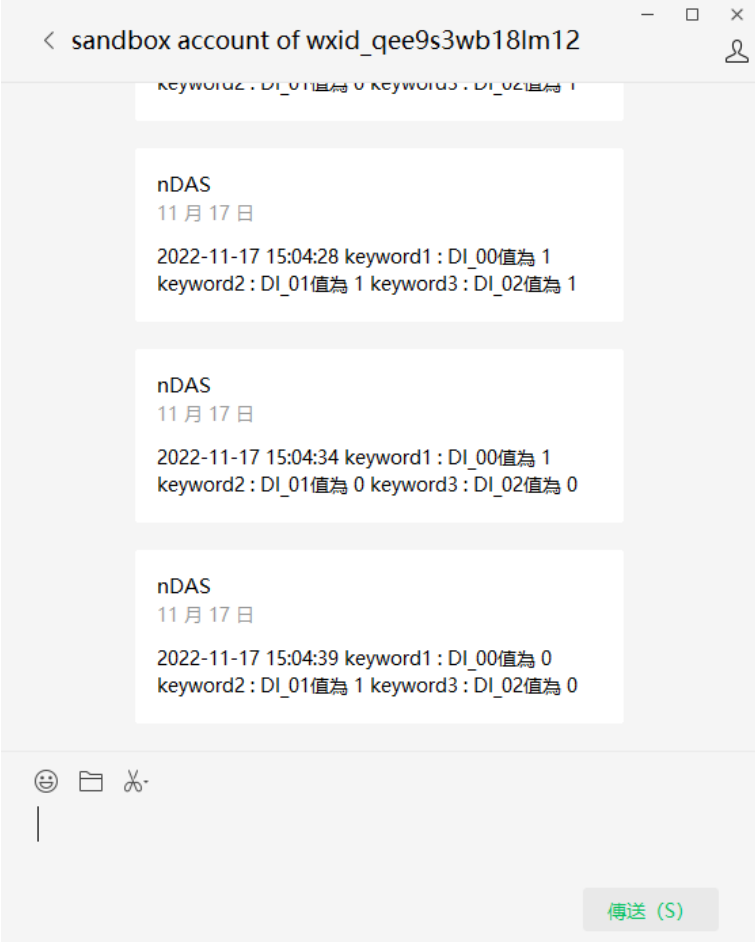


Figure 4-256 Message received by WeChat in actuality

4.2.2.3.9 SEND.TEAMS

This function allows you to send messages to users via Microsoft Teams. Before using it, you need to set up the Teams connection on the Link (please refer to section 4.4.6 for instructions). Once done, clicking on SEND.TEAMS in the variables and functions input area (as shown in Figure 4-257) will add the SEND.TEAMS function to the script editor (as shown in Figure 4-258).

SEND. TEAMS (id, title, message, send image, sec) function requires four parameters, each representing the following：

id：Sets the Teams connection to be used. To select a connection, choose a link in the variables and functions input area (as shown in Figure 4-259), and click on the desired Teams connection (as shown in Figure 4-260), which will be added to the script editor (as shown in Figure 4-261).

title：The title of the message to be sent.

message：The content of the message to be sent.

sec：The time interval in seconds during which this funciton will not be executed again. If set to 0, there will be no time interval.

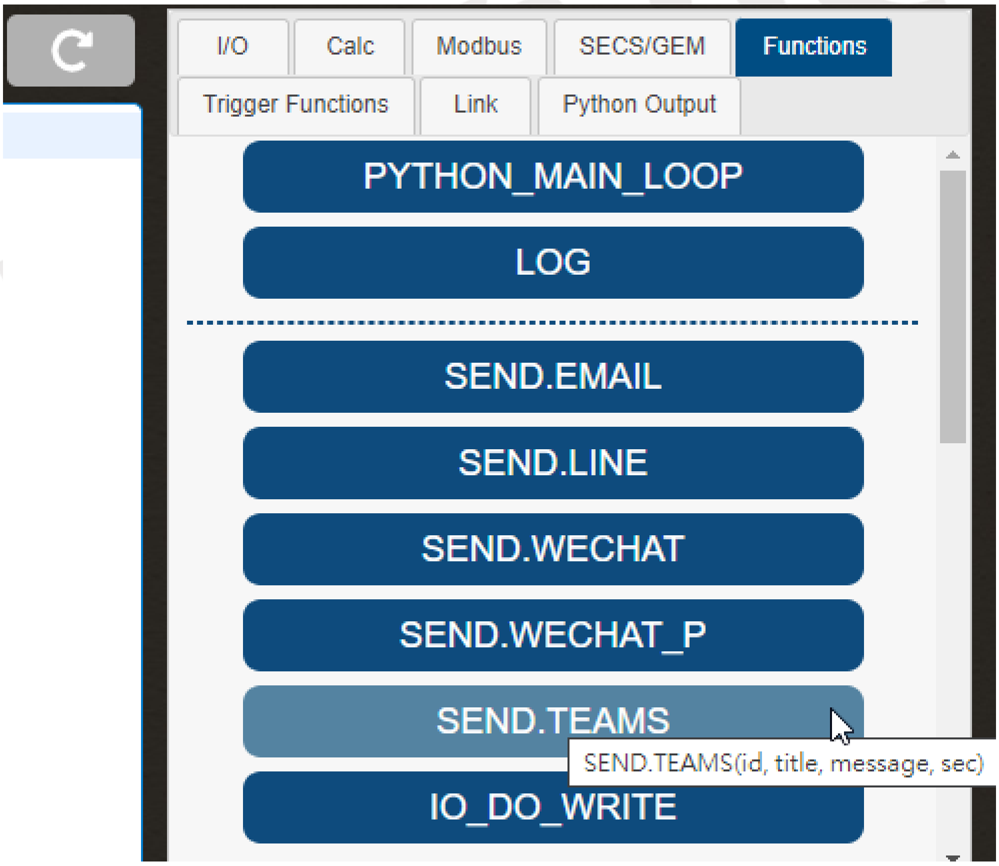


Figure 4-257 Click on “SEND.TEAMS”

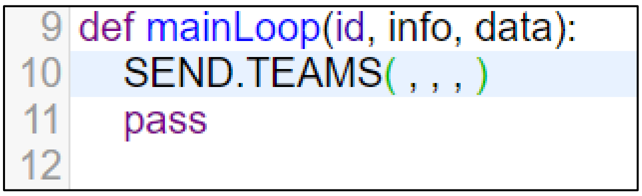


Figure 4-258 Add the SEND.TEAMS function in script editing area

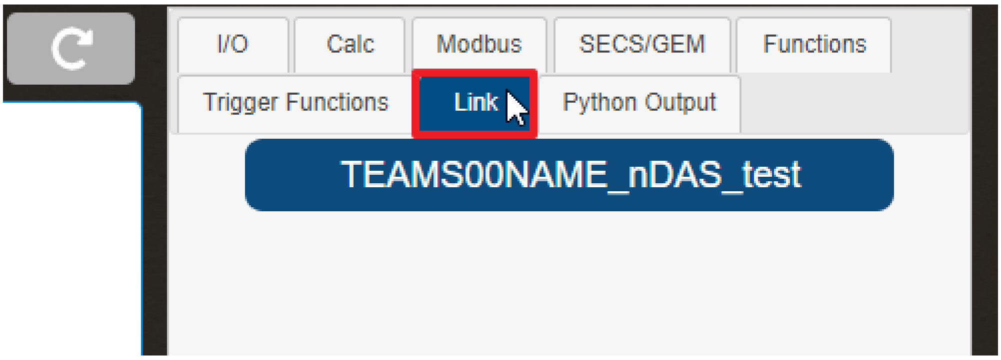


Figure 4-259 Choose the link in the variables and functions input area

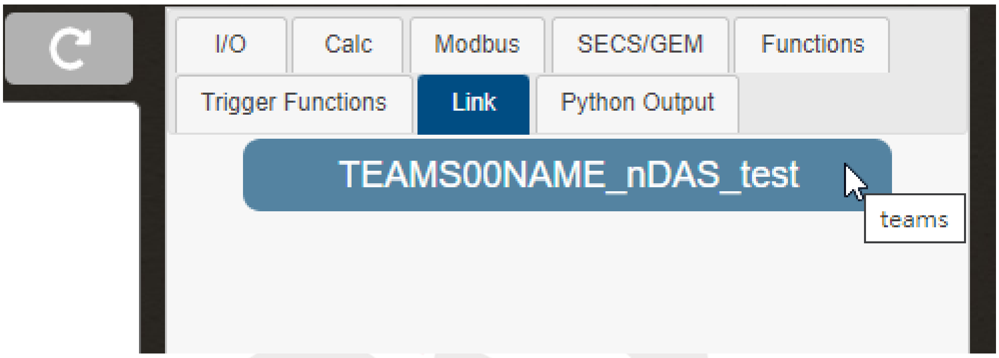


Figure 4-260 Click on the Teams connection to be used.

```
def mainLoop(id, info, data):
    SEND.TEAMS("TEAMS00NAME_nDAS_test", , , )
    pass

```

Figure 4-261 Add the Teams connection in the script editing area.

This is an example of sending periodic messages every 5 seconds, as shown in Figure 4-262. The message is actually received by Microsoft Teams, as shown in Figure 4-263.

```
def mainLoop(id, info, data):
    SEND.TEAMS("TEAMS00NAME_nDAS_test" ,"nDAS_demo" , "DI_00 = " + str(DI_00), 5)
    pass


```

Figure 4-262 Example script for SEND.TEAMS

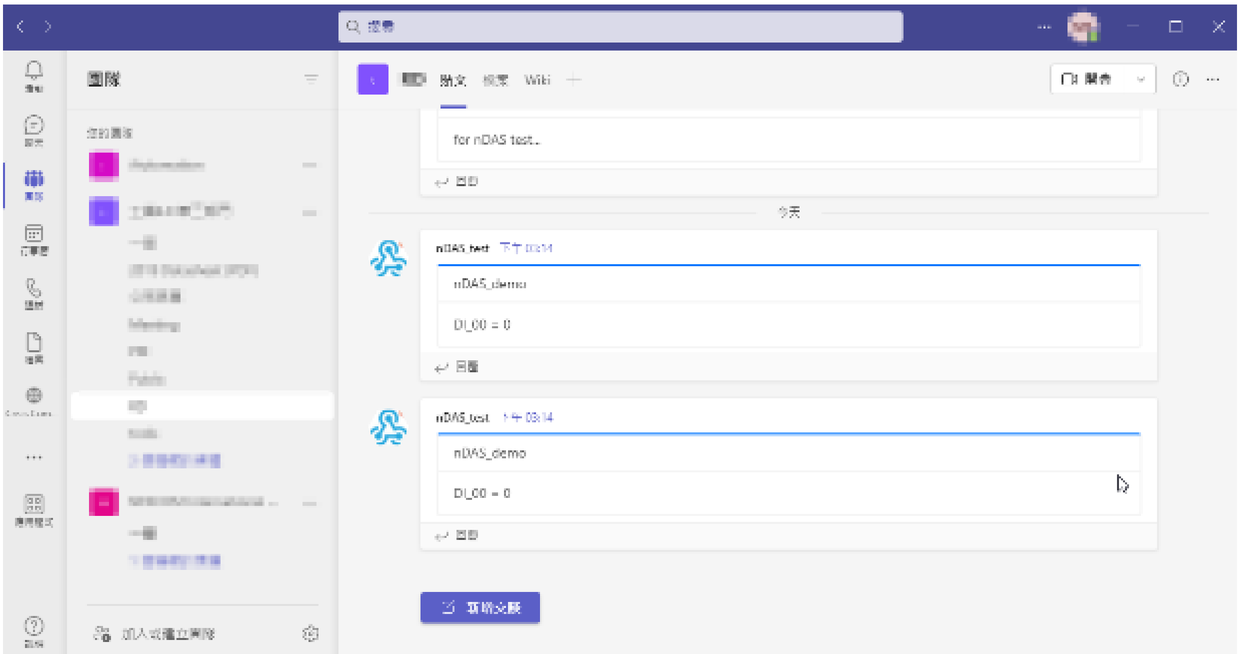


Figure 4-263 Message received by Microsoft Teams in actuality

4.2.2.3.10 OPCUA_ITEM

This function facilitates the retrieval of nodes from the OPCUA Server. Prior to its utilization, it's necessary to navigate to the linking page and configure the connection to the OPCUA Server (refer to section 4.4.7 for configuration details). Upon completing the configuration, within the variable and function input section, by selecting the OPCUA_ITEM function (as illustrated in Figure 4-264), the software will seamlessly integrate the OPCUA_ITEM function into the script-writing area (as depicted in Figure 4-265).

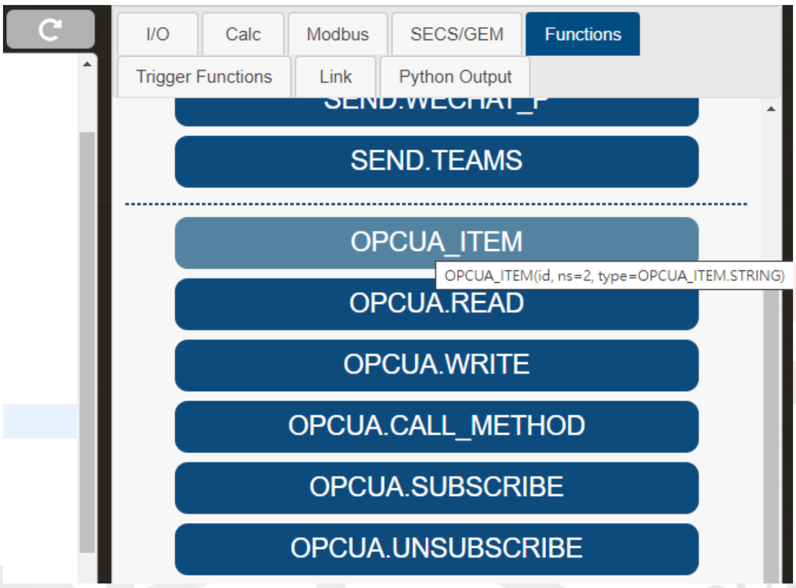


Figure 4-264 Click on "OPCUA_ITEM"

```
item = OPCUA_ITEM("DI_00")
# OPCUA.READ
print(OPCUA.READ("OPCUA00_127_0_0_1_48010" , item))
```

Figure 4-265 Add the OPCUA_ITEM function in script editing area

This function is employed in cases where other OPCUA methods necessitate the inclusion of an "item" type. In such scenarios, it becomes essential to input the name of the OPCUA node. As an illustrative example involving the nDAS OPCUA Server, users can proactively examine the node names provided by nDAS using the OPCUA Client within VIC Flow. This can be accomplished by referring to Figure 4-266 for guidance.

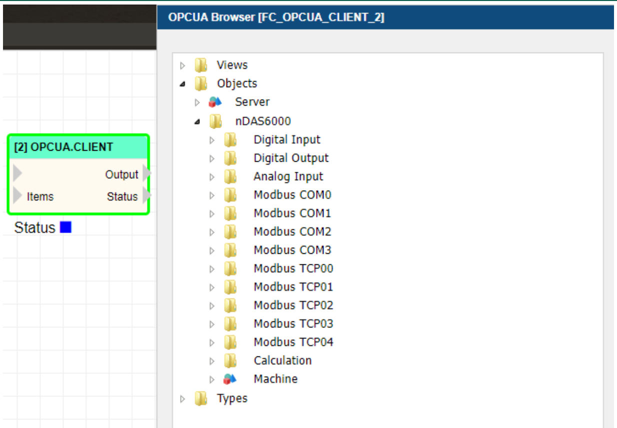


Figure 4-266 Browse all node by OPCUA Client operator of VIC Flow

4.2.2.3.11 OPCUA.READ

This function is designed to retrieve the node status from the OPCUA Server. Before utilizing it, navigate to the linking page and configure the connection to the OPCUA Server (refer to section 4.4.7 for configuration details). Upon completing the configuration, within the variable and function input section, click on the OPCUA.READ function (as depicted in Figure 4-267). This action will seamlessly integrate the OPCUA.READ function into the script-writing area. After inputting the requisite parameters, as shown in Figure 4-268, executing the script will yield results that are subsequently printed, as illustrated in Figure 4-269.

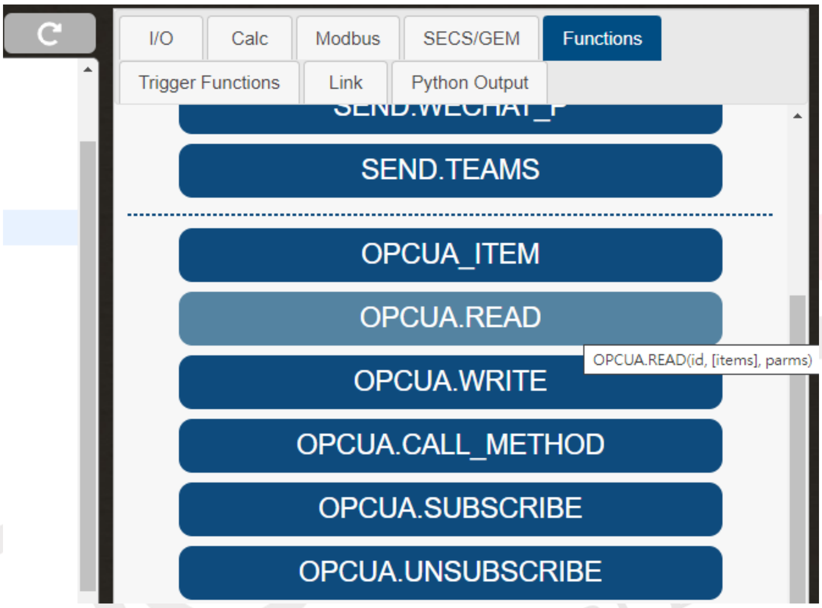


Figure 4-267 Click on "OPCUA.READ"

```
item = OPCUA_ITEM("DI_00")
# OPCUA.READ
print(OPCUA.READ("OPCUA00_127_0_0_1_48010" , item))
```

Figure 4-268 Add the OPCUA.READ function in script editing area

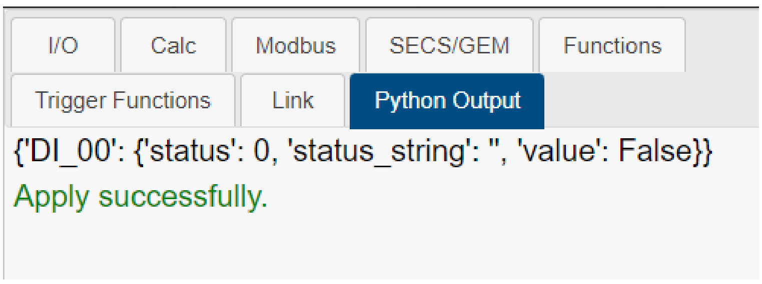


Figure 4-269 Result of OPCUA.READ

4.2.2.3.12 OPCUA.WRITE

This function facilitates the writing of values to nodes on the OPCUA Server. To use it, first navigate to the linking page and configure the connection to the OPCUA Server (refer to section 4.4.7 for configuration details). After completing the configuration, within the variable and function input section, click on the OPCUA.WRITE function (as depicted in Figure 4-270). This action will seamlessly integrate the OPCUA.WRITE function into the script-writing area. Subsequently, input the required parameters as illustrated in Figure 4-271.

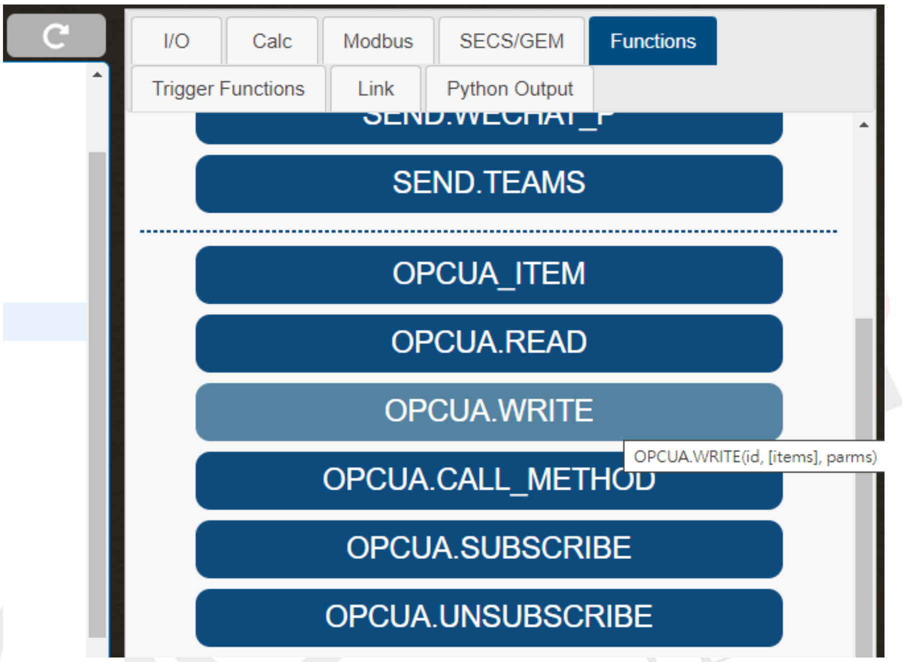


Figure 4-270 Click on "OPCUA.WRITE"

```
items = []
item = OPCUA_ITEM("DO_00")
item.data = False
items.append(item)
OPCUA.WRITE("OPCUA00_127_0_0_1_48010" , items)
```

Figure 4-271 Add the OPCUA.WRITE function in script editing area

4.2.2.3.13 OPCUA.CALL_METHOD

This function is utilized for invoking methods on the OPCUA Server. Prior to its use, navigate to the linking page and configure the connection to the OPCUA Server (refer to section 4.4.7 for configuration details). After completing the configuration, within the variable and function input section, click on the OPCUA.CALL_METHOD function (as depicted in Figure 4-272). This action will integrate the OPCUA.CALL_METHOD function seamlessly into the script-writing area. Subsequently, input the required parameters as exemplified in Figure 4-273.

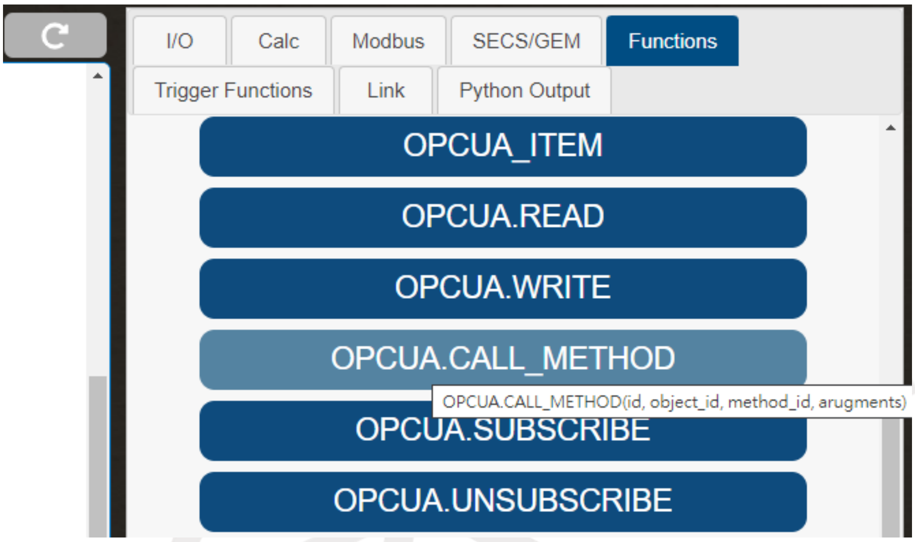


Figure 4-272 Click on "OPCUA.CALL_METHOD"

```
    # OPCUA.CALL_METHOD
arg = {"Command":"myCmd", "Parameters":"456"}

OPCUA.CALL_METHOD("OPCUA00_127_0_0_1_48010", \
                  "ns=2;s=Machine", \
                  "ns=2;s=Machine.RemoteCmd", arg)
```

Figure 4-273 Add the OPCUA.CALL_METHOD function in script editing area

4.2.2.3.14 OPCUA.SUBSCRIBE

This function is used to subscribe to nodes on the OPCUA Server. Prior to its utilization, navigate to the linking page and configure the connection to the OPCUA Server (refer to section 4.4.7 for configuration details). After completing the configuration, within the variable and function input section, click on the OPCUA.SUBSCRIBE function (as depicted in Figure 4-274). This action will seamlessly integrate the OPCUA.SUBSCRIBE function into the script-writing area. Following this, input the necessary parameters as shown in Figure 4-275.

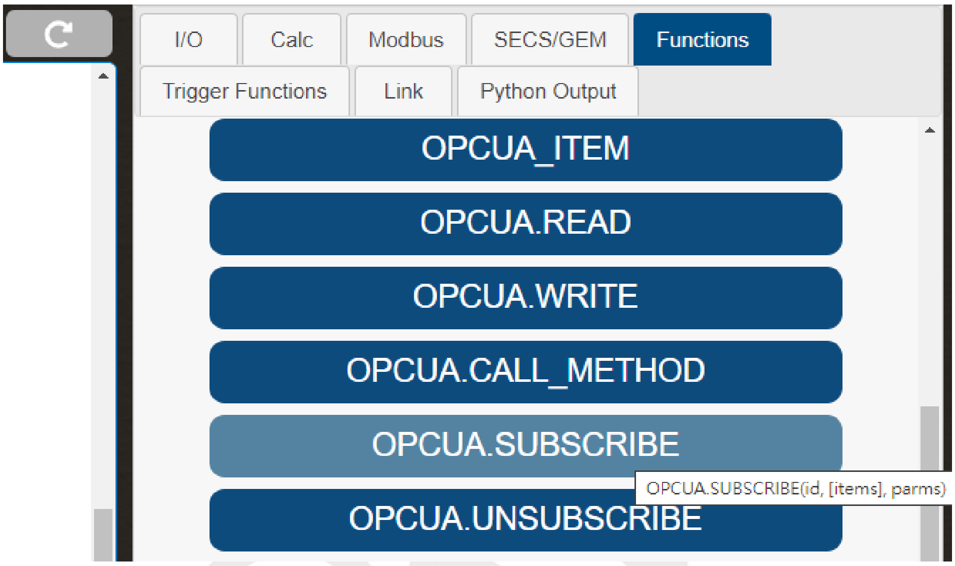


Figure 4-274 Click on "OPCUA.SUBSCRIBE"

```
item = OPCUA_ITEM("DI_00")
OPCUA.SUBSCRIBE("OPCUA00_127_0_0_1_48010" , item)
```

Figure 4-275 Add the OPCUA.SUBSCRIBE function in script editing area

4.2.2.3.15 OPCUA.UNSUBSCRIBE

This function is utilized to unsubscribe from the OPCUA Server. If nodes on the OPCUA Server were subscribed to in advance and the need arises to cancel those subscriptions, it's imperative to invoke this function. Prior to its use, navigate to the linking page and configure the connection to the OPCUA Server (refer to section 4.4.7 for configuration details). Once the configuration is complete, within the variable and function input section, click on the OPCUA.UNSUBSCRIBE function (as depicted in Figure 4-276). This action will seamlessly integrate the OPCUA.UNSUBSCRIBE function into the script-writing area. Subsequently, input the required parameters as demonstrated in Figure 4-277.

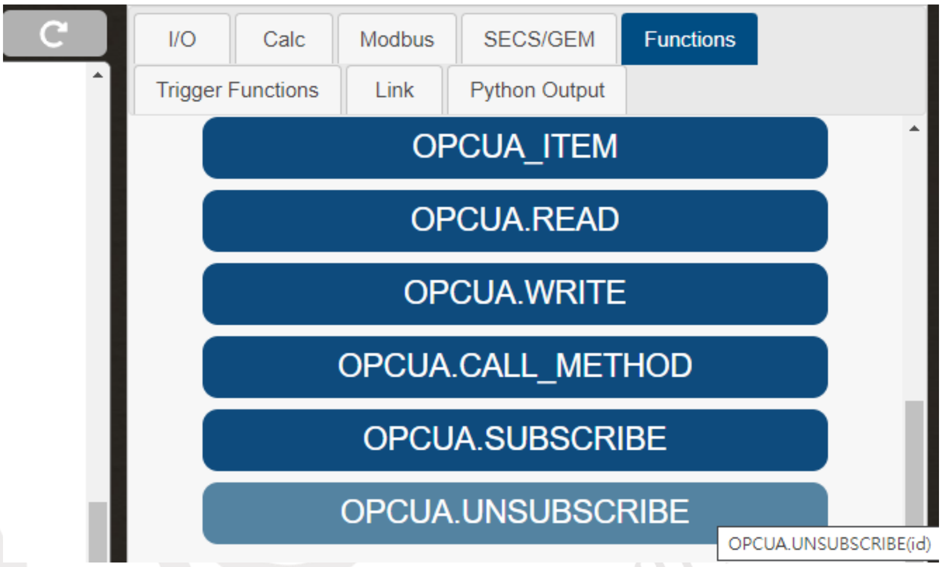


Figure 4-276 Click on "OPCUA.UNSUBSCRIBE"

```
OPCUA.UNSUBSCRIBE("OPCUA00_127_0_0_1_48010")
```

Figure 4-277 Add the OPCUA.UNSUBSCRIBE function in script editing area

4.2.2.3.16 PYTHON_MAIN_LOOP

The purpose of this function is to insert default Python script content. When the PYTHON_MAIN_LOOP is clicked in the function input area for variables and functions (as shown in Figure 4-278), the default script content will be inserted into the script writing area (as shown in Figure 4-279).

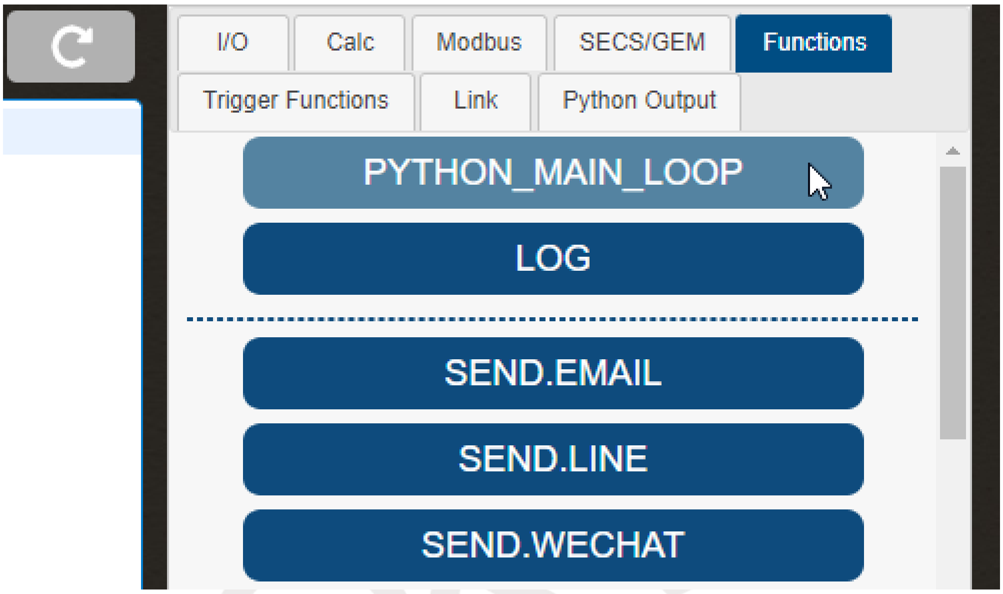


Figure 4-278 Click on "PYTHON_MAIN_LOOP"

```
from vic import *


def mainLoop(id, info, data):
    pass

```

Figure 4-279 Inserting Default Content in Script Editing Area

*Special Note：PYTHON_MAIN_LOOP will also input logs from I/O and Modbus Channel. If data is written to I/O or Modbus Channel within PYTHON_MAIN_LOOP, the data displayed on the UI may not be as expected. To avoid this situation, enable the "At Fixed Interval" option in Data Log Setting to ensure consistent time intervals (please refer to section 4.1.4.1 for the settng method).

4.2.2.3.17 WRITE_TEXT

This function writes content to a text file. By clicking WRITE_TEXT in the variable and function input area (as shown in Figure 4-280), the WRITE_TEXT function will be added to the script editor (as shown in Figure 4-281).

WRITE_TEXT (filepath, content, sec) function needs to be set with three parameters, each representing the following：

filepath：Sets the file path location for the text file. If only the file name is set, the file will be created in the /opt/nDAS/Export folder on the nDAS device.

content：Sets the content to be written.

sec：The time interval in seconds during which this funciton will not be executed again. If set to 0, there will be no time interval.

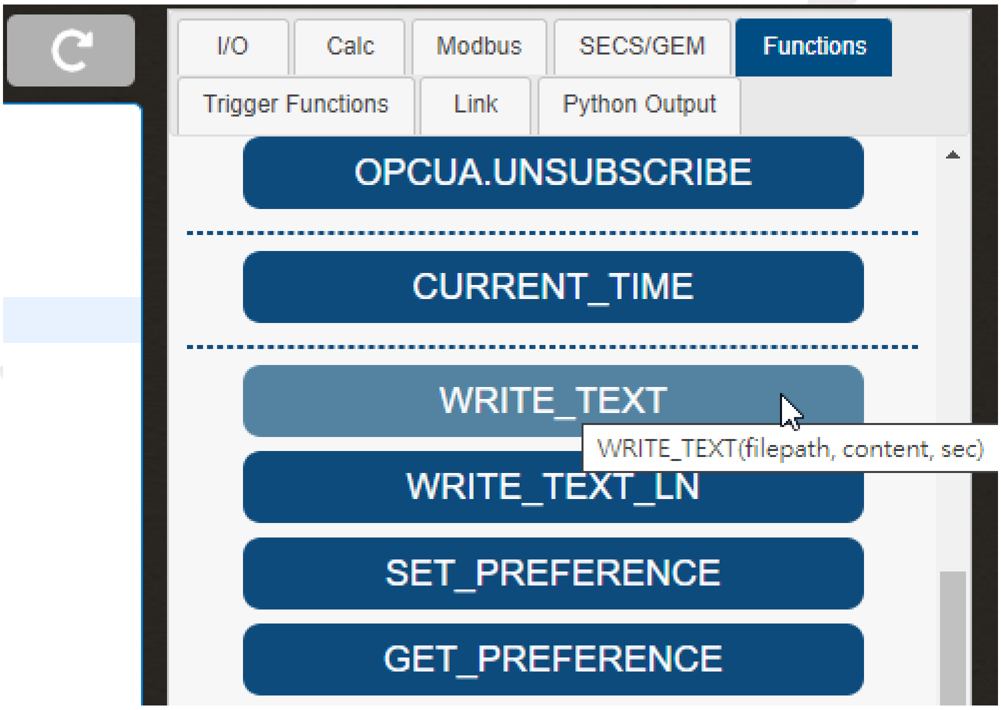


Figure 4-280 Click on “WRITE_TEXT”

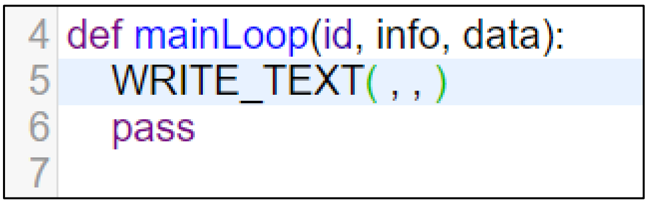


Figure 4-281 Add the WRITE_TEXT function in the script editing area

4.2.2.3.18 WRITE_TEXT_LN

This function writes content to a text file and automatically adds a newline after writing the content. When you click on WRITE_TEXT_LN in the variable and function input area (as shown in Figure 4-282), a WRITE_TEXT_LN function will be added to the script editor with the content you want to write (as shown in Figure 4-283).

WRITE_TEXT_LN (filepath, content, sec) function needs to be set with three parameters, each representing the following：

filepath：Sets the file path location for the text file. If only the file name is set, the file will be created in the /opt/nDAS/Export folder on the nDAS device.

content：Sets the content to be written.

sec：The time interval in seconds during which this funciton will not be executed again. If set to 0, there will be no time interval.

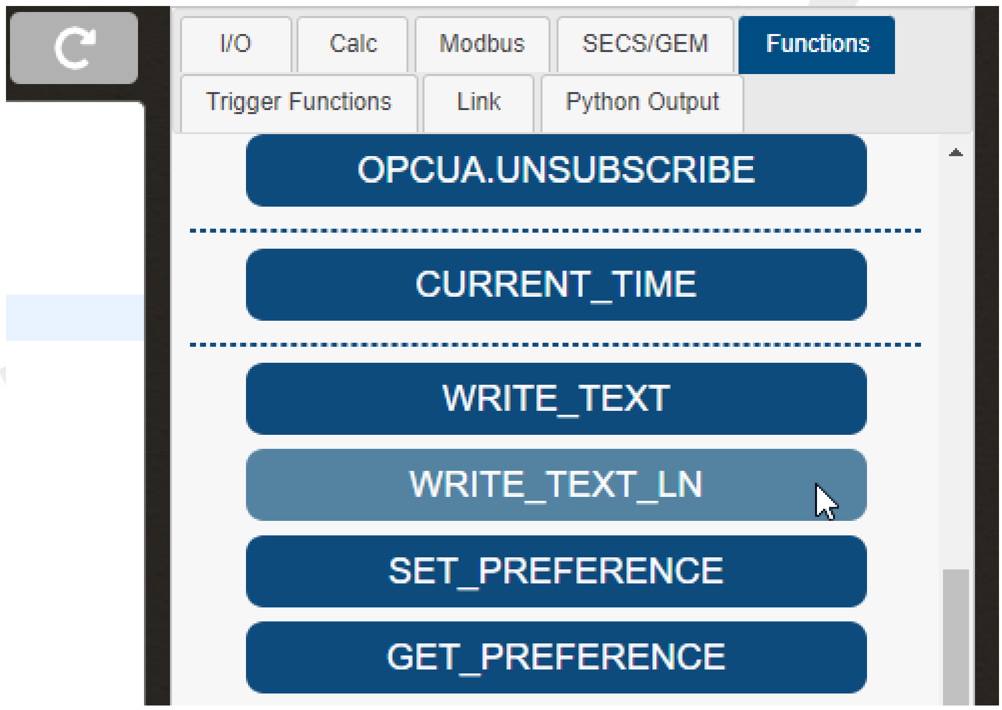


Figure 4-282 Click on “WRITE_TEXT_LN”

```
def mainLoop(id, info, data):
    WRITE_TEXT_LN( , , )
    pass
```

Figure 4-283 Add the WRITE_TEXT_LN funciton in the script editing area

4.2.2.3.19 SET_PREFERENCE

This function enables the setting of system variables, which will persist even after nDAS is restarted. To access the SET_PREFERENCE function in the variable and function input area (as shown in Figure 4-284), simply click on it. This will add the SET_PREFERENCE function to the script editor (as shown in Figure 4-285).

SET_PREFERENCE (key, value) function requires two parameters to be set, each with the following meanings：

key：Sets the parameter name.

value：Sets the value of the parameter.

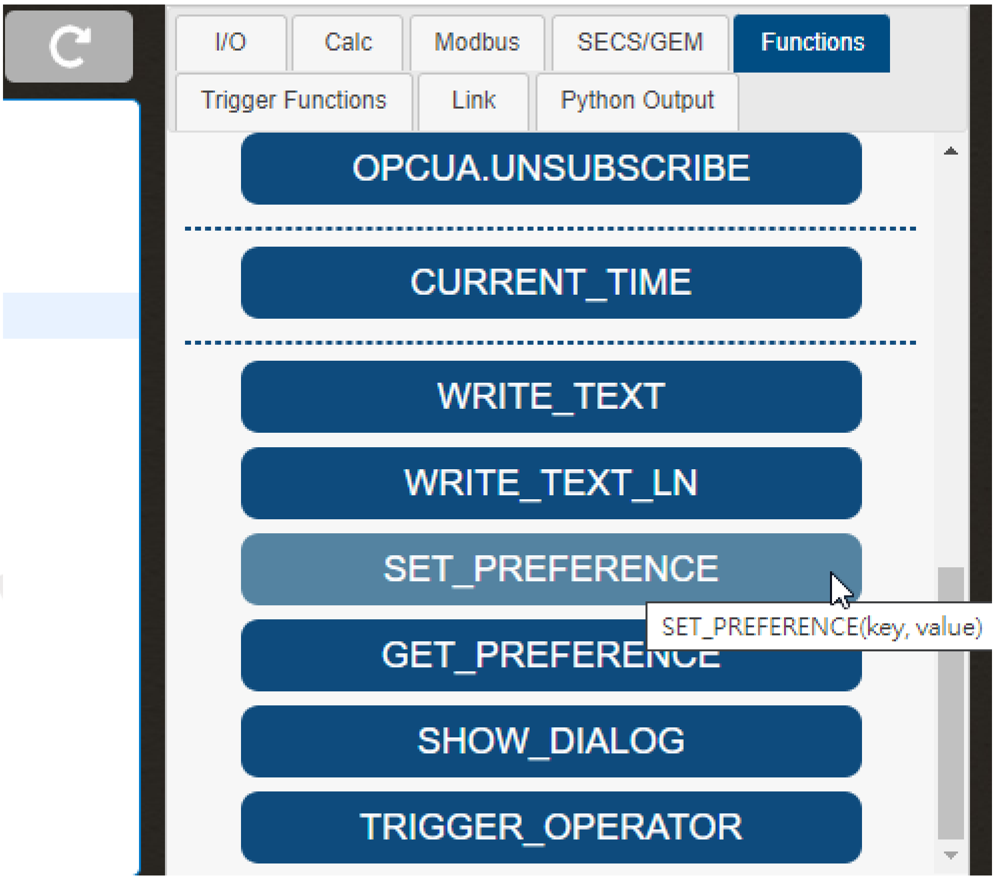


Figure 4-284 Click on “SET_PREFERENCE”

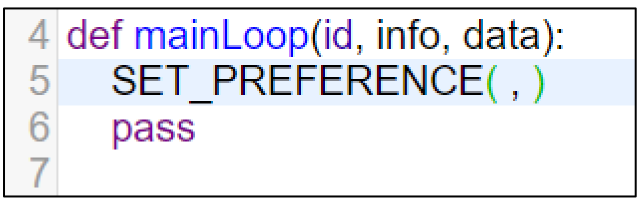


Figure 4-285 Add the SET_PREFERENCE function in the script editing area

4.2.2.3.20 GET_PREFERENCE

This function retrieves the value of a system variable, and returns a default value if the variable does not exist. To add the GET_PREFERENCE function to the script editor, click on GET_PREFERENCE in the variable and function input area (as shown in Figure 4-286). The GET_PREFERENCE function will be added to the script editor (as shown in Figure 4-287).

GET_PREFERENCE (key, default_value) function requires two parameters to be set, each with the following meanings：

key：Sets the parameter name.

default_value：The default value for the parameter.

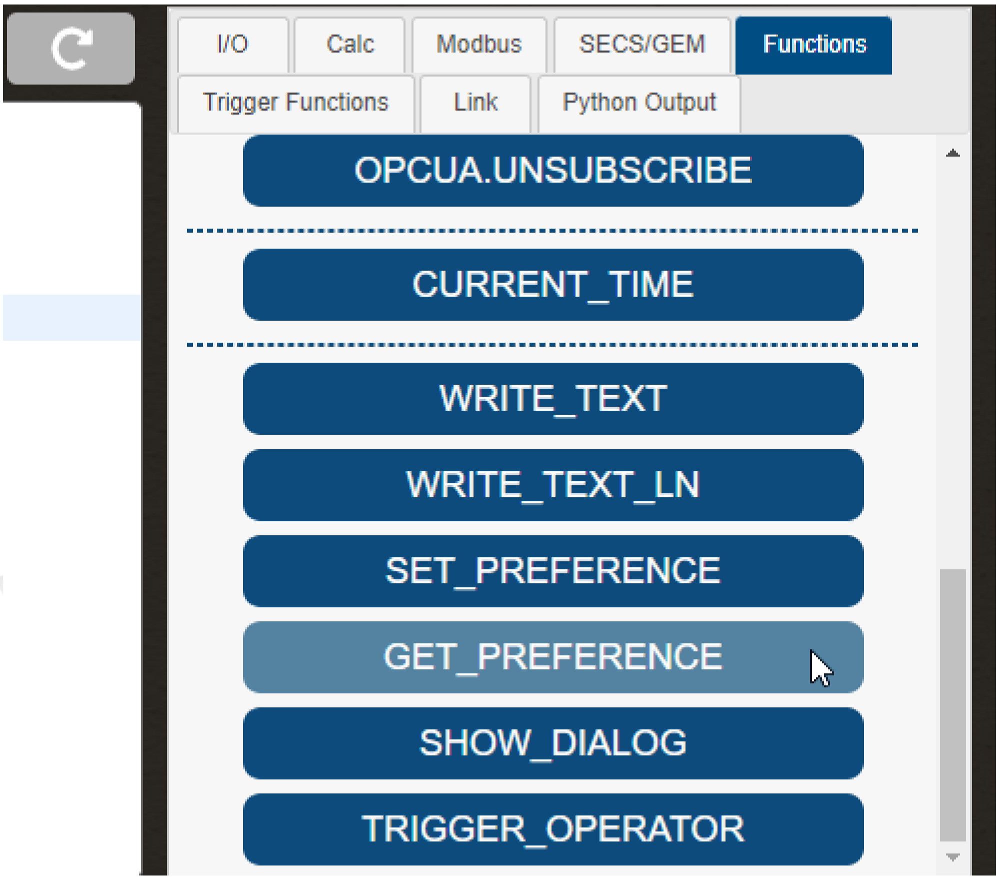


Figure 4-286 Click on “GET_PREFERENCE”

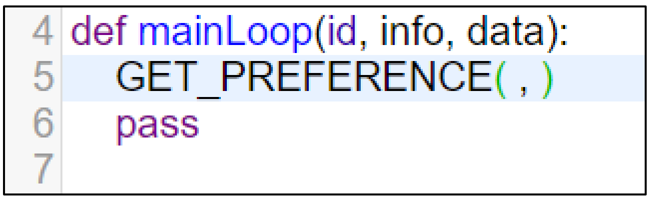


Figure 4-287 Add the GET_PREFERENCE function in the script editing area

4.2.2.3.21 SHOW_DIALOG

This function can pop up a dialog in the nDAS operation interface and display the set message. By clicking on SHOW_DIALOG in the variable and function input area (as shown in Figure 4-288), the SHOW_DIALOG funciton will be added to the script editor (as shown in Figure 4-289).

SHOW_DIALOG(type, title, content, sec) function requires four parameters to be set, each representing the following：

type：Sets the type of the dialog, please refer to Table 4-34.

title：Sets the title of the dialog.

content：Sets the content of the dialog.

sec：The time interval in seconds during which this funciton will not be executed again. If set to 0, there will be no time interval.

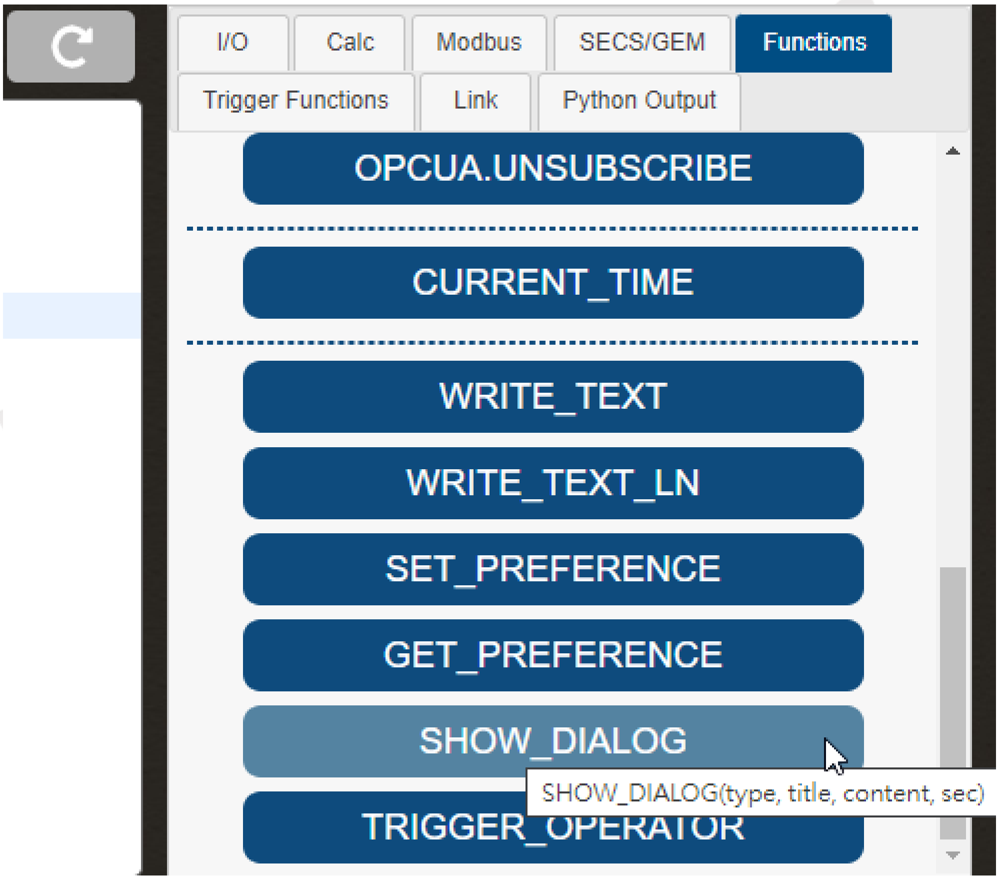


Figure 4-288 Click on “SHOW_DIALOG”

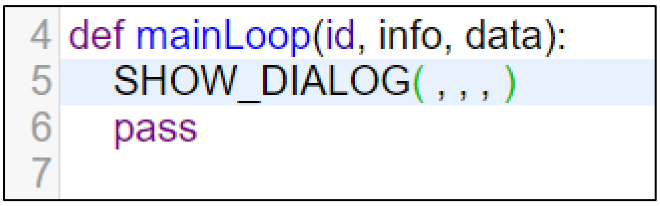
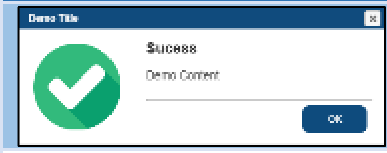
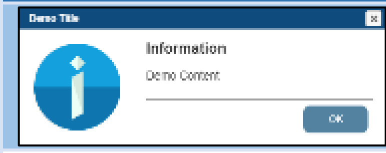


Figure 4-289 Add the SHOW_DIALOG function in the script editing area

Table 4-34 Dialog Type Table

Type	Dialog	Type	Dialog
0 (Success)		2 (Information)	
1 (Failed)		3 (Warning)	

4.2.2.3.22 TRIGGER_OPERATOR

This function triggers a specified VIC flow trigger operator. Clicking on TRIGGER_OPERATOR in the function input area (as shown in Figure 4-290), the TRIGGER_OPERATOR function will be added to script editor (as shown in Figure 4-291).

TRIGGER_OPERATOR(id) function requires one parameter to be set, with the following meaning：

id：sets the ID of the VIC flow trigger operator to be triggered.

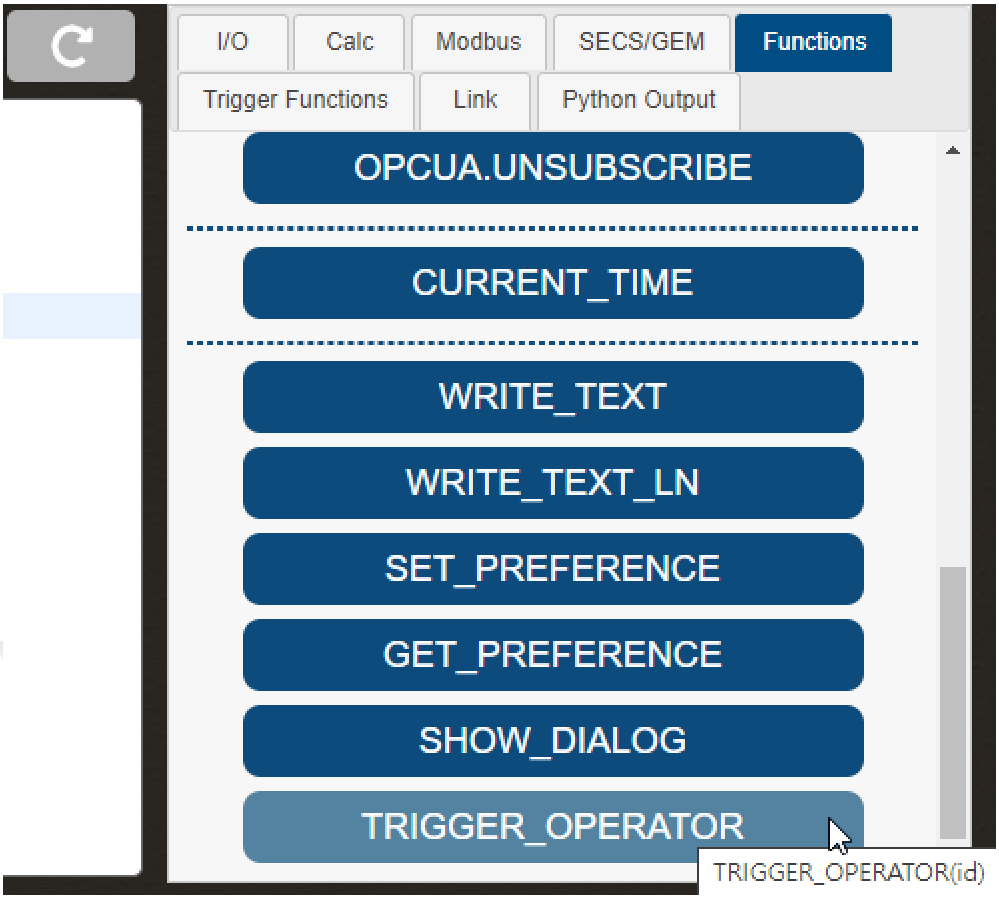


Figure 4-290 Click on “TRIGGER_OPERATOR”

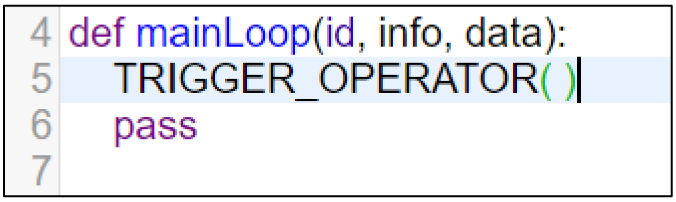


Figure 4-291 Add the TRIGGER_OPERATOR function in the script editing area

4.2.2.4 SECS/GEM

4.2.2.4.1 Equipment

4.2.2.4.1.1 SET_ALARM

This function is utilized to set the current state of an ALID as an alarm state. In the Variable and Function Input Area of the SECS/GEM in the equipment, clicking on SET_ALARM (as shown in Figure 4-292) results in the addition of the SET_ALARM function in the script writing area (as shown in Figure 4-293).

The SET_ALARM(alid, TRUE/FALSE) function necessitates the configuration of two parameters, each representing the following:

alid: Specifies the ALID for which the state should be altered.

TRUE/FALSE: Determines whether to activate the alarm. If an alarm is to be activated, set this parameter to TRUE; conversely, set it to FALSE.

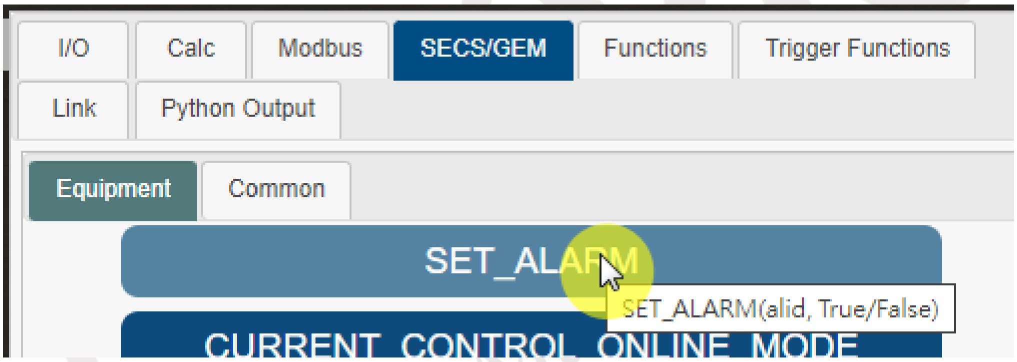


Figure 4-292 Click on "SET_ALARM"

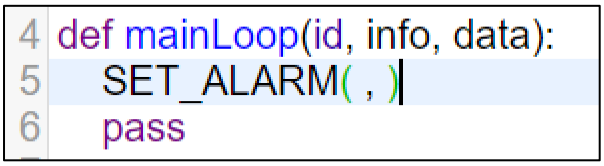


Figure 4-293 Add the SET_ALARM function in the script editing area

4.2.2.4.1.2 CURRENT_CONTROL_ONLINE_MODE

This function serves to acquire the current ON-LINE control mode of the equipment side. In the Variable and Function Input Area of the SECS/GEM within the equipment, clicking on CURRENT_CONTROL_ONLINE_MODE (as depicted in Figure 4-294) leads to the inclusion of the CURRENT_CONTROL_ONLINE_MODE function in the script writing area (as illustrated in Figure 4-295).

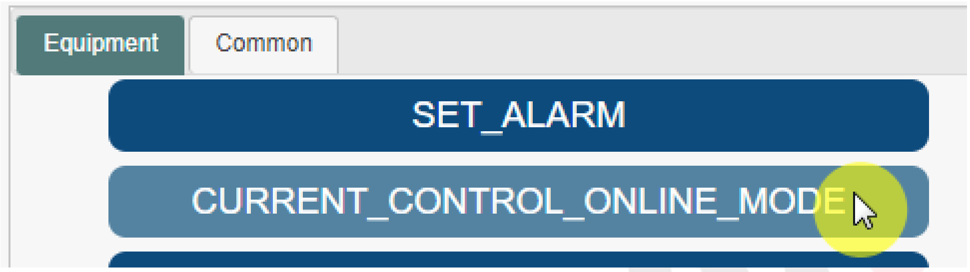


Figure 4-294 Click on "CURRENT_CONTROL_ONLINE_MODE"

```
def mainLoop(id, info, data):
    CURRENT_CONTROL_ONLINE_MODE()
    pass
```

Figure 4-295 Add the CURRENT_CONTROL_ONLINE_MODE function in the script editing area

4.2.2.4.1.3 SET_CONTROL_OFFLINE

This function facilitates the adjustment of the current control state of the equipment side to OFF-LINE. Within the Variable and Function Input Area of the SECS/GEM in the equipment, clicking on SET_CONTROL_OFFLINE (as indicated in Figure 4-296) results in the incorporation of the SET_CONTROL_OFFLINE function in the script writing area (as shown in Figure 4-297).

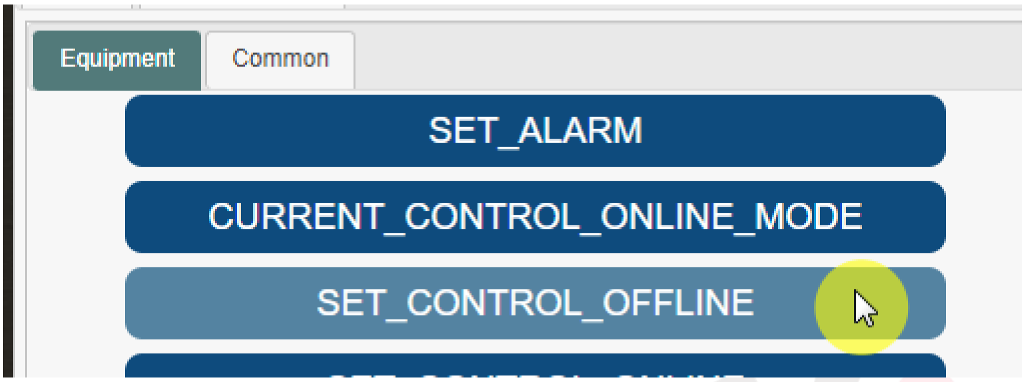


Figure 4-296 Click on "SET_CONTROL_OFFLINE"

```
def mainLoop(id, info, data):
    SET_CONTROL_OFFLINE()
    pass
```

Figure 4-297 Add the SET_CONTROL_OFFLINE function in the script editing area

4.2.2.4.1.4 SET_CONTROL_ONLINE

This function enables the adjustment of the current control state of the equipment side to ON-LINE. Within the Variable and Function Input Area of the SECS/GEM in the equipment, clicking on SET_CONTROL_ONLINE (as shown in Figure 4-298) results in the inclusion of the SET_CONTROL_ONLINE function in the script writing area (as illustrated in Figure 4-299).

Figure 4-298 Click on "SET_CONTROL_ONLINE"

```
def mainLoop(id, info, data):
    SET_CONTROL_ONLINE()
    pass
```

Figure 4-299 Add the SET_CONTROL_ONLINE function in the script editing area

4.2.2.4.1.5 SET_CONTROL_ONLINE_LOCAL

This function allows the configuration of the current ON-LINE control state of the equipment side to LOCAL. Within the Variable and Function Input Area of the SECS/GEM in the equipment, clicking on SET_CONTROL_ONLINE_LOCAL (as depicted in Figure 4-300) results in the inclusion of the SET_CONTROL_ONLINE_LOCAL function in the script writing area (as illustrated in Figure 4-301).

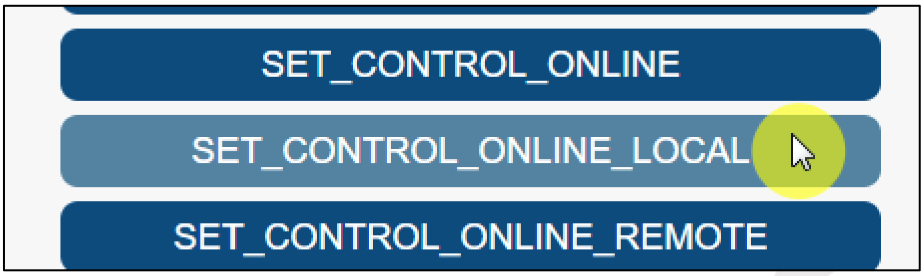


Figure 4-300 Click on "SET_CONTROL_ONLINE_LOCAL"

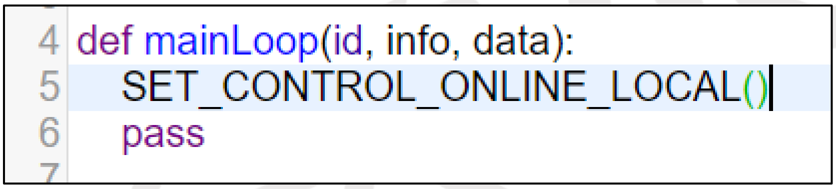


Figure 4-301 Add the SET_CONTROL_ONLINE_LOCAL function in the script editing area

4.2.2.4.1.6 SET_CONTROL_ONLINE_REMOTE

This function enables the adjustment of the current ON-LINE control state of the equipment side to REMOTE. Within the Variable and Function Input Area of the SECS/GEM in the equipment, clicking on SET_CONTROL_ONLINE_REMOTE (as shown in Figure 4-302) results in the addition of the SET_CONTROL_ONLINE_REMOTE function in the script writing area (as illustrated in Figure 4-303).

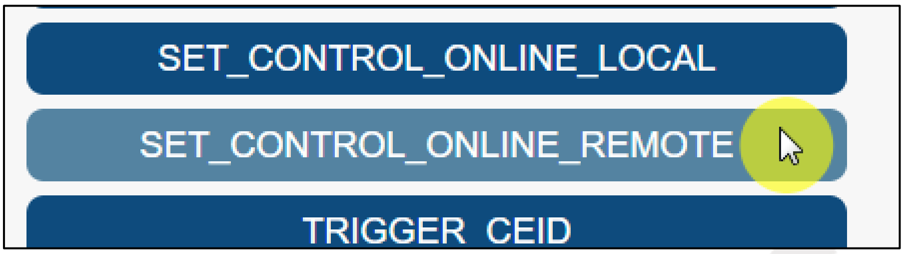


Figure 4-302 Click on "SET_CONTROL_ONLINE_REMOTE"

```
def mainLoop(id, info, data):
    SET_CONTROL_ONLINE_REMOTE()
    pass
```

Figure 4-303 Add the SET_CONTROL_ONLINE_REMOTE function in the script editing area

4.2.2.4.1.7 TRIGGER_CEID

This function allows the configuration of sending an Event Report (S6F11). Within the Variable and Function Input Area of the SECS/GEM in the equipment, clicking on TRIGGER_CEID (as depicted in Figure 4-304) results in the inclusion of the TRIGGER_CEID function in the script writing area (as illustrated in Figure 4-305).

The TRIGGER_CEID(ceid) function requires the configuration of a single parameter, which signifies the following:

ceid: Specifies the Event Report (CEID) to be sent.

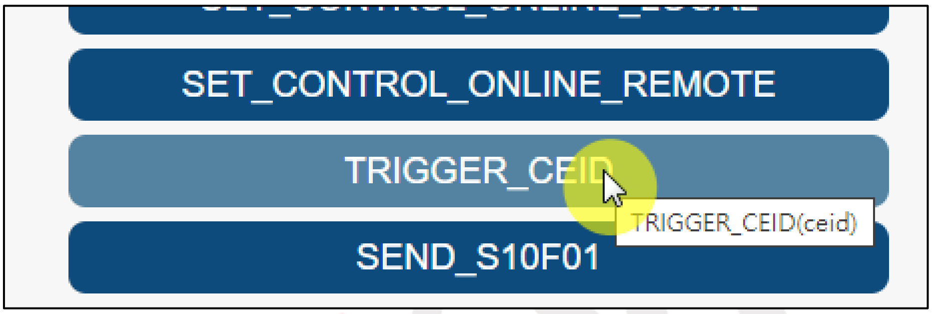


Figure 4-304 Click on "TRIGGER_CEID"

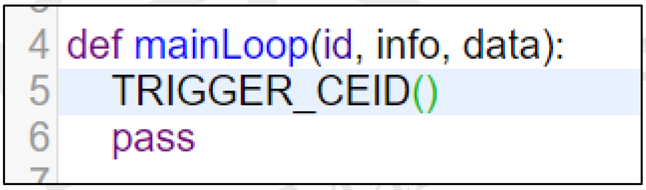


Figure 4-305 Add the TRIGGER_CEID function in the script editing area

4.2.2.4.1.8 SEND_S10F01

This function facilitates the issuance of the S10F1 command to the Host side. In the Variable and Function Input Area of the SECS/GEM in the equipment, clicking on SEND_S10F01 (as shown in Figure 4-306) leads to the inclusion of the SEND_S10F01 function in the script writing area (as illustrated in Figure 4-307).

The SEND_S10F01(tid, text) function necessitates the configuration of two parameters, each representing the following:

tid: Terminal ID.

text: Message content to be transmitted to the Host side.

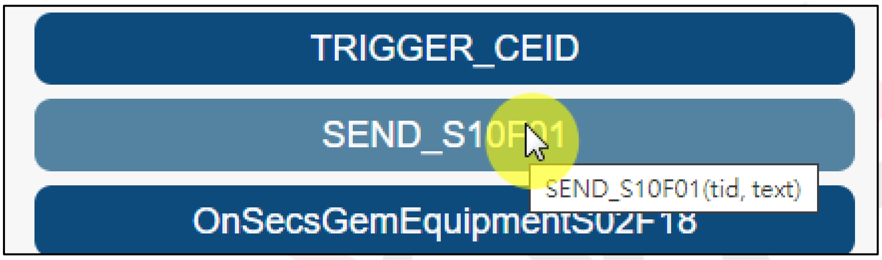


Figure 4-306 Click on "SEND_S10F01"

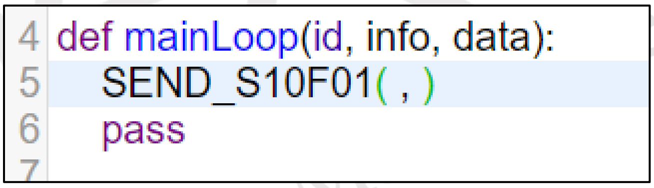


Figure 4-307 Add the SEND_S10F01 function in the script editing area

4.2.2.4.1.9 OnSecsGemEquipS02F18

When the equipment side receives the S2F18 command sent by the Host side, this function is triggered. Within the Variable and Function Input Area, clicking on OnSecsGemEquipS02F18 in the trigger function (as illustrated in Figure 4-308) results in the addition of the OnSecsGemEquipS02F18 function in the script writing area (as shown in Figure 4-309). Users can then customize the content of the function according to their requirements.

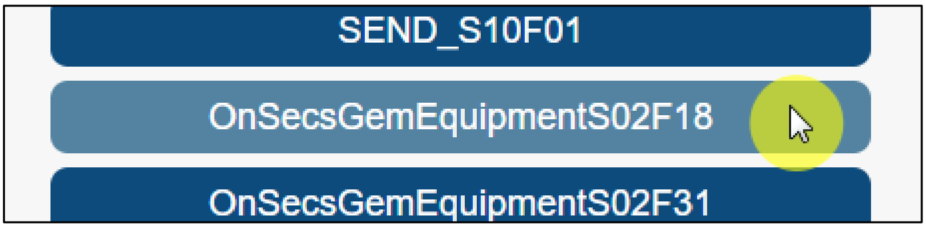


Figure 4-308 Click on "OnSecsGemEquipS02F18"

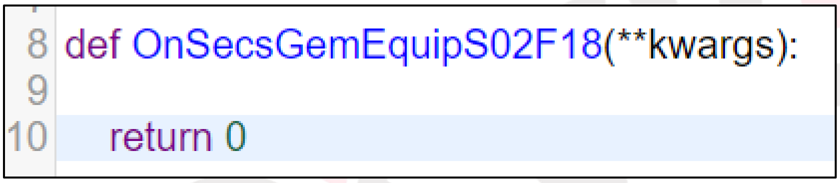


Figure 4-309 Add the OnSecsGemEquipS02F18 function in the script editing area

4.2.2.4.1.10 OnSecsGemEquipS02F31

When the equipment side receives the S2F31 command sent by the Host side, this function is triggered. Within the Variable and Function Input Area, clicking on OnSecsGemEquipS02F31 in the trigger function (as shown in Figure 4-310) results in the addition of the OnSecsGemEquipS02F31 function in the script writing area (as illustrated in Figure 4-311). Users can then customize the content of the function according to their requirements.

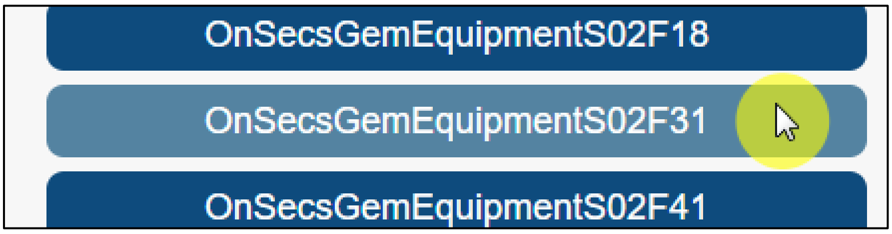


Figure 4-310 Click on "OnSecsGemEquipS02F31"

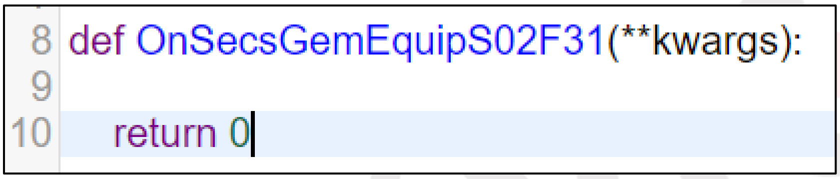


Figure 4-311 Add the OnSecsGemEquipS02F31 function in the script editing area

4.2.2.4.1.11 OnSecsGemEquipS02F41

When the equipment side receives the S2F41 command sent by the Host side, this function is triggered. Within the Variable and Function Input Area, clicking on OnSecsGemEquipS02F41 in the trigger function (as illustrated in Figure 4-312) results in the addition of the OnSecsGemEquipS02F41 function in the script writing area (as shown in Figure 4-313). Users can then customize the content of the function according to their requirements.

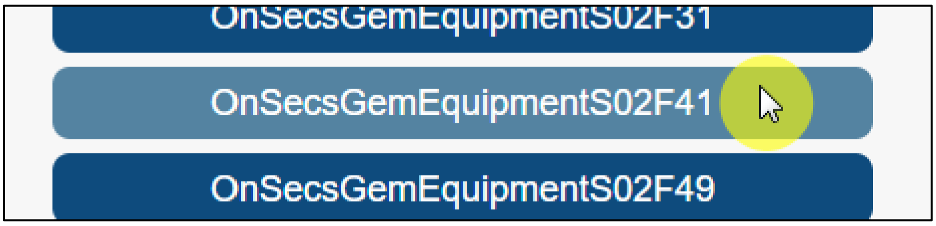


Figure 4-312 Click on "OnSecsGemEquipS02F41"

```
def OnSecsGemEquipS02F41(**kwargs):
    pass
```

Figure 4-313 Add the OnSecsGemEquipS02F41 function in the script editing area

4.2.2.4.1.12 OnSecsGemEquipS02F49

When the equipment side receives the S2F49 command sent by the Host side, this function is triggered. Within the Variable and Function Input Area, clicking on OnSecsGemEquipS02F49 in the trigger function (as shown in Figure 4-314) results in the addition of the OnSecsGemEquipS02F49 function in the script writing area (as depicted in Figure 4-315). Users can then customize the content of the function according to their requirements.

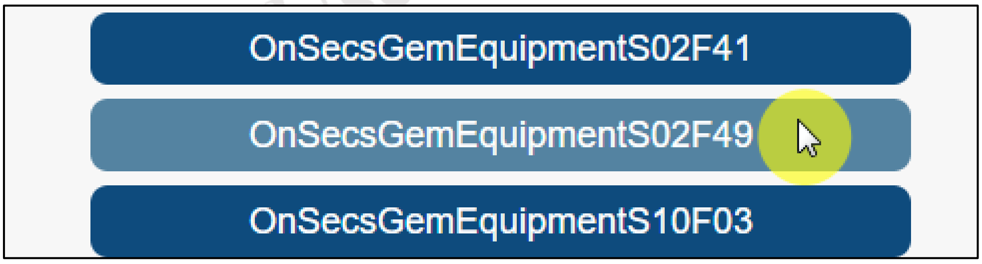


Figure 4-314 Click on "OnSecsGemEquipS02F49"

```
def OnSecsGemEquipS02F49(**kwargs):
    pass
```

Figure 4-315 Add the OnSecsGemEquipS02F49 function in the script editing area

4.2.2.4.1.13 OnSecsGemEquipS10F03

When the equipment side receives the S10F3 command sent by the Host side, this function is triggered. Within the Variable and Function Input Area, clicking on OnSecsGemEquipS10F03 in the trigger function (as shown in Figure 4-316) results in the addition of the OnSecsGemEquipS10F03 function in the script writing area (as depicted in Figure 4-317). Users can then customize the content of the function according to their requirements.

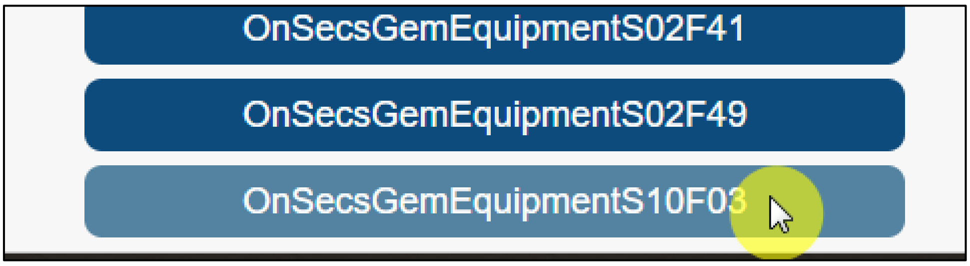


Figure 4-316 Click on "OnSecsGemEquipS10F03"

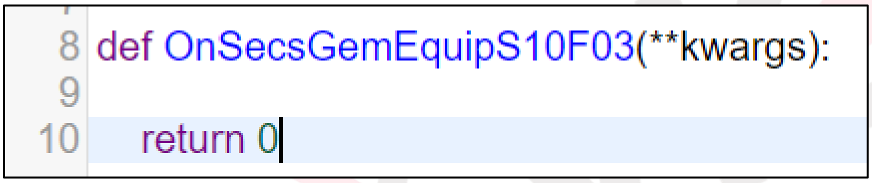


Figure 4-317 Add the OnSecsGemEquipS10F03 function in the script editing area

4.2.2.4.2 Common

4.2.2.4.2.1 ARE_YOU_THERE

The ARE_YOU_THERE function is used to send the Are-You-There (S1F1) message from the equipment side. In the SECS/GEM section of the Variable and Function Input Area, clicking on ARE_YOU_THERE (as shown in Figure 4-318) will add the ARE_YOU_THERE function to the script writing area (as depicted in Figure 4-319).

The ARE_YOU_THERE function requires three parameters to be set, each representing the following:

*role: Set the role to be sent. The default value is set as being sent by the Host side. However, the system only supports sending from the equipment side, so this parameter needs to be set to the string "E".

*linkId: This parameter is used to set the Host link name to be used. Again, since the system only supports sending from the equipment side, this parameter should be set as an empty string.

*callback: Represents the function to be used after receiving the response. If set, it indicates asynchronous processing. If not set, it indicates synchronous processing.

Note: * indicates optional parameters.

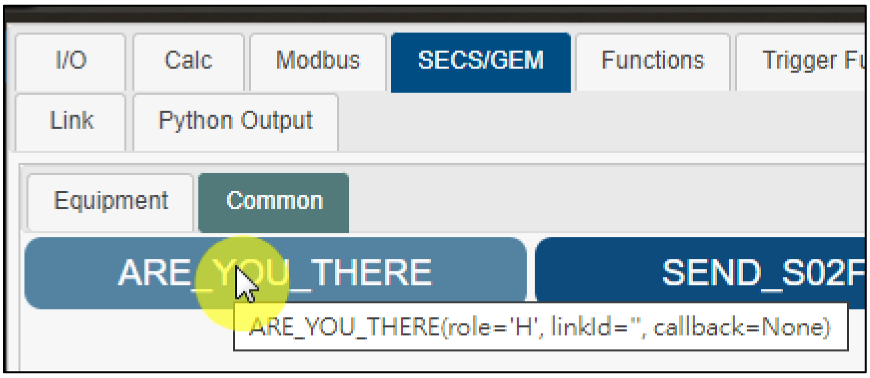


Figure 4-318 Click on "ARE_YOU_THERE"

```
def mainLoop(id, info, data):
    ARE_YOU_THERE()
    pass
```

Figure 4-319 Add the ARE_YOU_THERE function in the script editing area

4.2.2.4.2.2 SEND_S02F17

The SEND_S02F17 function is used to send the Request On-Line Time (S2F17) message from the equipment side. In the SECS/GEM section of the Variable and Function Input Area, clicking on SEND_S02F17 (as shown in Figure 4-320) will add the SEND_S02F17 function to the script writing area (as depicted in Figure 4-321).

The SEND_S02F17 function requires three parameters to be set, each representing the following:

*role: Set the role to be sent. The default value is set as being sent from the equipment side, so this parameter should be set to the string "E".

*linkId: This parameter is used to set the Host link name to be used. However, since the system only supports sending from the equipment side, this parameter should be set as an empty string.

*callback: Represents the function to be used after receiving the response. If set, it indicates asynchronous processing. If not set, it indicates synchronous processing.

Note: * indicates optional parameters.

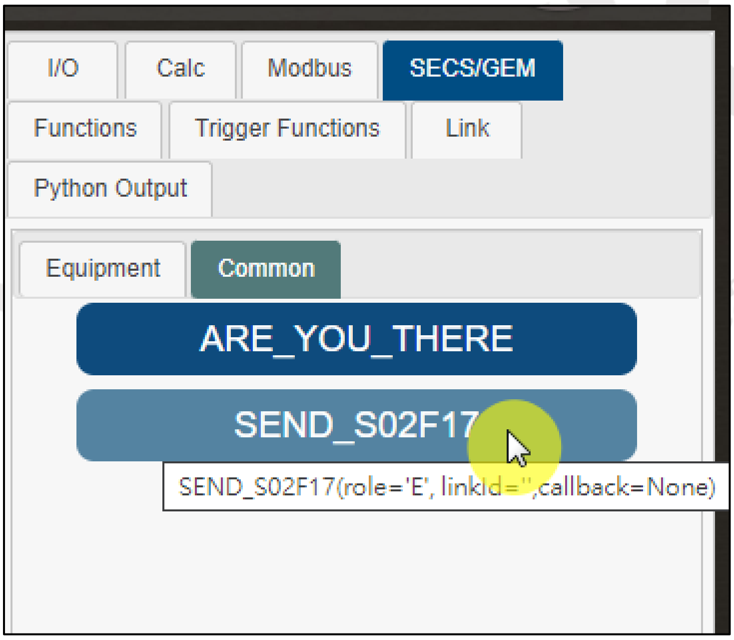


Figure 4-320 Click on "SEND_S02F17"

```
def mainLoop(id, info, data):
    SEND_S02F17()
    pass
```

Figure 4-321 Add the SEND_S02F17 function in the script editing area

4.2.2.5 Trigger Functions

4.2.2.5.1 OnInitScript()

Upon initialization of the script and project loading, this function is triggered. By clicking on OnInitScript in the trigger function input area for variables and functions (as shown in Figure 4-322), the OnInitScript function will be added to the script editing area (as shown in Figure 4-323), allowing users to customize the contents of the function according to their needs.

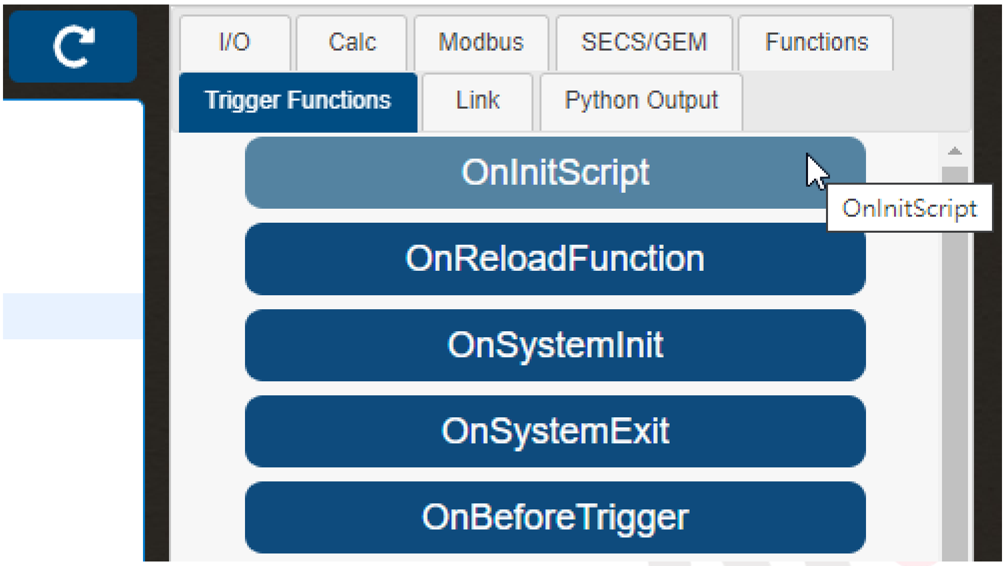


Figure 4-322 Click on “OnInitScript”

```
from vic import *

def OnInitScript():
    pass


def mainLoop(id, info, data):

    pass

```

Figure 4-323 Add the OnInitScript function in the script editing area

4.2.2.5.2 OnReloadFunction()

Clicking the Reload button or pressing the shortcut key (F9) triggers this function. By clicking OnReloadFunction in the trigger function area of variables and functions input (as shown in Figure 4-324), the OnReloadFunction function will be added to the script writing area (as shown in Figure 4-325)). Users can write the contents of the function according to their own needs.

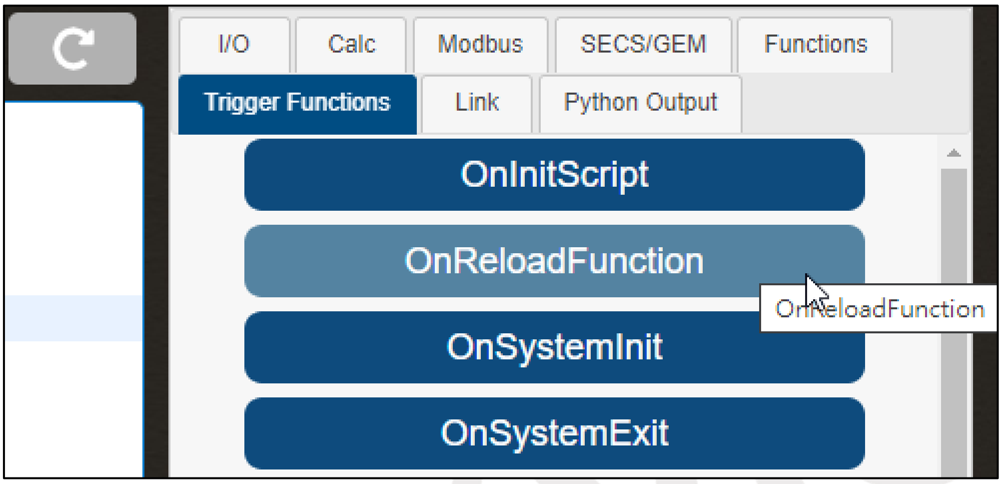


Figure 4-324 Click on "OnReloadFunction"

```
from vic import *

def OnReloadFunction():
    pass


def mainLoop(id, info, data):

    pass

```

Figure 4-325 Add the OnReloadFunction function in the script editing area

4.2.2.5.3 OnSystemInit()

This function is triggered automatically when the program is opened and initialization is completed (after the project is loaded). To add the OnSystemInit function in the script editing area, click on OnSystemInit in the trigger function of variables and function inputs (as shown in Figure 4-326). The user can then write the contents of the function according to their needs (as shown in Figure 4-327).

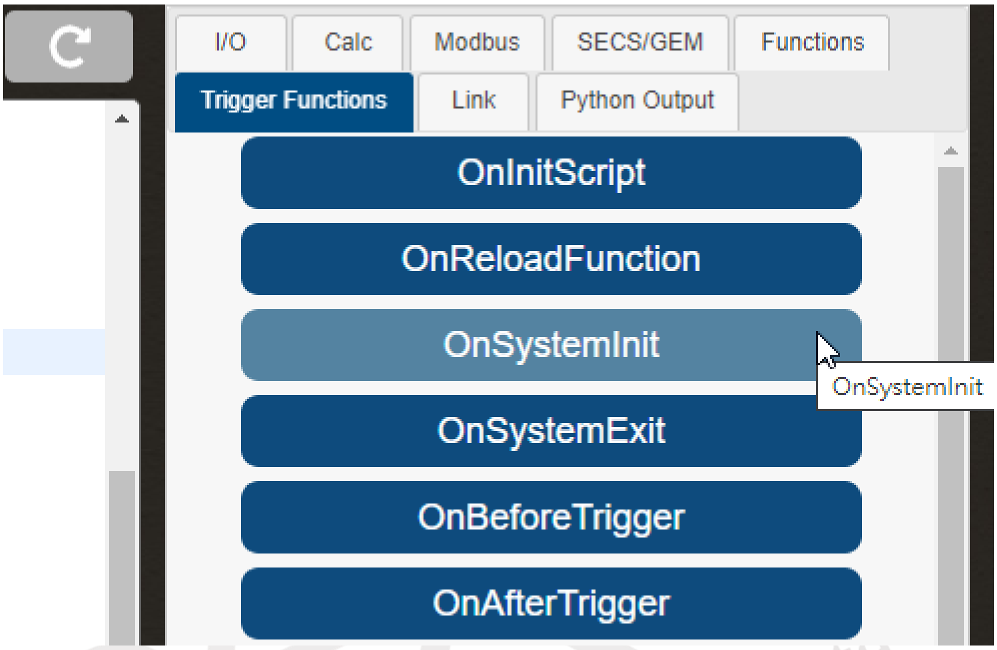


Figure 4-326 Click on “OnSystemInit”

```python
from vic import *

def OnSystemInit():
    pass


def mainLoop(id, info, data):
    pass

```

Figure 4-327 Add the OnSystemInit function in the script editing area

4.2.2.5.4 OnSystemExit()

This function is triggered when the program is closed. To add the OnSystemExit function in the script writing area, click OnSystemExit in the trigger function section of the variables and function input area (as shown in Figure 4-328). The user can then customize the content of the function according to their needs (as shown in Figure 4-329).

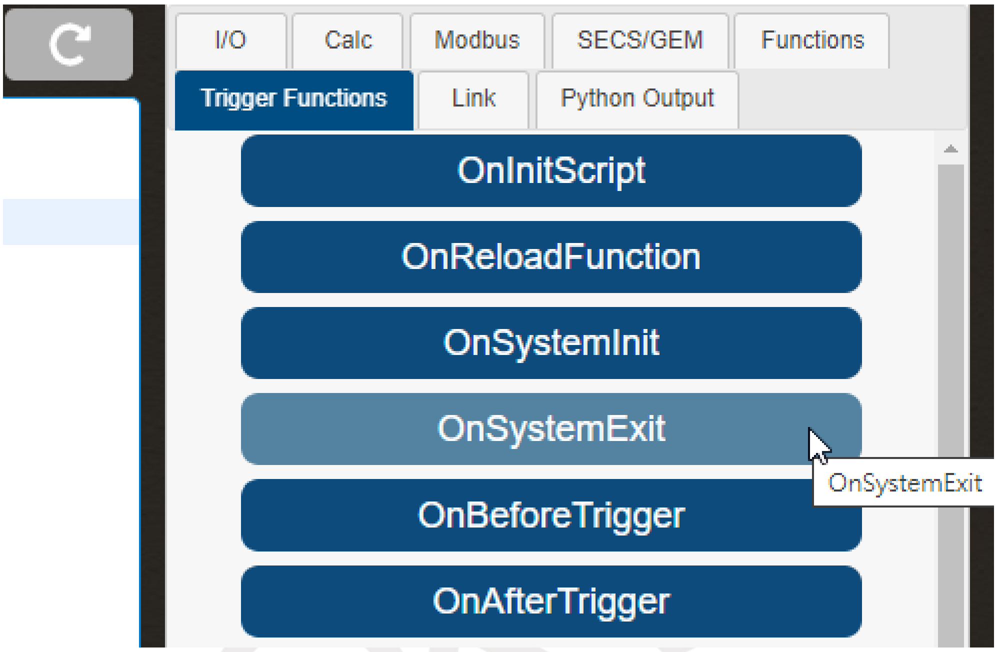


Figure 4-328 Click on "OnSystemExit"

```
from vic import *

def OnSystemExit():
    pass


def mainLoop(id, info, data):
    pass

```

Figure 4-329 Add the OnSystemExit function in the script editing area

4.2.2.5.5 OnBeforeTrigger

When the Trigger Operator is triggered, this function will be triggered. By clicking OnBeforeTrigger in the trigger function section of variables and function inputs (as shown in Figure 4-330), the OnBeforeTrigger function will be added to the script writing area (as shown in Figure 4-331). Users can write the content of the function according to their needs.

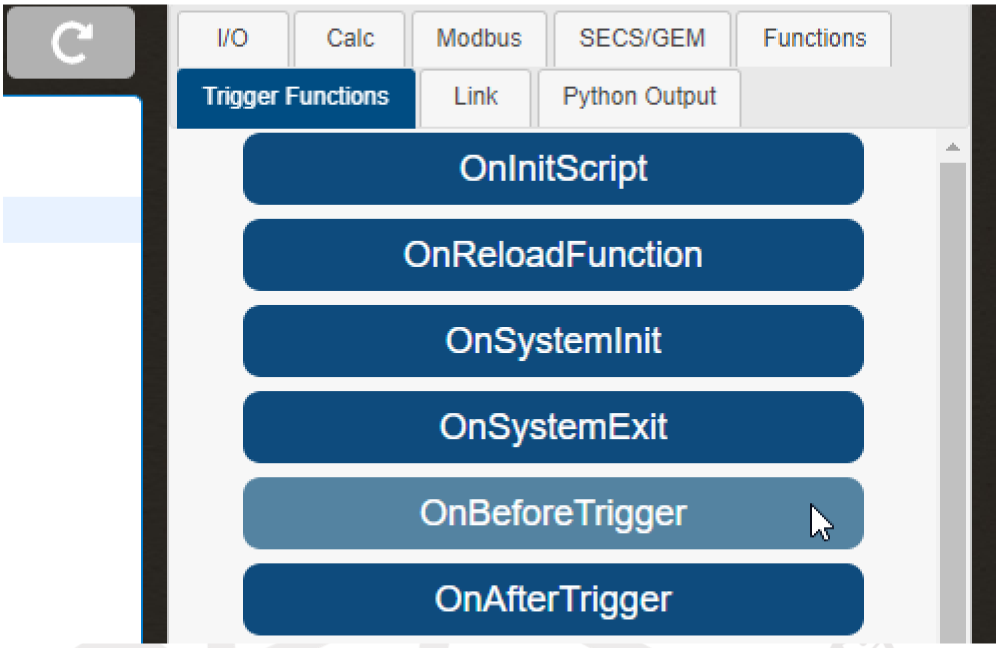


Figure 4-330 Click on "OnBeforeTrigger"

```
from vic import *

def OnBeforeTrigger(id):
    pass


def mainLoop(id, info, data):
    pass

```

Figure 4-331 Add the OnBeforeTrigger function in the script editing area

4.2.2.5.6 OnAfterTrigger

Once the Trigger Operator is activated, this function will also be triggered. Clicking OnAfterTrigger in the variable and function input area (as shown in Figure 4-332) will add the OnAfterTrigger function to the script writing area (as shown in Figure 4-333). Users can write the content of the function according to their needs.

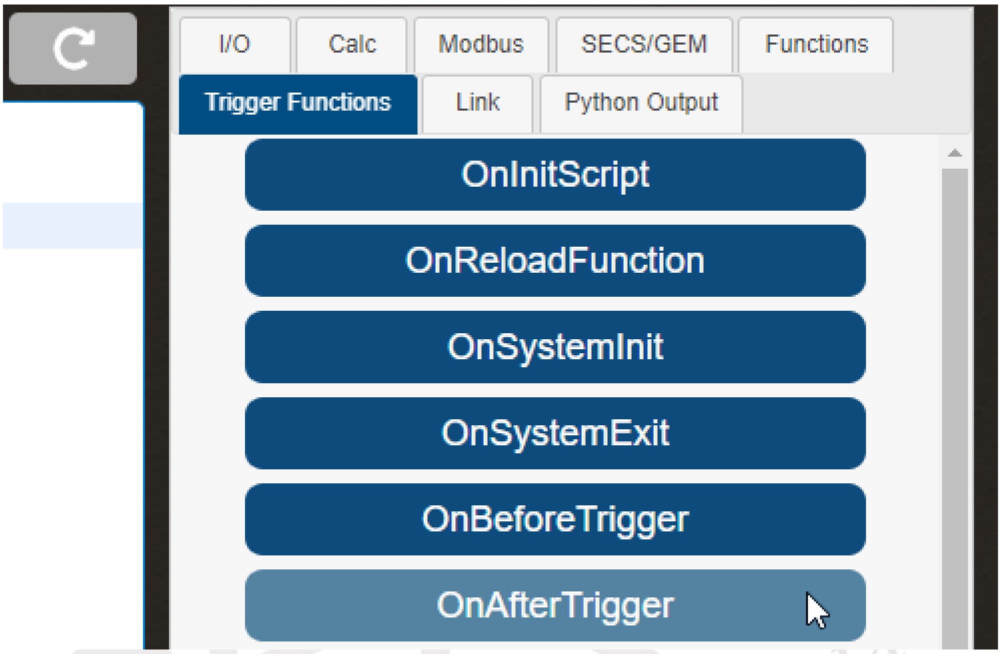


Figure 4-332 Click on “OnAfterTrigger”

```
from vic import *

def OnAfterTrigger(id):
    pass


def mainLoop(id, info, data):
    pass

```

Figure 4-333 Add the OnAfterTrigger function in the script editing area

4.2.2.5.7 OnOpcuaDataChange

When the subscribed OPCUA node value changes, it triggers this function. In the Variables and Functions input section, click on the "OnOpcuaDataChange" function (as shown in Figure 4-334), and this will add the "OnOpcuaDataChange" function to the script writing area (as illustrated in Figure 4-335). Users can customize the content of the function according to their needs. The "id" and "data" parameters represent the subscribed OPCUA id and node status, respectively.

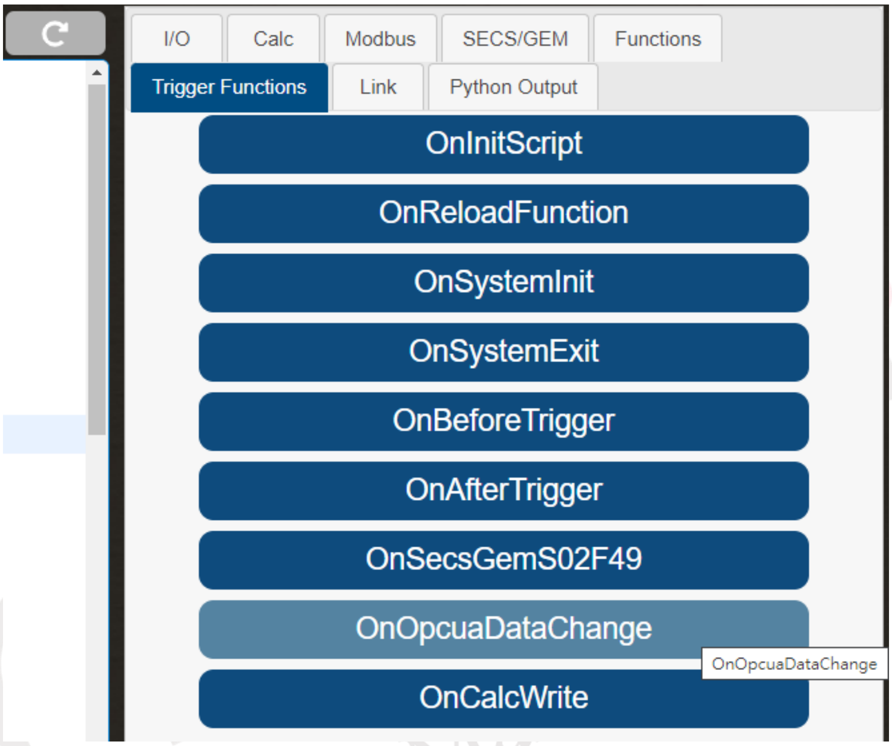


Figure 4-334 Click on "OnOpcuaDataChange"

```

def OnOpcuaDataChange(id, data):
    pass

```

Figure 4-335 Add the OnOpcuaDataChange function in the script editing area

4.2.2.5.8 OnCalcWrite

When the RESTful API writes values to the CALC operator, it triggers this function. In the Variables and Functions input section, click on the "OnCalcWrite" function (as shown in Figure 4-336), and this will add the "OnCalcWrite" function to the script writing area (as depicted in Figure 4-337). Users can customize the content of the function according to their needs. The "id" and "value" parameters represent the incoming CALC operator index and value, respectively.

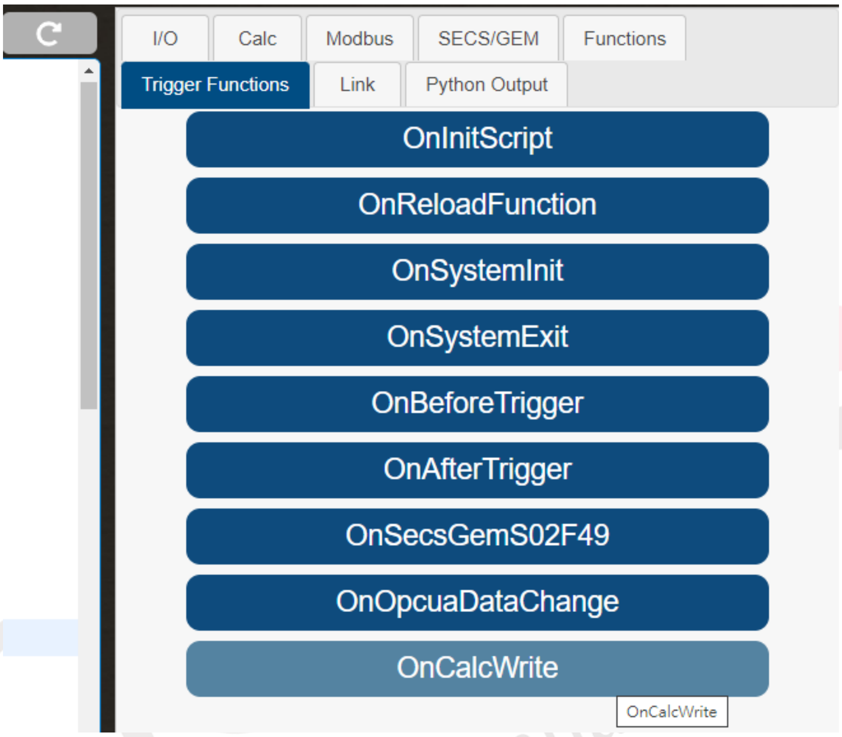


Figure 4-336 Click on "OnCalcWrite"

```
def OnCalcWrite(id, value):
    pass
```

Figure 4-337 Add the OnCalcWrite function in the script editing area

4.2.2.6 Modbus

4.2.2.6.1 Bit Channel

It is feasible to engage in the scripting of Modbus Bit Channel value read and write operations, drawing inspiration from section 4.1.2.4.1. Within the domain of variable and function input, the act of selecting the Bit Channel within the Modbus context is pivotal (as delineated in Figure 4-338). This consequential selection begets the manifestation of the Modbus Bit Channel variable.

In the context of nDAS configuration, a total of 128 utilizable Modbus Channels are at one's disposal, although the interface extends visibility to a mere 20. In instances where the user's intentions encompass Modbus Bit Channels that transcend the boundaries of the interface's revelation, a self-directed approach to input becomes requisite. This entails the personal entry of communication port and channel parameters. The stipulated methodology for input is encapsulated within the syntax "BIT_CHANNEL[port][n]". It is of significance to note that in the event of COM Port utilization, the 'port' value aligns with the COM Port index. Conversely, in the case of TCP employment, the 'port' is allocated within the numerical range of 100 to 104, while 'n' spans the continuum of 0, 1, 2, all the way through to 127.

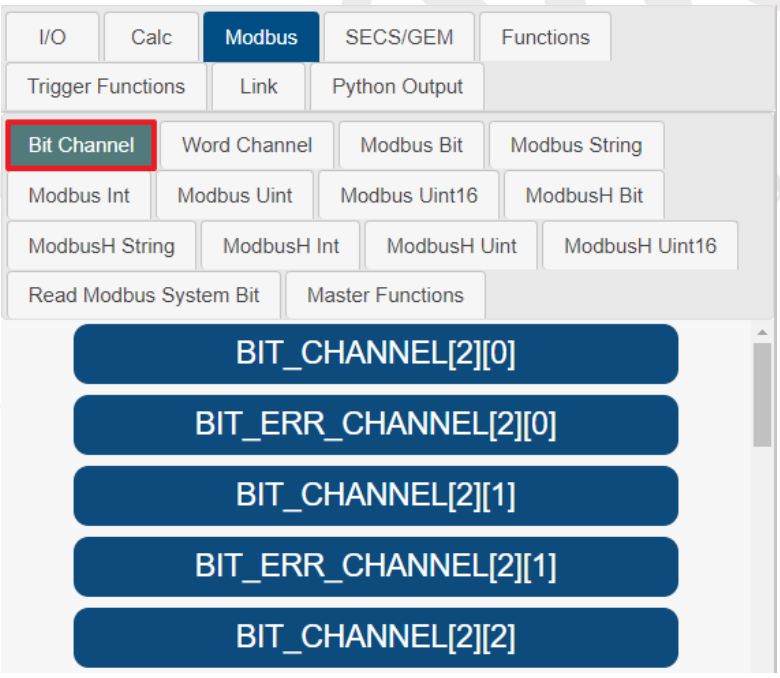


Figure 4-338 Selecting Bit Channel for Variables and Function Input in Modbus.

4.2.2.6.2 Word Channel

It is possible to read and write Modbus Word Channel values in a script, as described in section 4.1.2.4.1. To select a Modbus Word Channel variable in the variable and function input area, choose the Word Channel option under Modbus (as shown in Figure 4-339). The nDAS system has a total of 128 Modbus Channels available, but only 20 are displayed in the interface. If a user needs to use Modbus Word Channels beyond those displayed in the interface, they must input the communication port and channel manually using the following format: "WORD_CHANNEL[port][n]". For COM ports, the port number starts at 0 and goes up to the maximum port number minus 1. For TCP, the port numbers are 100-104, and n ranges from 0 to 127.

Figure 4-339 Selecting Word Channel for Variable and Function Input in Modbus

4.2.2.6.3 Modbus Bit

The Modbus Bit function facilitates the local Input Status (1x) bit read and write operations. To utilize this feature, navigate to the variable and function input section, and select the Modbus Bit within the Modbus category (as depicted in Figure 4-340). Upon selection, the Modbus Bit variable will be displayed. This software supports Modbus Bit addresses ranging from 1 to 65535; however, the interface only showcases addresses 1 to 25. If users intend to use Modbus Bit variables beyond those displayed, they should manually input the Modbus Bit variable using the syntax "MODBUSBIT[n]", where n represents values from 0 to 65535.

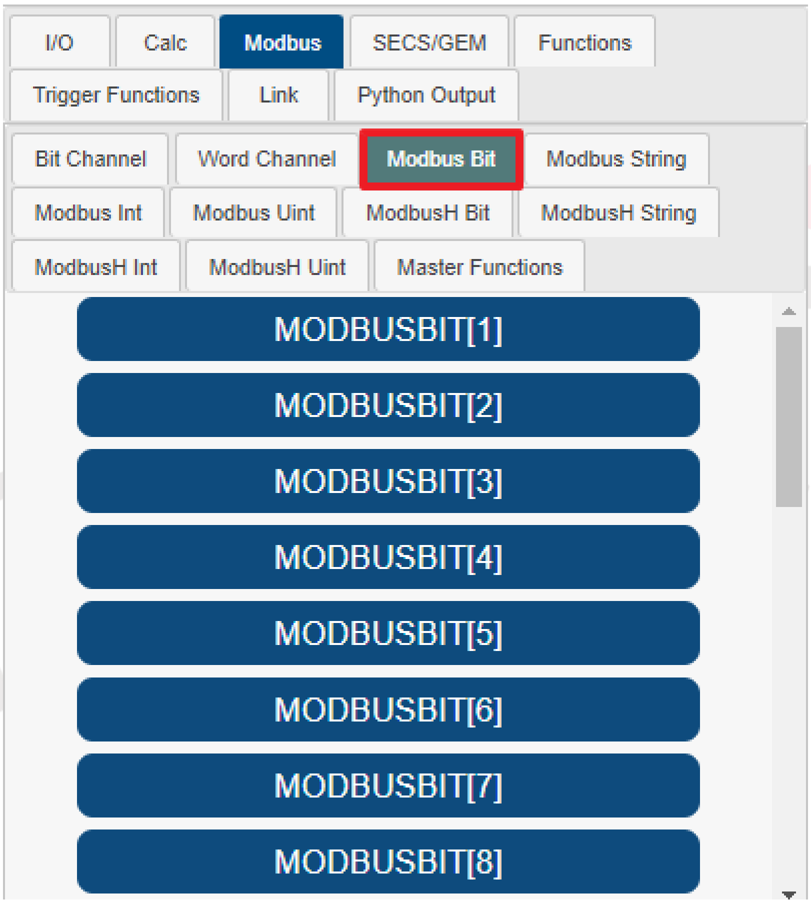


Figure 4-340 Selecting Modbus Bit for Variable and Function Input in Modbus

4.2.2.6.4 Modbus String

The Modbus String function facilitates local Input Register (3x) string read and write operations. To utilize this function, navigate to the variable and function input section, and choose the Modbus String option within the Modbus category (as shown in Figure 4-341). Upon selection, the Modbus String variable will be presented. This software supports Modbus String addresses ranging from 1 to 65535, while the interface displays addresses 1 to 25.

Should users require to work with Modbus String variables beyond those displayed, they should manually input the Modbus String variable using the syntax "MODBUSSTR[n]", where n corresponds to values ranging from 0 to 65535.

Please note: If the variable does not contain any data, the corresponding field will be displayed as "blank".

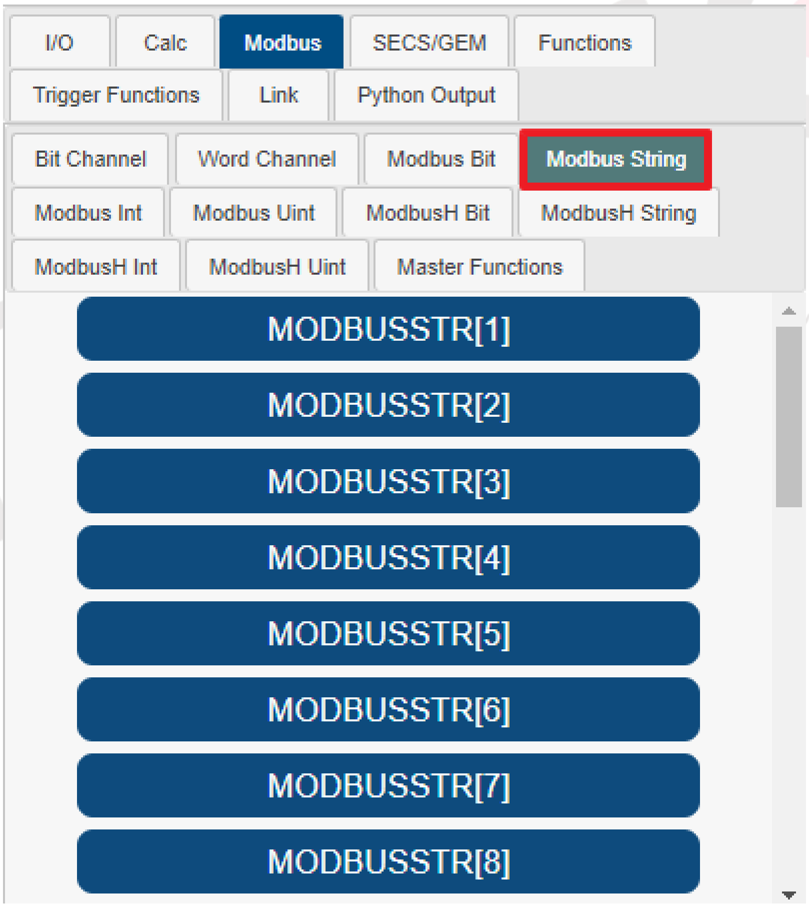


Figure 4-341 Selecting Modbus String for Variable and Function Input in Modbus

4.2.2.6.5 Modbus Int

The Modbus Int function facilitates local Input Register (3x) numeric read and write operations. To make use of this function, access the variable and function input section and select the Modbus Int option within the Modbus category (as shown in Figure 4-342). Upon selection, the Modbus Int variable will be displayed. This software allows for Modbus Int addresses ranging from 1 to 65535, with the interface presenting 25 addresses.

If users need to work with Modbus Int variables beyond the displayed range, they should manually input the Modbus Int variable using the syntax "MODBUSINT[n]", where n corresponds to values from 0 to 65535. A Modbus Int consists of 32 bits and represents a signed integer, with a range of -2,147,483,648 to 2,147,483,647.

Please be aware: If a situation arises where the provided data is in string format, the software will first attempt to convert the string to a numeric value. In cases where the conversion fails, the value will be represented as 0.

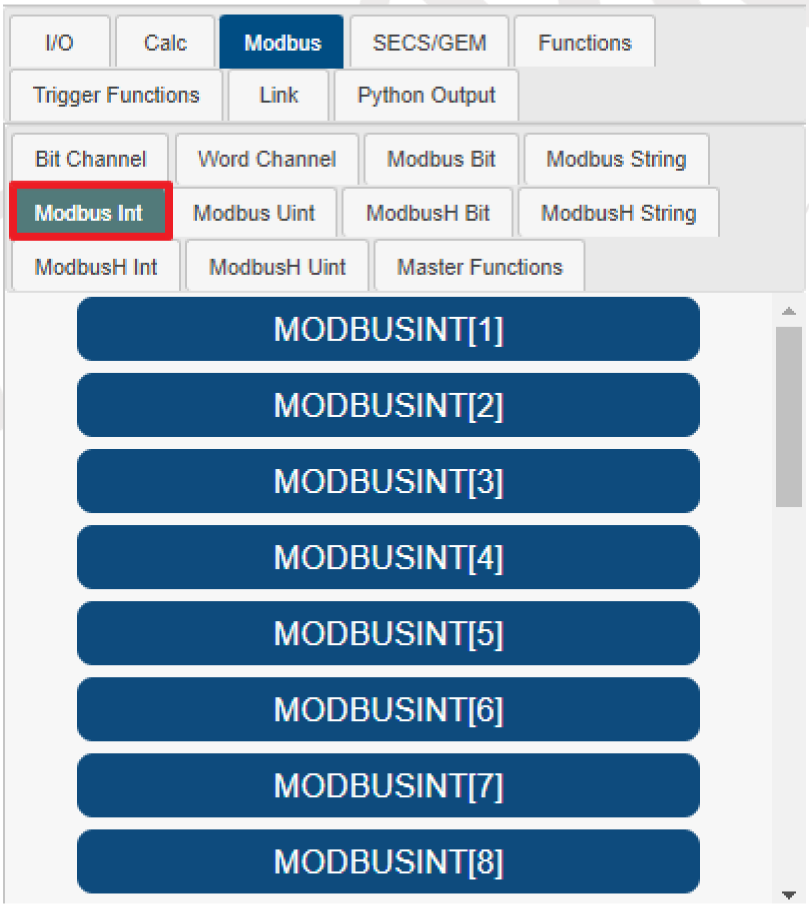


Figure 4-342 Selecting Modbus Int for Variable and Function Input in Modbus

4.2.2.6.6 Modbus Uint

The Modbus Uint function is designed for local Input Register (3x) numeric read and write operations, specifically for unsigned integers. To utilize this function, navigate to the variable and function input section and select the Modbus Uint option within the Modbus category (as shown in Figure 4-343). Upon selection, the Modbus Uint variable will be displayed. The software allows the use of Modbus Uint addresses ranging from 1 to 65535, with the interface presenting 25 addresses.

If users intend to work with Modbus Uint variables beyond the displayed range, they should manually input the Modbus Uint variable using the syntax "MODBUSUINT[n]", where n corresponds to values from 0 to 65535. A Modbus Uint consists of 32 bits and represents an unsigned integer, with a range of 0 to 4,294,967,295.

Please note: In scenarios where data is provided as a string, the software will first attempt to convert the string to a numeric value. If the conversion process fails, the value will be represented as 0.

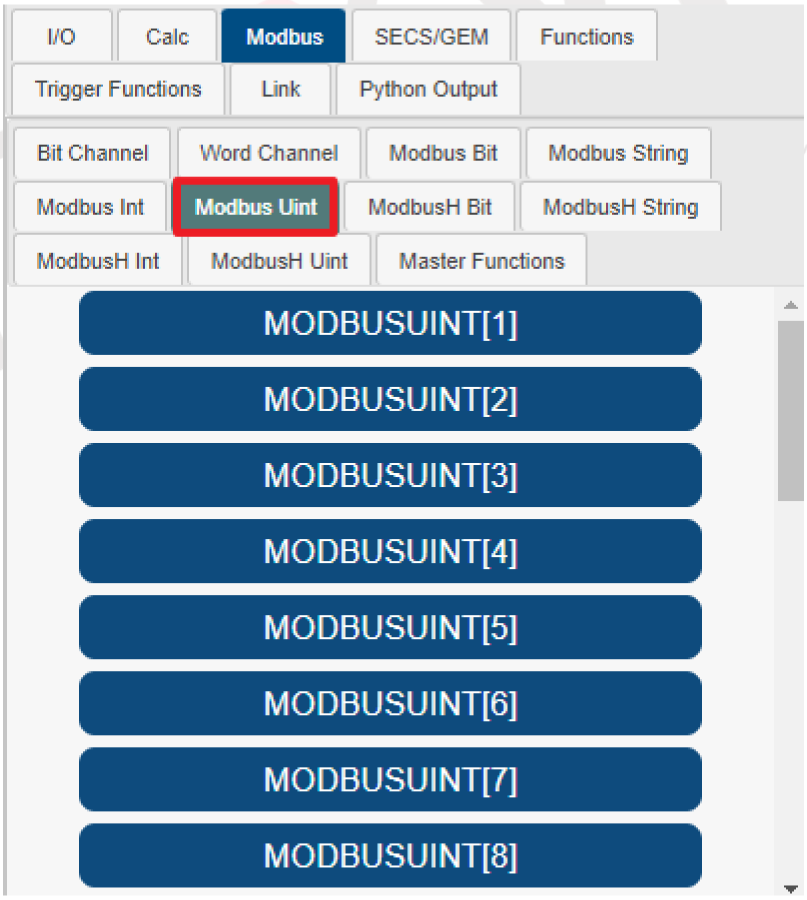


Figure 4-343 Selecting Modbus Uint for Variable and Function Input in Modbus

4.2.2.6.7 Modbus Uint16

The Modbus Uint16 function is designed for local Input Register (3x) numeric read and write operations, specifically for unsigned 16-bit integers. To utilize this function, navigate to the variable and function input section and select the Modbus Uint16 option within the Modbus category (as shown in Figure 4-344). Upon selection, the Modbus Uint16 variable will be displayed. The software allows the use of Modbus Uint16 addresses ranging from 1 to 65535, with the interface presenting 25 addresses.

If users intend to work with Modbus Uint16 variables beyond the displayed range, they should manually input the Modbus Uint16 variable using the syntax "MODBUSUINT16[n]", where n corresponds to values from 0 to 65535. A Modbus Uint16 consists of 16 bits and represents an unsigned integer, with a range of 0 to 65535.

Please note: In scenarios where data is provided as a string, the software will first attempt to convert the string to a numeric value. If the conversion process fails, the value will be represented as 0.

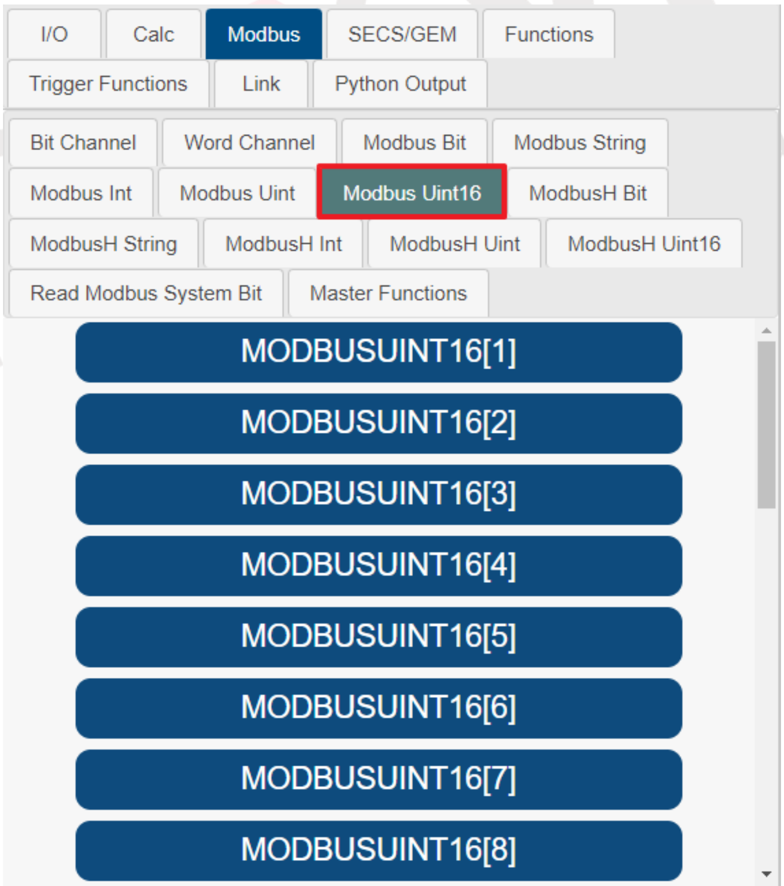


Figure 4-344 Selecting Modbus Uint16 for Variable and Function Input in Modbus

4.2.2.6.8 ModbusH Bit

ModbusH bits provide access to the local Coil Status (0x) bits for both reading and writing. For a comprehensive list of addresses for the local Coil Status, please refer to section 4.1.2.2. Within the realm of variables and function inputs, select the ModbusH bits of the Modbus protocol (as depicted in Figure 4-345). This action will promptly unveil the ModbusH bit variables. This software accommodates Modbus H bit addresses spanning from 1 to 65535, yet the interface exclusively showcases 25 at a time. Should a user aspire to employ ModbusH bit variables exceeding the limited interface depiction, diligent manual input of said variables is requisite. The method of input shall adhere to the format "MODBUSHBIT[n]," where n corresponds to integers ranging from 0 to 65535.

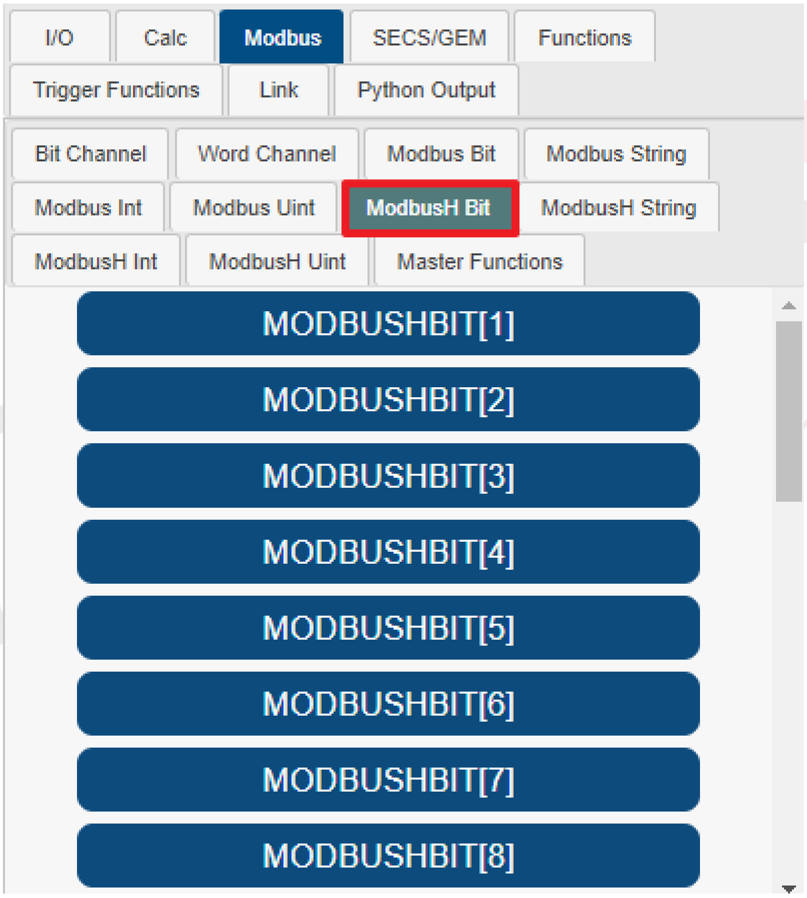


Figure 4-345 Selecting ModbusH Bit for Variable and Function Input in Modbus

4.2.2.6.9 ModbusH String

The ModbusH strings afford access to the local Holding Register (4x) strings for both reading and writing. The comprehensive listing of addresses for the local Holding Register can be found in section 4.1.2.3. Within the domain of variables and function inputs, opt for the ModbusH strings of the Modbus protocol (as depicted in Figure 4-346). This action shall promptly render the display of ModbusH string variables. This software facilitates utilization of Modbus H string addresses ranging from 1 to 65535. However, the interface exclusively accommodates the display of 25 at a given moment. In the event that a user intends to employ ModbusH string variables surpassing the limited interface display, personal entry of the ModbusH string variables is mandatory. The prescribed method of entry adheres to the format "MODBUSHSTR[n]," where n encompasses integers ranging from 0 to 65535.

Kindly note: If said variable lacks data, the corresponding field shall manifest as "blank."

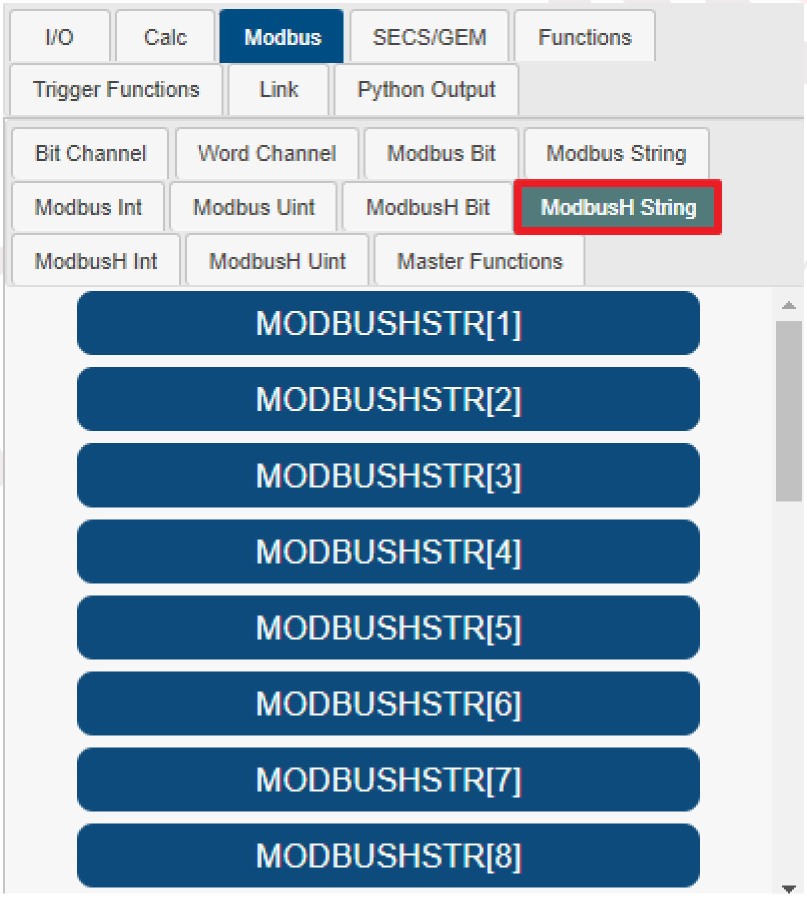


Figure 4-346 Selecting ModbusH String for Variable and Function Input in Modbus

4.2.2.6.10 ModbusH Int

The ModbusH Int protocol facilitates access to the local Holding Register (4x) numeric values for both reading and writing. The comprehensive list of addresses for the local Holding Register can be found in section 4.1.2.3. Within the domain of variables and function inputs, choose the ModbusH Int from the Modbus protocol (as depicted in Figure 4-347). This selection shall promptly bring forth the presentation of ModbusH Int variables. This software extends its compatibility to embrace Modbus H Int addresses spanning from 1 to 65535. Nonetheless, the interface exclusively showcases 25 at any given point in time. In the event that a user seeks to employ ModbusH Int variables exceeding the interface's limited display, the user is compelled to undertake independent input of ModbusH Int variables. The method of input is delineated as "MODBUSHINT[n]," where n encompasses integers ranging from 0 to 65535. Each individual ModbusH Int consists of 32 bits, encapsulating an integer furnished with both positive and negative magnitudes. The permissible scope of values encompasses -2,147,483,648 to 2,147,483,647.

Please take note: When encountering instances where data is configured as strings, a preliminary conversion to numerical values shall occur. In cases where the conversion proves unsuccessful, a representation of 0 shall be employed.

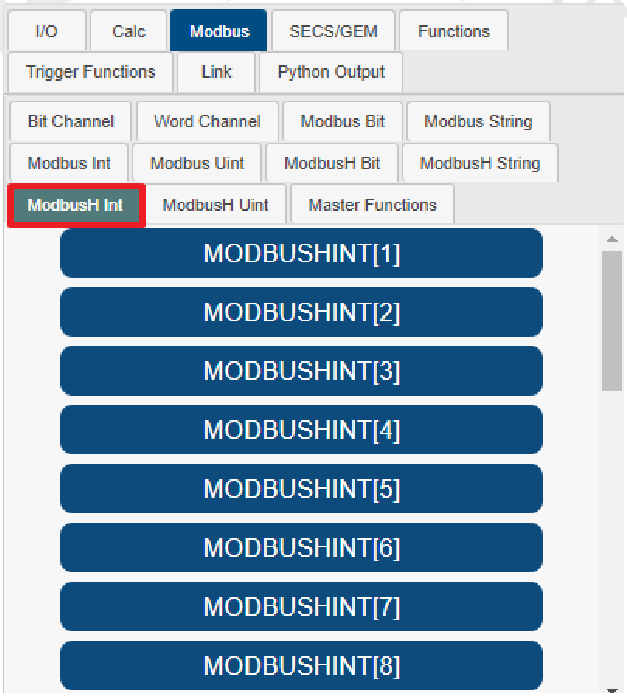


Figure 4-347 Selecting ModbusH Int for Variable and Function Input in Modbus

4.2.2.6.11 ModbusH Uint

The ModbusH Uint protocol facilitates access to the local Holding Register (4x) numeric values for both reading and writing. The comprehensive list of addresses for the local Holding Register can be found in section 4.1.2.3. Within the realm of variables and function inputs, select the ModbusH Uint from the Modbus protocol (as depicted in Figure 4-348). This choice shall promptly reveal the display of ModbusH Uint variables. This software affords utilization of Modbus H Uint addresses spanning from 1 to 65535. However, the interface exclusively presents 25 at any single instance. In the scenario that a user intends to employ ModbusH Uint variables exceeding the confines of the interface's display, personal input of ModbusH Uint variables is necessitated. The mode of input is articulated as "MODBUSHUINT[n]," with n embodying integers ranging from 0 to 65535. Each individual ModbusH Uint comprises 32 bits, forming an integer void of both positive and negative qualities. The allowable range of values extends from 0 to 4,294,967,295.

Please take heed: In the presence of data configurations expressed as strings, a preliminary conversion into numerical values will transpire. In situations where the conversion meets an impasse, a representation of 0 shall be embraced.

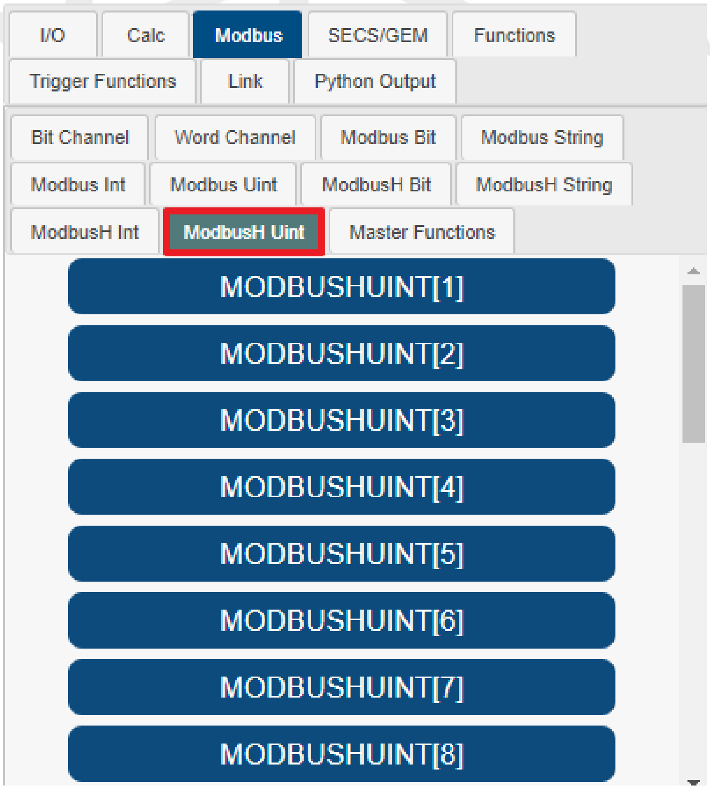


Figure 4-348 Selecting ModbusH Uint for Variable and Function Input in Modbus

4.2.2.6.12 ModbusH Uint16

The ModbusH Uint16 protocol provides the means for accessing local Holding Register (4x) numeric values, facilitating both reading and writing. The comprehensive list of addresses for the local Holding Register can be found in section 4.1.2.3. Within the realm of variables and function inputs, opt for the ModbusH Uint16 from the Modbus protocol (as depicted in Figure 4-349). This selection shall promptly manifest the display of ModbusH Uint16 variables. This software accommodates the utilization of Modbus H Uint16 addresses spanning from 1 to 65535. Nevertheless, the interface exclusively showcases a mere 25 at any single juncture. In the eventuality that a user aspires to employ ModbusH Uint16 variables exceeding the restricted interface display, personalized input of ModbusH Uint16 variables is incumbent. The prescribed method of input is rendered as "MODBUSHUINT16[n]," where n assumes integers ranging from 0 to 65535. Each distinct ModbusH Uint16 encompasses 16 bits, comprising an integer unburdened by both positive and negative attributes. The permissible spectrum of values extends from 0 to 65535.

Kindly note: When faced with scenarios involving data configurations expressed as strings, an initial conversion into numerical values will be undertaken. In instances where the conversion encounters an impasse, an embrace of 0 shall serve as representation.

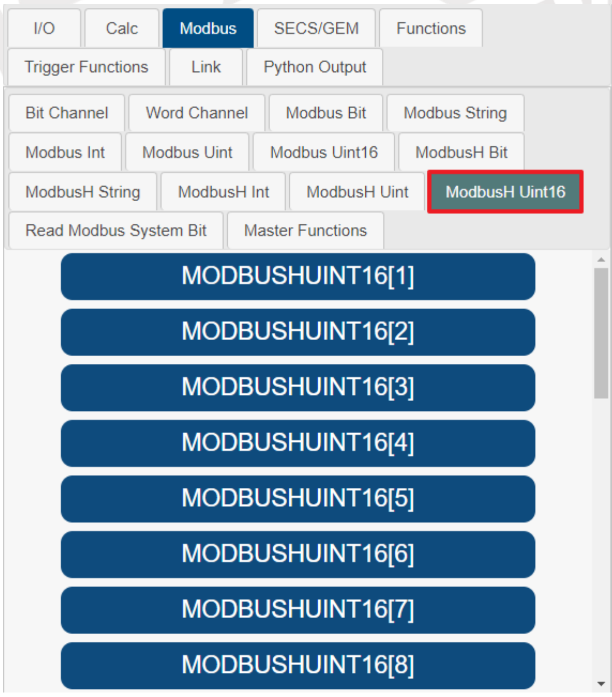


Figure 4-349 Selecting ModbusH Uint16 for Variable and Function Input in Modbus

4.2.2.6.13 Read Modbus System Bit

Reading nDAS system Modbus bits can be used to obtain the current status of the system. This software has a total of three system Modbus Bit planned, and their detailed descriptions are shown in the Table 4-35. The communication field is Input Status (1x).

Table 4-35 Modbus System Bit Table

Address(1x)	Content
12001	Presence of user login: If a user is logged in, the value is 1; if no user is logged in, the value is 0.
12002	System hard disk capacity of 0.5 gigabytes assessment: If the capacity is lower, the value is 1; if higher, the value is 0.
12003	System hard disk capacity of 1 gigabyte assessment: If the capacity is lower, the value is 1; if higher, the value is 0.

4.2.2.6.14 Master Functions

The Master Functions in Modbus can be utilized to read data from other Modbus Slave/Server. To access the function selection table, choose the Master Functions option in the Variables and Functions input area of Modbus (as shown in Figure 4-350).

Note：Before using this feature, it is necessary to configure the Modbus Master/Client connection in the communication page, for which the setup method can be found in section 4.4.2.1。

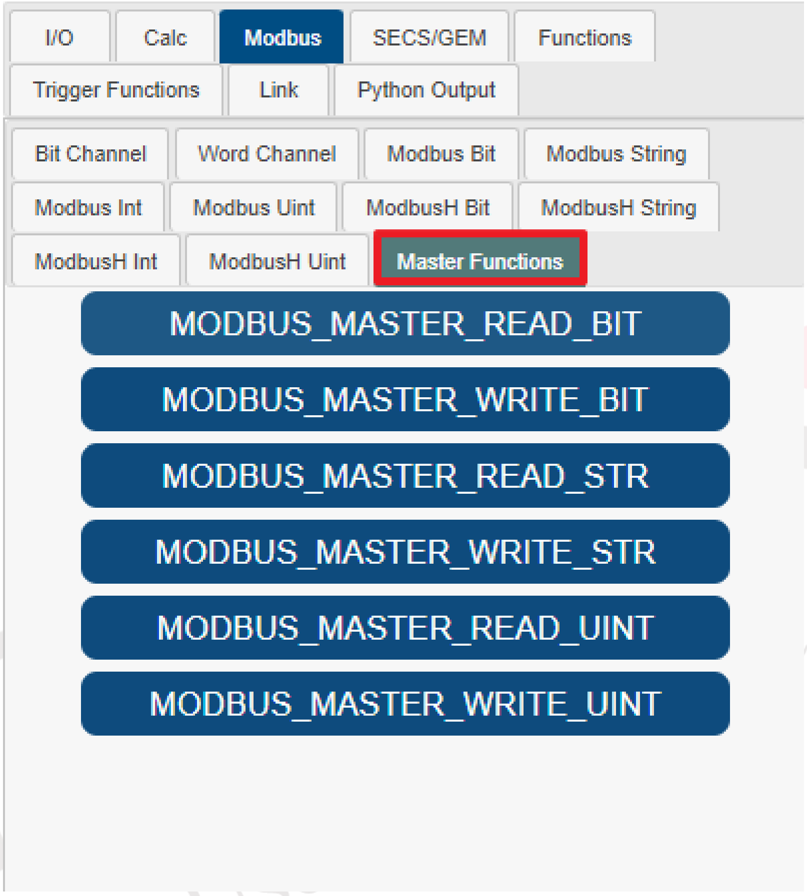


Figure 4-350 Selecting Function for Variable and Function Input in Modbus

4.2.2.6.14.1 MODBUS_MASTER_READ_BIT

This function can be used to read bit data from a Modbus TCP Server or Modbus RTU Slave. Prior to using this function, you must first set up the Modbus TCP Client or Modbus RTU Master connection by following the instructions provided in sections 4.4.1 of Link. Once the setup is complete, click on MODBUS_MASTER_READ_BIT under the Master Functions in the Modbus section of the variable and function input area (as shown in Figure 4-351), which will add the MODBUS_MASTER_READ_BIT function to the script editing area (as shown in Figure 4-352). MODBUS_MASTER_READ_BIT (id, prefix, regs, sec, device_id) function requires four parameters, each representing the following：

id：Select the Modbus Client/Slave connection to use by choosing the link in the variable and function input area (as shown in Figure 4-353) and clicking on the desired Modbus connection (as shown in Figure 4-354). This will add it to the script editor (as shown in Figure 4-355).

prefix：The Modbus function code to read, where 0 represents Coil Status and 1 represents Input Status.

regs：The address of the registers to read.

sec：The time interval in seconds during which this funciton will not be executed again. If set to 0, there will be no time interval.

device_id：modbus slave id

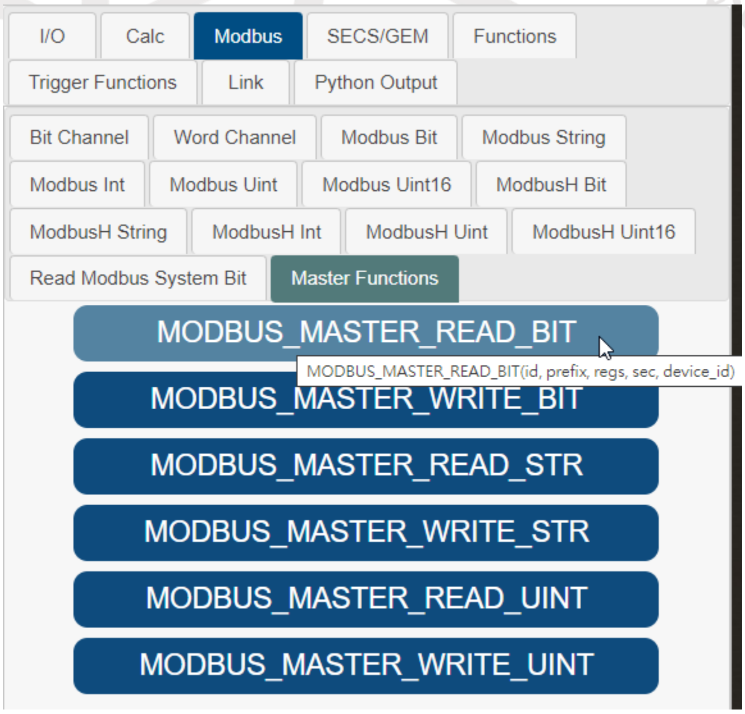


Figure 4-351 Click on "MODBUS_MASTER_READ_BIT"

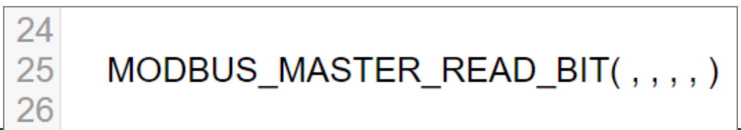


Figure 4-352 Add the MODBUS_MASTER_READ_BIT function in the script editing area

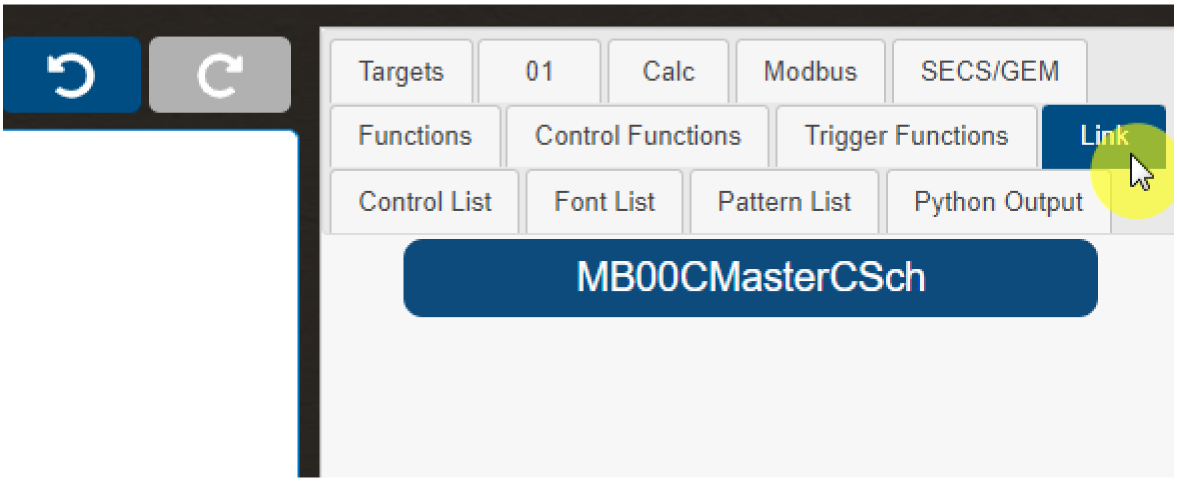


Figure 4-353 Choose the link in the variables and functions input area

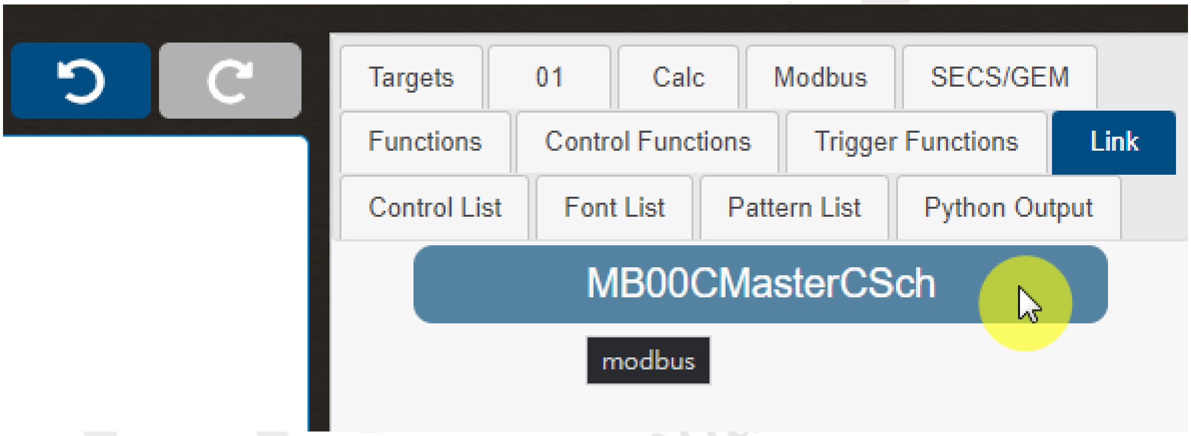


Figure 4-354 Click on the Modbus connection to be used.

```
MODBUS_MASTER_READ_BIT("MB00CMasterCSch" , , , , )
```

Figure 4-355 Add the Modbus connection in the script editing area

4.2.2.6.14.2 MODBUS_MASTER_WRITE_BIT

This function can be used to write bit data from a Modbus TCP Server or Modbus RTU Slave. Prior to using this function, you must first set up the Modbus TCP Client or Modbus RTU Master connection by following the instructions provides in section 4.4.1 of Link. Once the setup is complete, click on MODBUS_MASTER_WRITE_BIT under the Master Functions in the Modbus section of the variable and function input area (as shown in Figure 4-356), which will add the MODBUS_MASTER_WRITE_BIT function to the script editing area (as shown in Figure 4-357).

MODBUS_MASTER_WRITE_BIT (id, prefix, regs, value, sec, device_id) function requires five parameters, each representing the following：

id：Select the Modbus Client/Slave connection to use by choosing the link in the variable and function input area (as shown in Figure 4-358) and clicking on the desired Modbus connection (as shown in Figure 4-359). This will add it to the script editor (as shown in Figure 4-360).

prefix：The Modbus function code to write, where 0 represents Coil Status.

regs：The address of the register to be written.

value：The value to be written.

sec：The time interval in seconds during which this funciton will not be executed again. If set to 0, there will be no time interval.

device_id：modbus slave id

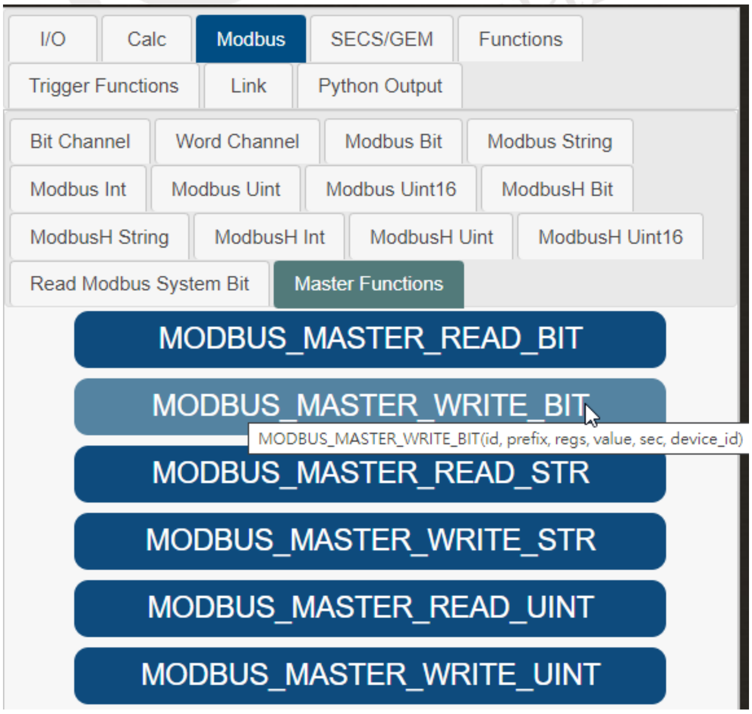


Figure 4-356 Click on "MODBUS_MASTER_WRITE_BIT"

```
MODBUS_MASTER_WRITE_BIT( , , , , , )
```

Figure 4-357 Add the MODBUS_MASTER_WRITE_BIT function in the script editing area

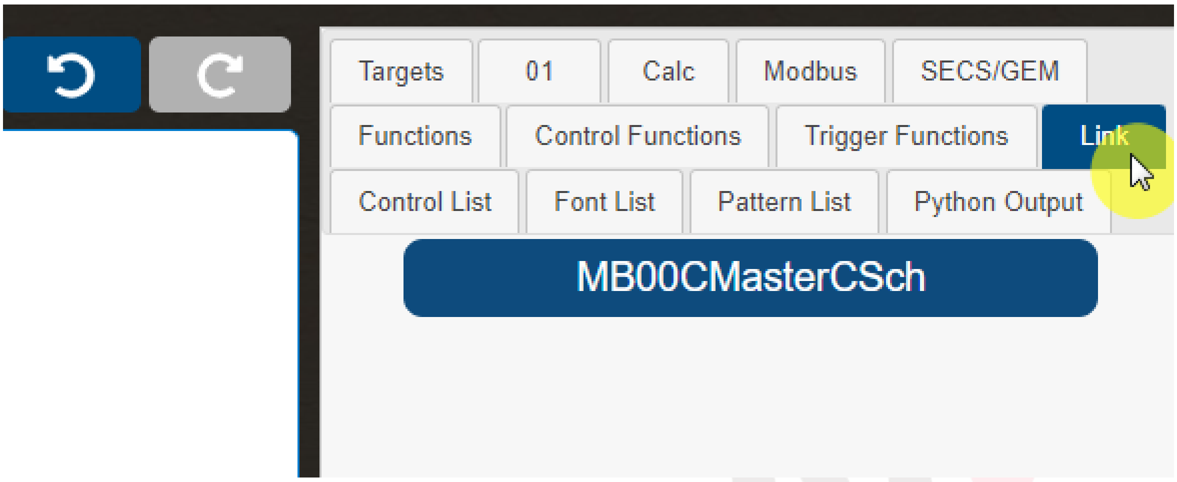


Figure 4-358 Choose the link in the variables and functions input area

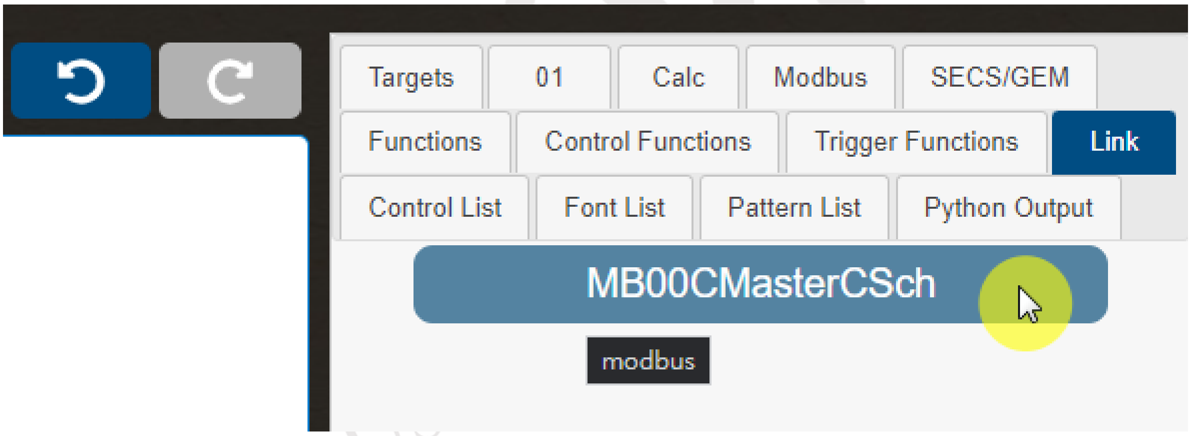


Figure 4-359 Click on the Modbus connection to be used.

```
MODBUS_MASTER_WRITE_BIT("MB00CMasterCSch" , , , , , )
```

Figure 4-360 Add the Modbus connection in the script editing area

4.2.2.6.14.3 MODBUS_MASTER_READ_STR

This function can be used to read string data from a Modbus TCP Server or Modbus RTU Slave in ASCII format. Prior to using this function, you must first set up the Modbus TCP Client or Modbus RTU Master connection by referring to sections 4.4.1 in the Link. Once the setup is complete, click on MODBUS_MASTER_READ_STR under the Master Functions in the Modbubs section of the variable and function input area (as shown in Figure 4-361), which will add the MODBUS_MASTER_READ_STR function to the script editing area (as shown in Figure 4-362).

MODBUS_MASTER_READ_STR (id, prefix, regs, regs_count, sec, device_id) function requires five parameters, each representing the following：

id：Select the Modbus Client/Slave connection to use by choosing the link in the variable and function input area (as shown in Figure 4-363) and clicking on the desired Modbus connection (as shown in Figure 4-364). This will add it to the script editor (as shown in Figure 4-365).

prefix：The Modbus function code to read, where 3 represents Input Register and 4 represents Holding Register.

regs：The address of the registers to read.

regs_count：Number of registers to be read in sequence, starting from the address.

sec：The time interval in seconds during which this funciton will not be executed again. If set to 0, there will be no time interval.

device_id：modbus slave id

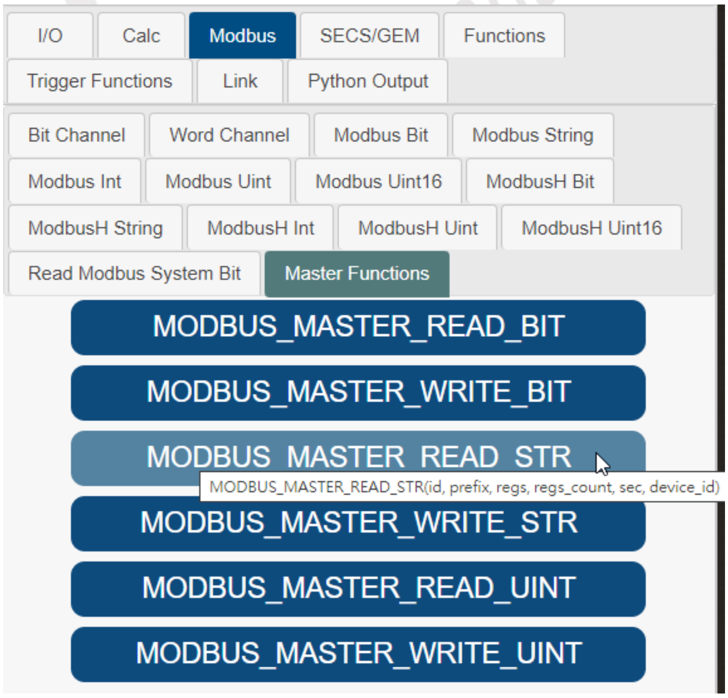


Figure 4-361 Click on "MODBUS_MASTER_READ_STR"

```
MODBUS_MASTER_READ_STR( , , , , , )
```

Figure 4-362 Add the MODBUS_MASTER_READ_STR function in the script editing area

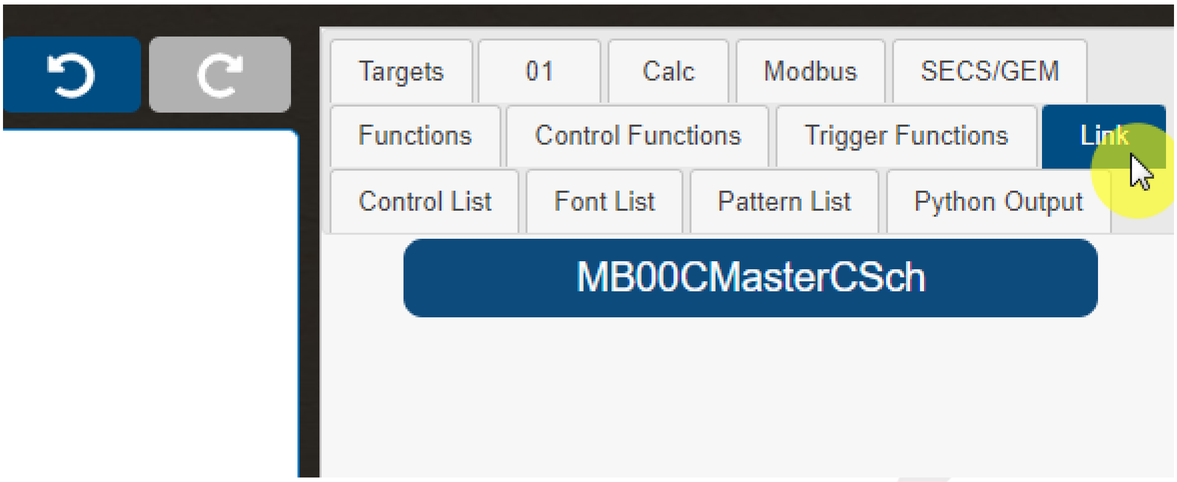


Figure 4-363 Choose the link in the variables and functions input area

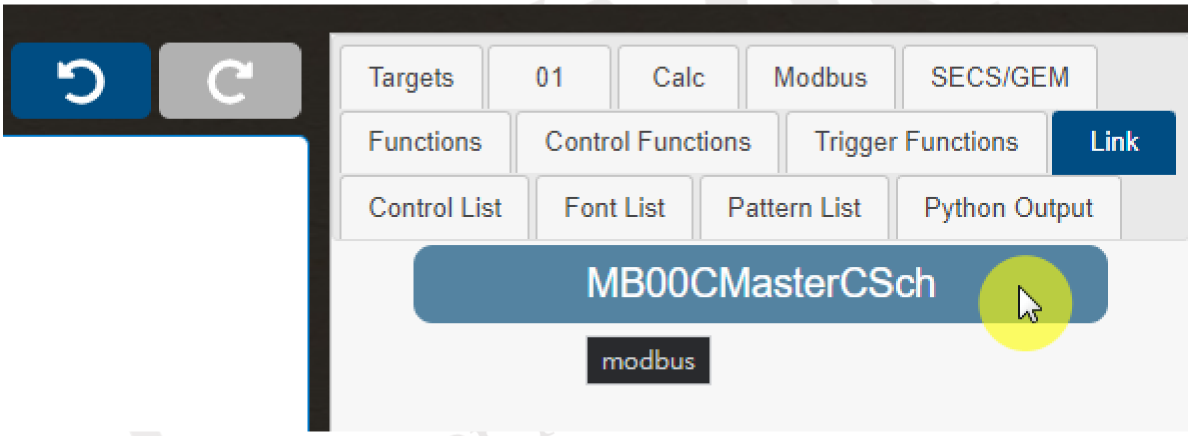


Figure 4-364 Click on the Modbus connection to be used.

```
MODBUS_MASTER_READ_STR("MB00CMasterCSch" , , , , , )
```

Figure 4-365 Add the Modbus connection in the script editing area

4.2.2.6.14.4 MODBUS_MASTER_WRITE_STR

This function can be used to write string data to Modbus TCP Server or Modbus RTU Slave. The string data that can be written is in ASCII format. Prior to using this function, it is necessary to configure the connection to the Modbus TCP Client or Modbus RTU Master by following the instructions in sections 4.4.1. Once the setup is complete, click on MODBUS_MASTER_WRITE_STR under the Master Functions in the Modbubs section of the variable and function input area (as shown in Figure 4-366, which will add the MODBUS_MASTER_WRITE_STR function to the script editing area (as shown in Figure 4-367).

MODBUS_MASTER_WRITE_STR (id, prefix, regs, value, sec, device_id) function requires five parameters, each representing the following：

id：Select the Modbus Client/Slave connection to use by choosing the link in the variable and function input area (as shown in Figure 4-368) and clicking on the desired Modbus connection (as shown in Figure 4-369). This will add it to the script editor (as shown in Figure 4-370).

prefix：The Modbus function code to write, where 4 represents Holding Register.

regs：The address of the register to be written.

value：The value to be written.

sec：The time interval in seconds during which this funciton will not be executed again. If set to 0, there will be no time interval.

device_id：modbus slave id

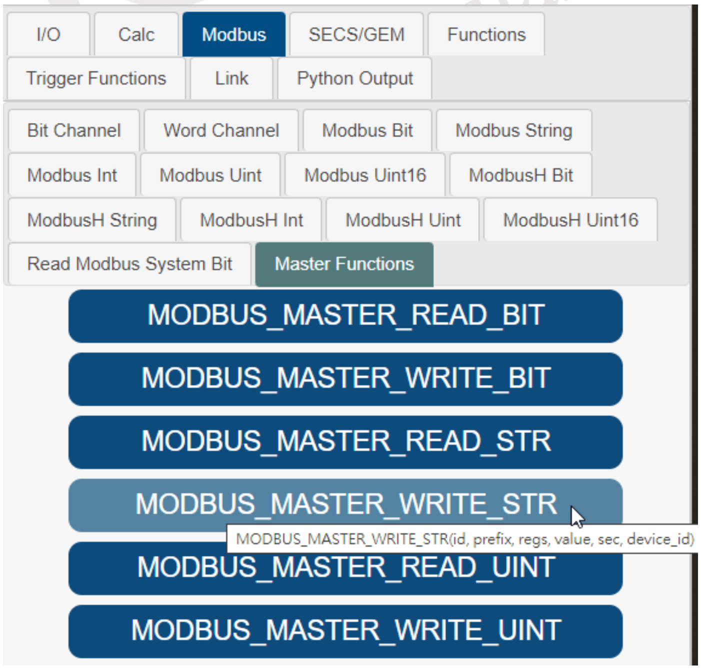


Figure 4-366 Click on "MODBUS_MASTER_WRITE_STR"

```
MODBUS_MASTER_WRITE_STR( , , , , , )
```

Figure 4-367 Add the MODBUS_MASTER_WRITE_STR function in the script editing area

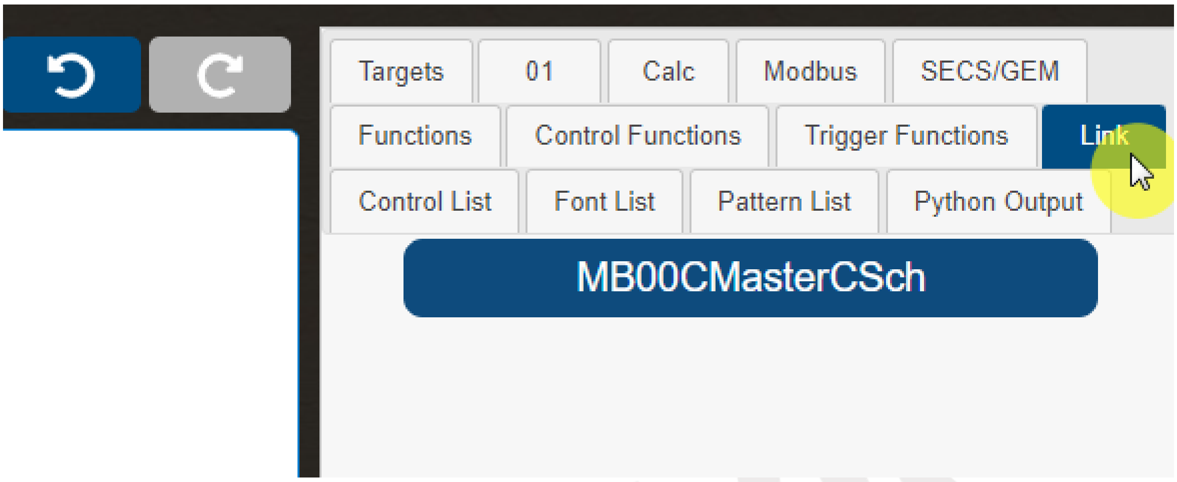


Figure 4-368 Choose the link in the variables and functions input area

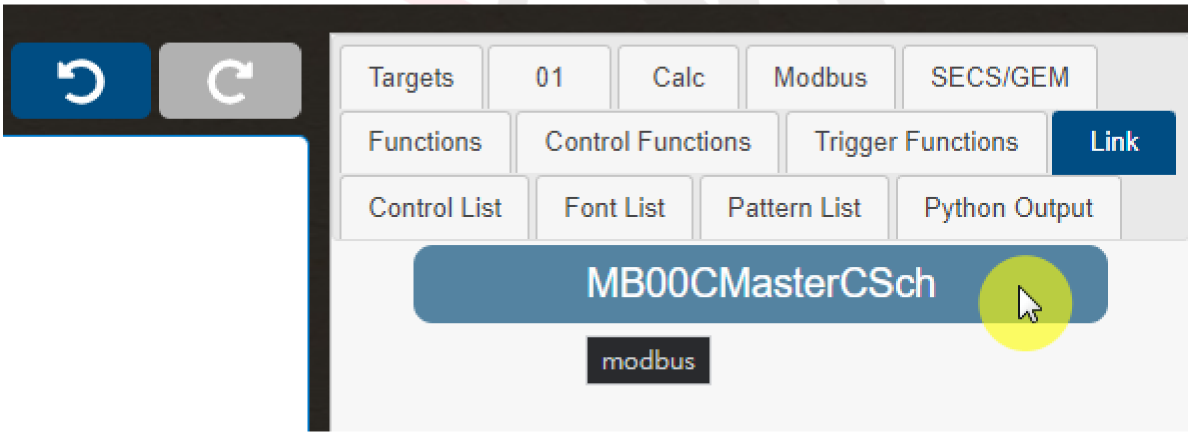


Figure 4-369 Click on the Modbus connection to be used.

```
MODBUS_MASTER_WRITE_STR("MB00CMasterCSch" , , , , , )
```

Figure 4-370 Add the Modbus connection in the script editing area

4.2.2.6.14.5 MODBUS_MASTER_READ_UINT

This function can be used to read unsigned integer data from a Modbus TCP Server or Modbus RTU Slave. The string data format that can be read is Uint16. Prior to using this function, it is necessary to configure the connection to the Modbus TCP Client or Modbus RTU Master by following the instructions in sections 4.4.1. Once the setup is complete, click on MODBUS_MASTER_READ_UINT under the Master Functions in the Modbubs section of the variable and function input area (as shown in Figure 4-371), which will add the MODBUS_MASTER_READ_UINT function to the script editing area (as shown in Figure 4-372).

MODBUS_MASTER_READ_UINT (id, prefix, regs, sec, device_id) function requires four parameters, each representing the following：

id：Select the Modbus Client/Slave connection to use by choosing the link in the variable and function input area (as shown in Figure 4-373) and clicking on the desired Modbus connection (as shown in Figure 4-374). This will add it to the script editor (as shown in Figure 4-375).

prefix：The Modbus function code to read, where 3 represents Input Register and 4 represents Holding Register.

regs：The address of the registers to read.

sec：The time interval in seconds during which this funciton will not be executed again. If set to 0, there will be no time interval.

device_id：modbus slave id

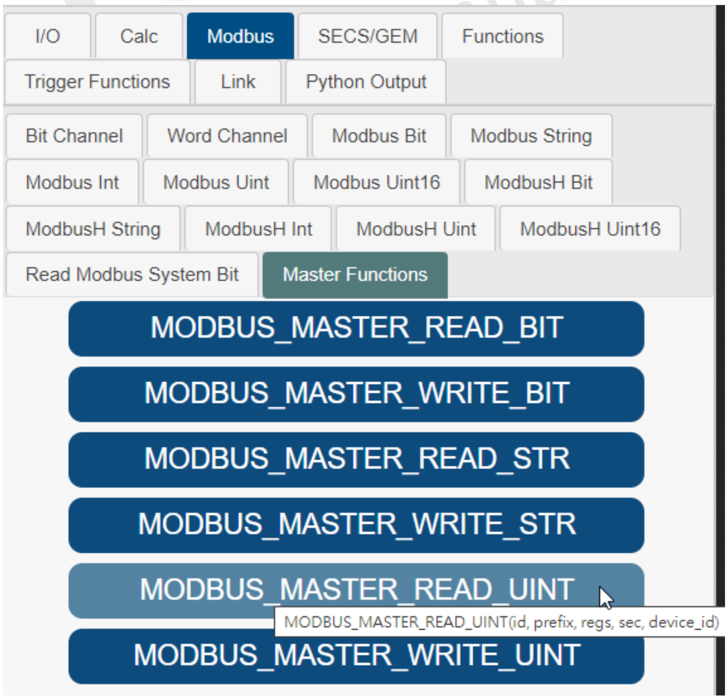


Figure 4-371 Click on "MODBUS_MASTER_READ_UINT"

```
MODBUS_MASTER_READ_UINT( , , , , )
```

Figure 4-372 Add the MODBUS_MASTER_READ_UINT function in the script editing area

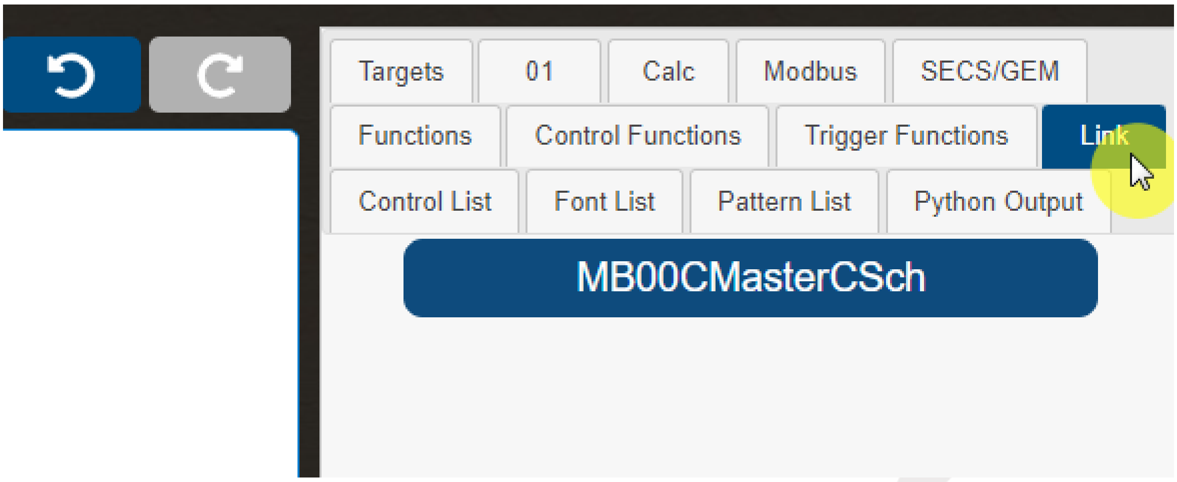


Figure 4-373 Choose the link in the variables and functions input area

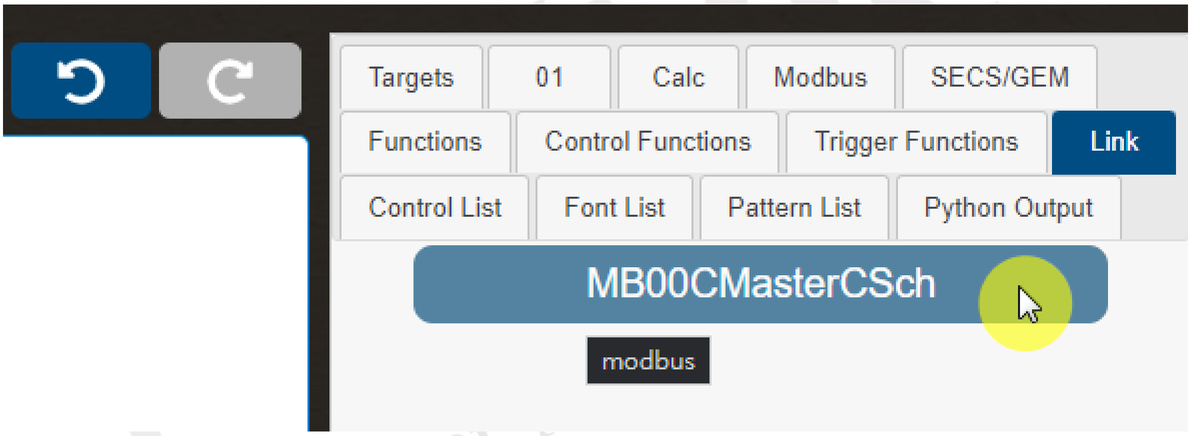


Figure 4-374 Click on the Modbus connection to be used.

```
MODBUS_MASTER_READ_UINT("MB00CMasterCSch" , , , , )
```

Figure 4-375 Add the Modbus connection in the script editing area

4.2.2.6.14.6 MODBUS_MASTER_WRITE_UINT

This function can be used to write unsigned integer data to Modbus TCP Server or Modbus RTU Slave. The data format that can be written is Uint16. Prior to using this function, it is necessary to configure the connection to the Modbus TCP Client or Modbus RTU Master by following the instructions in sections 4.4.1. Once the setup is complete, click on MODBUS_MASTER_WRITE_UINT under the Master Functions in the Modbubs section of the variable and function input area (as shown in Figure 4-376), which will add the MODBUS_MASTER_WRITE_UINT function to the script editing area (as shown in Figure 4-377).

MODBUS_MASTER_WRITE_UINT (id, prefix, regs, value, sec, device_id) function requires five parameters, each representing the following：

id：Select the Modbus Client/Slave connection to use by choosing the link in the variable and function input area (as shown in Figure 4-378) and clicking on the desired Modbus connection (as shown in Figure 4-379). This will add it to the script editor (as shown in Figure 4-380).

prefix：The Modbus function code to write, where 4 represents Holding Register.

regs：The address of the registers to write.

value：The value to be written.

sec：The time interval in seconds during which this funciton will not be executed again. If set to 0, there will be no time interval.

device_id：modbus slave id

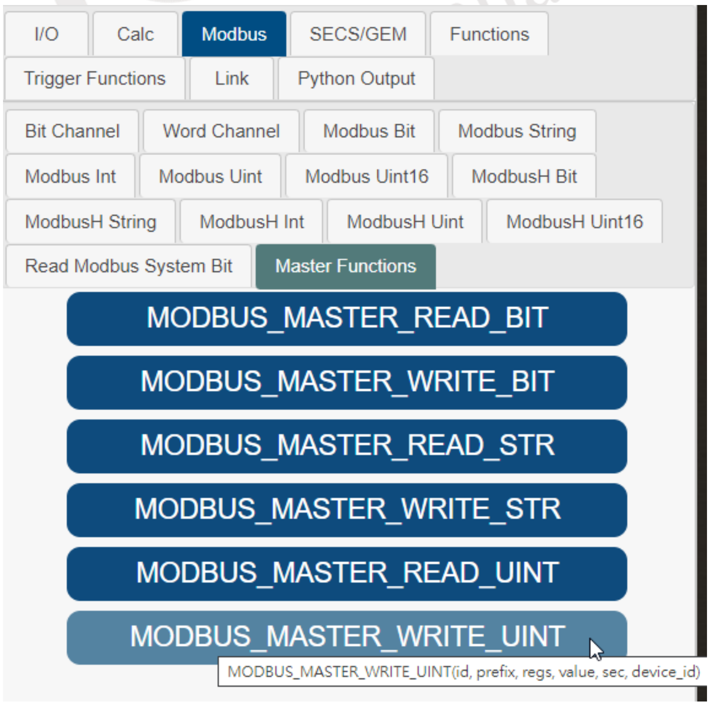


Figure 4-376 Click on "MODBUS_MASTER_WRITE_UINT"

```
MODBUS_MASTER_WRITE_UINT( , , , , , )
```

Figure 4-377 Add the MODBUS_MASTER_WRITE_UINT function in the script editing area

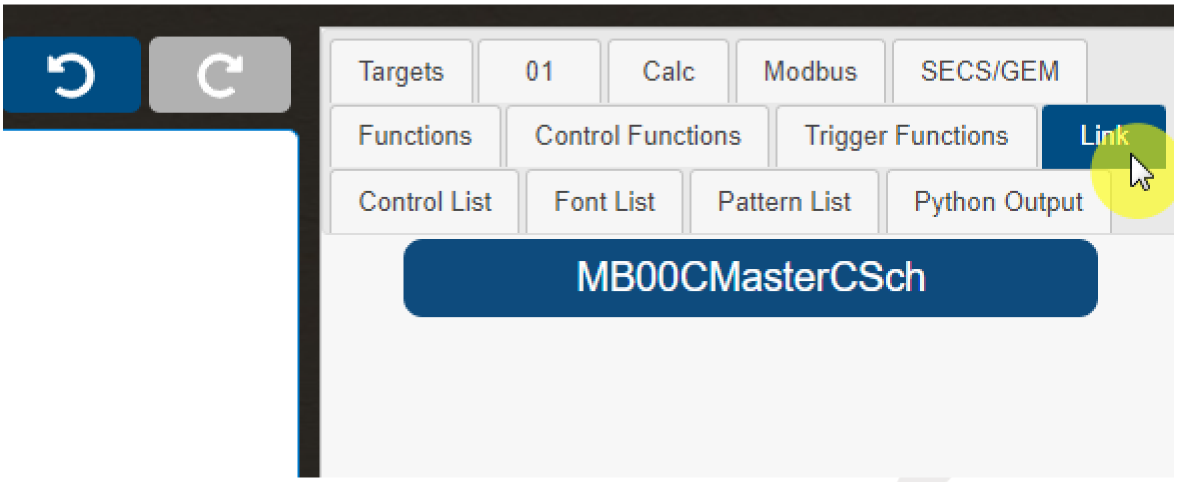


Figure 4-378 Choose the link in the variables and functions input area

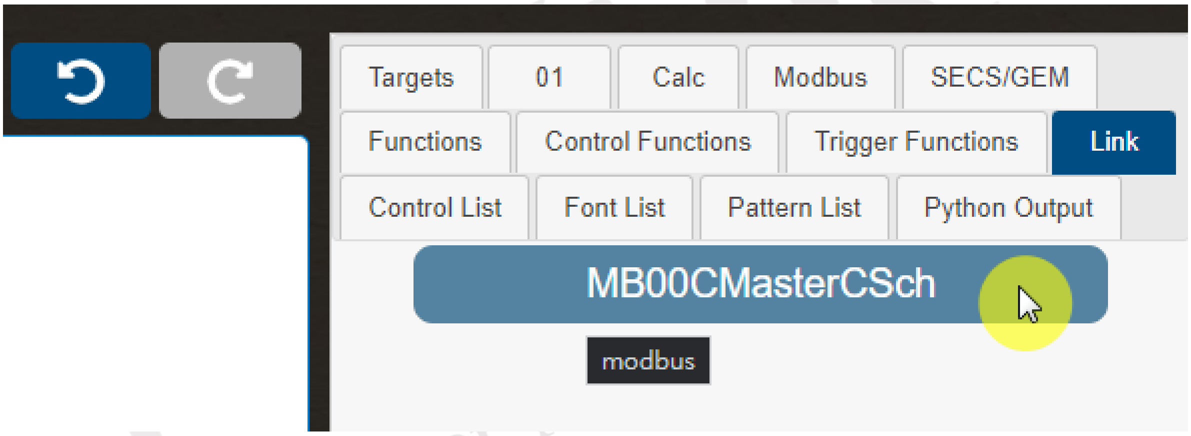


Figure 4-379 Click on the Modbus connection to be used.

```
MODBUS_MASTER_WRITE_UINT("MB00CMasterCSch" , , , , , )
```

Figure 4-380 Add the Modbus connection in the script editing area

4.2.2.7 Link

This function represents the communication link to be used. To select a link in the variable and function input area, a list of links is displayed, as shown in Figure 4-381. The communication variables appearing here will be synchronized with the number of communication settings in the project link page (as shown in Figure 4-382).

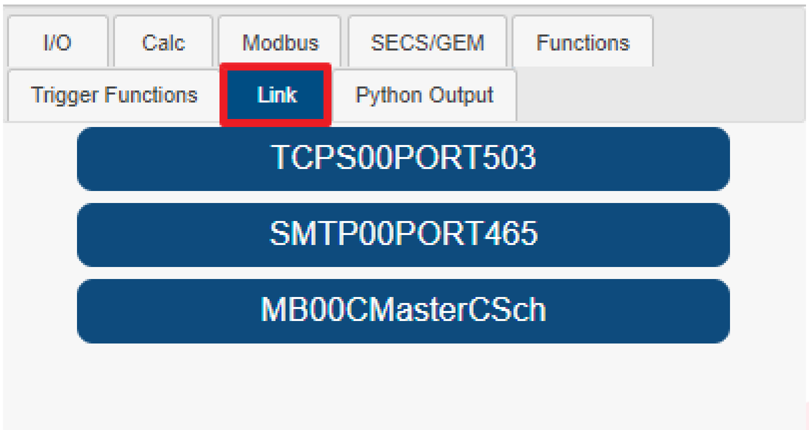


Figure 4-381 Choose the link in the variables and functions input area

Figure 4-382 Communication settings within the project Link

4.2.3 Custom Function Names and Input Parameters

Declare a function outside the MainLoop of nDAS script, which can be triggered and sent parameters to nDAS via RESTful, and the passed parameters can be executed together with the function. The writing method is as follows: In the example shown in Figure 4-383, he function content is to display the two received parameters in the Python output, then add and multiply the two parameters, and finally return the operation result. The example in the figure triggers the Demo_trigger function through RESTful, and passes in two parameters, Demo_1 and Demo_2 (as shown in Figure 4-384), and finally executes the result (as shown in Figure 4-385) and returns it (as shown in Figure 4-386).

Note：The supported data types for the return value are limited to Int, String, Double, Boolean, and Dictionary.

```
def Function_Name(**kwargs):
    Function_Content…
    print(kwargs['Parameter name'])
    return {'Return_Name_1':Return_Value_1,
'Return_Name_2':Return_Value_2}
```

```
from vic import *

def Demo_trigger(**kwargs):
    print("Demo_1=", kwargs["Demo_1"])
    print("Demo_2=", kwargs["Demo_2"])
    add_result = int(kwargs["Demo_1"]) + int(kwargs["Demo_2"])
    multi_result = int(kwargs["Demo_1"]) * int(kwargs["Demo_2"])

    return {"add_result":add_result, "multi_result":multi_result}

def mainLoop(id, info, data):
    pass
```

Figure 4-383 Example of nDAS script writing

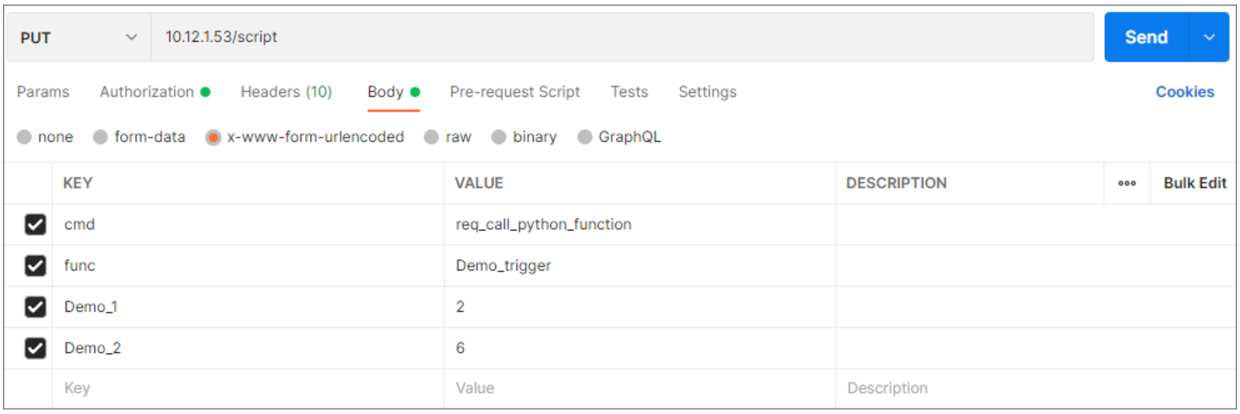


Figure 4-384 RESTful Command

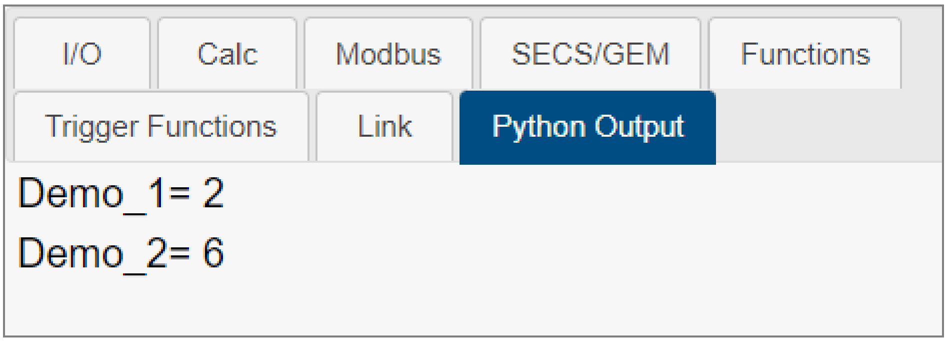


Figure 4-385 Python Output

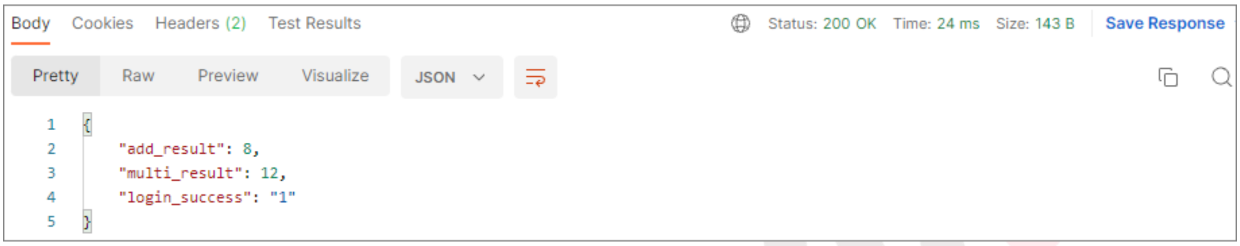


Figure 4-386 Retrieving the Return Result via RESTful

4.2.4 Importing custom Python

The two ways of importing in Python are importing files and importing packages, which will be described in subsequent sections.

4.2.4.1 Importing Files

nDAS is a Linux system, and the files should be placed in the /opt/nDAS/bin/py directory. Here's how to import your own pre-written (no longer modified) py file：Assuming the file name of the py file is Demo_Import.py, and its content is shown in Figure 4-387, the import method and execution are shown in Figure 4-388, and the execution result is shown in Figure 4-389.

```
def demo():
    print('Hello World')
```

Figure 4-387 Demo_Import.py Content

```
from vic import *
import Demo_Import


def mainLoop(info, arr):
    Demo_Import.demo()
```

Figure 4-388 Importing and executing a py file

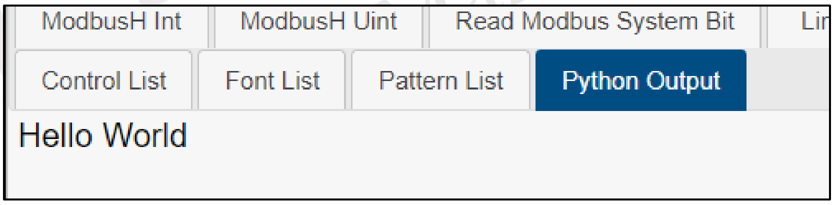


Figure 4-389 Result of executing nDAS

The following will introduce the import of a self-written and still under development (subject to modification) py file: Assuming that the file name is Demo_Develop_Import.py, the contents are shown in Figure 4-390. The import method and execution are shown in Figure 4-391, and the execution result is shown in Figure 4-392.

```
def demo_develop():
    print('Hello Developer')
```

Figure 4-390 Demo_Develop_Import.py Content

```
from vic import *
from imp import reload
import Demo_Develop_Import
reload(Demo_Develop_Import)

def mainLoop(info, arr):
    Demo_Develop_Import.demo_develop()
```

Figure 4-391 Importing method and executing

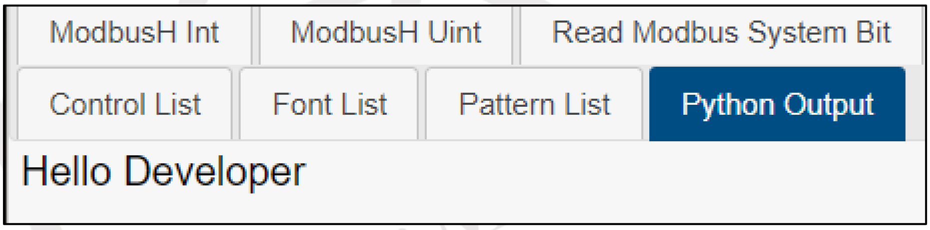


Figure 4-392 Result of executing nDAS

4.2.4.2 Importing Package

nDAS offers users the capability to autonomously install Python packages (Python 3.7). The underlying operational process involves executing "pip3 install." The installation procedure is initiated by clicking the "Install Package" button (as illustrated in Figure 4-393), thereby prompting the display of a window dedicated to the installation of Python packages (as depicted in Figure 4-394). Upon entry of the package name and subsequent activation of the "install" command (as illustrated in Figure 4-395), the installation process shall commence.

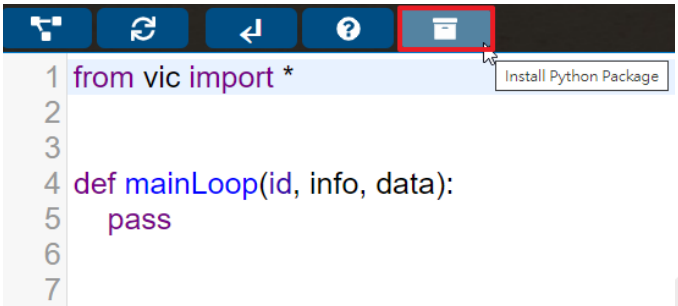


Figure 4-393 Click on install Python package button

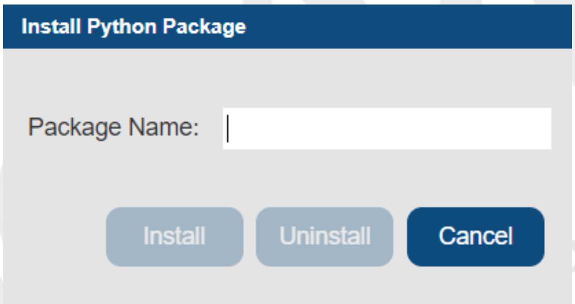


Figure 4-394 Show Dialog of installing Python package

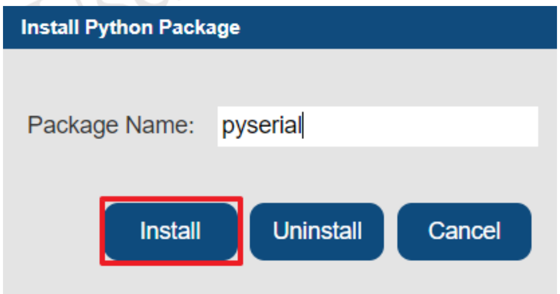


Figure 4-395 Click on button to install Python package(example for pyserial)

4.3 Monitor

In administrator mode of the nDAS monitor, click on the monitor icon (as shown in Figure 4-396).

Monitor content has I/O status, and calculation.

Note：If the script page has no content, the calculation will be hidden.

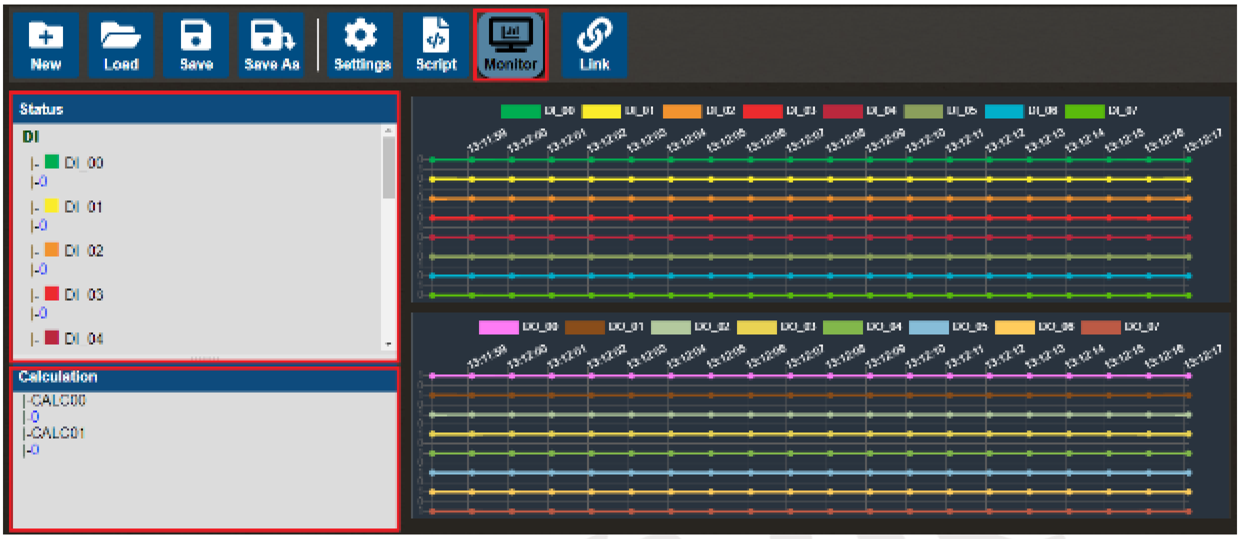


Figure 4-396 Click on the monitor icon

4.4 Link

By clicking on the link icon and selecting from the drop-down menu, as shown in Figure 4-397, the page will automatically switch to the configuration for the selected communication method. The software supports various communication methods, including TCP/IP, Modbus TCP, SMTP, LINE Notify, WeChat, Microsoft Teams, SECS/GEM, OPCUA Client, and RESTful, which can be used for data logging and transmission.

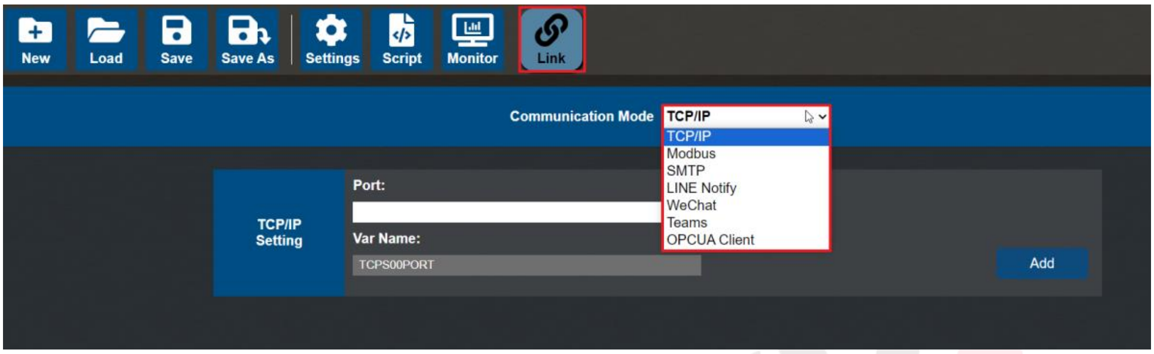


Figure 4-397 Click on link icon

4.4.1 TCP/IP

The configuration is shown in Figure 4-398, where the IP address is the computer's IP and the port is defined by the user (in this example, 503 is used as the port number). After completing the configuration, click the "Add" button to complete the connection. If the connection is successful, the connection count will change as shown in Figure 4-399. The TCP/IP output content of this software is in ASCII, which is parsed into a JSON string. The format of each data record is as follows (as shown in the structure table in Table 4-36)：

1. There is a header of one byte at the beginning, with the content "v".
2. The next 4 bytes represent the length of the recognition data for this record.

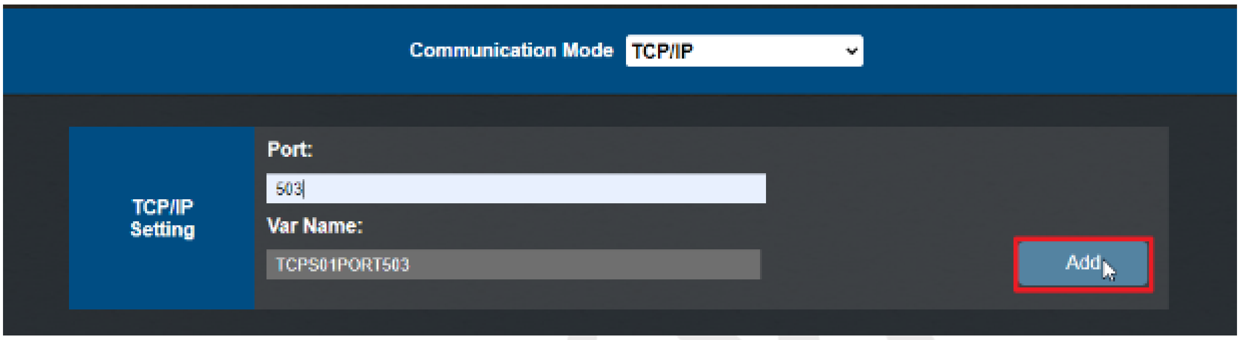


Figure 4-398 TCP/IP Configuration

Figure 4-399 Successful completion of TCP/IP configuration and establishment of a connection

Table 4-36 TCP/IP output data structure table

Header	Data Length	Recognition Data
1 byte	4 bytes	recognition data length

4.4.2 Modbus

4.4.2.1 TCP/IP Master

The Modbus TCP/IP configuration, as shown in Figure 4-400, allows users to set the IP address of the Modbus TCP/IP server to connect to, and the communication port can be defined by the user (default is 502). After completing the settings, click the “Add” button to add the connection. If the connection is successful, the system will display as shown in Figure 4-401. if the connection fails, the front block will turn red, as shown in Figure 4-402.

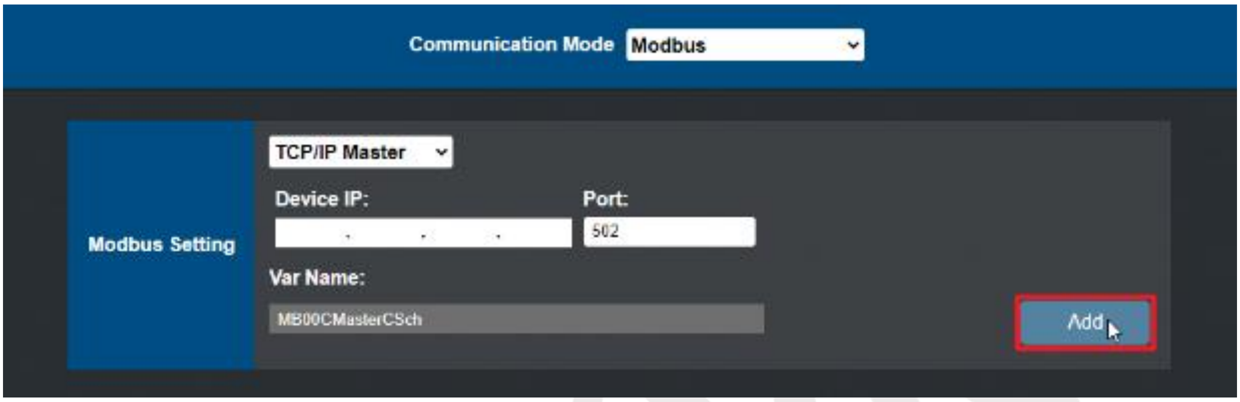


Figure 4-400 Modbus TCP/IP Master configuration

Figure 4-401 Modbus TCP/IP Master configuration completed and successfully connected

Figure 4-402 Modbus TCP/IP Master configuration completed but connection failed

4.4.3 SMTP

This software provides the function of connecting to an SMTP server, allowing the user to send notifications via email. To do so, the “SEND.EMAIL” function in the script must be used (please refer to 4.2.2.3.4 for instructions on how to use this function). The configuration is shown in Figure 4-403, and the parameter settings are shown in Table 4-37. After all the parameter settings have been completed, clicking the “Add” button to complete the addition of the connection.

Table 4-37 SMTP parameter settings table

Parameter	Content
Server Address	SMTP server address
Port	Communication port for the SMTP server
User Name	Account for the sender's email address to be used
Password	Password for the sender's email address to be used
Email	Email address for the sender's email address to be used
Var Name	Automatically generated variable name

Figure 4-403 SMTP configuration

4.4.4 LINE Notify

This software provides LINE Notify, which allows users to send notifications through LINE. To do so, the “SEND.LINE” function in the script must be used (please refer to 4.2.2.3.6 for instructions on how to use this function). The configuration is shown in Figure 4-404, and the parameter settings are shown in Table 4-38. After all the parameter settings have been completed, clicking the “Add” button to complete the addition of the connection. For the LINE Notify Token acquisition process, please refer to section 5.2. Restrictions are listed in Table 4-39.

Note：Official announcements should take precedence over this information.

Table 4-38 LINE Notify parameter settings table

Parameter	Content
Token	LINE Notify Token
GroupName	The final name of the generated variable
Var Name	Automatically generated variable name

Figure 4-404 LINE Notify configuration

Table 4-39 LINE Notify restrictions table

Name	Limitations
Transmitting pure text message	1000 times per hour
Transmitting messages containing image	50 times per hour
Maximum text length	1000 characters per message

4.4.5 Wechat

This software provides Wechat, which allows users to send notifications through WeChat. To do so, the “SEND.WECHAT” or “SEND.WECHAT_P”function in the script must be used (please refer to 4.2.2.3.7 and 4.2.2.3.8 for instructions on how to use this function). The configuration is shown in Figure 4-405, and the parameter settings are shown in Table 4-40. After all the parameter settings have been complete, clicking the “Add” button to complete the addition of the connection. Method to obrtain the appID and appsecret of Wechat can be found in section 5.3 of the documentation. The limitation of Wechat test accounts are illustrated in Table 4-41.

Note：Official announcements should take precedence over this information.

Table 4-40 Wechat parameter settings table

Parameter	Content
appID	appID for Wechat test account
appsecret	appsecret for Wechat test account
Var Name	Automatically generated variable name

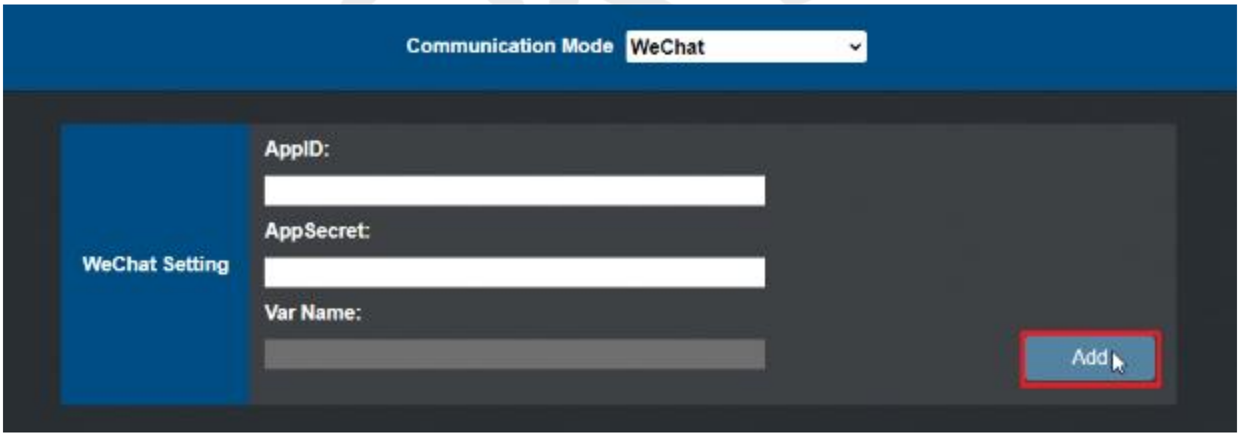


Figure 4-405 Wechat configuration

If there are any issues with the appID or appsecret, you can test them using the "WeChat Official Account Interface Debugging Tool" (https://mp.weixin.qq.com/debug/).

Table 4-41 WeChat Test Account Limitations

Name	Limitations
Sending text message	100,000 times per day
Updating the number of followers for test account	500 times per day

Template	Maximum of 10
Number of users that can be followed	Maximum of 100

If a user follows this testing official account after creating the WeChat link, they should click the "Update Group" button (as shown in Figure 4-406) in the created WeChat link module to update the number of followers who have followed the testing official account.

Figure 4-406 Click the "Update Group" button to update the number of followers

4.4.6 Teams

This software provides Microsoft Teams, which allows users to send notification through Teams. To do so, the “SEND.TEAMS” function in the script must be used (please refer to 4.2.2.3.9 for instructions on how to use this function). The configuratrion is shown in Figure 4-407, and the parameter settings are shown in Table 4-42. After all the parameter settings have been completed, clicking the “Add” button to complete the addition of the connection. To obtain a Teams Webhook, please refer to section 5.5.

Table 4-42 Teams parameter settings table

Parameter	Content
Name	Set the name of the Teams link
Webhook	Teams Webhook
Var Name	Automatically generated variable name

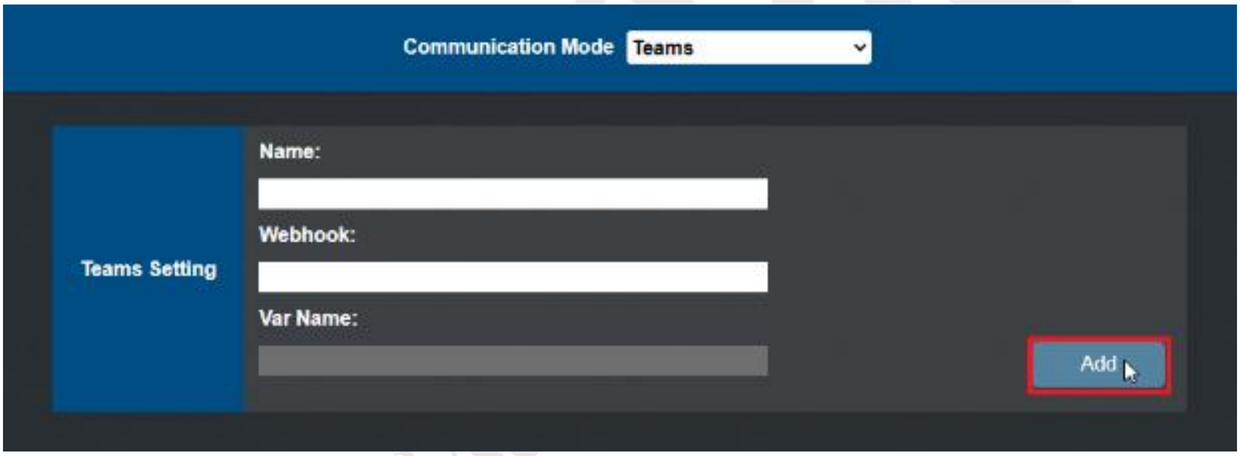


Figure 4-407 Teams configuration

4.4.7 OPCUA Client

This software offers OPCUA connection configuration, enabling it to act as a Host and connect to other OPCUA Servers. The configuration page is depicted in Figure 4-408, and the parameter settings are outlined in Table 4-43. Upon completing the configuration, clicking the "Add" button finalizes the new connection setup. If the connection is successful, the system will appear as shown in Figure 4-409; in the event of a failed connection, the respective section will turn red, as illustrated in Figure 4-410.

Table 4-43 OPCUA parameter settings table

Parameter name	Content
Url	Enter the URL of the OPCUA Server.
Security Policy	Specify the security policy.
Security Mode	Set the security mode.
Authentication	Choose the authentication method, which includes options for anonymous, username, and certificate.
Var Name	Enable automatic variable name generation.

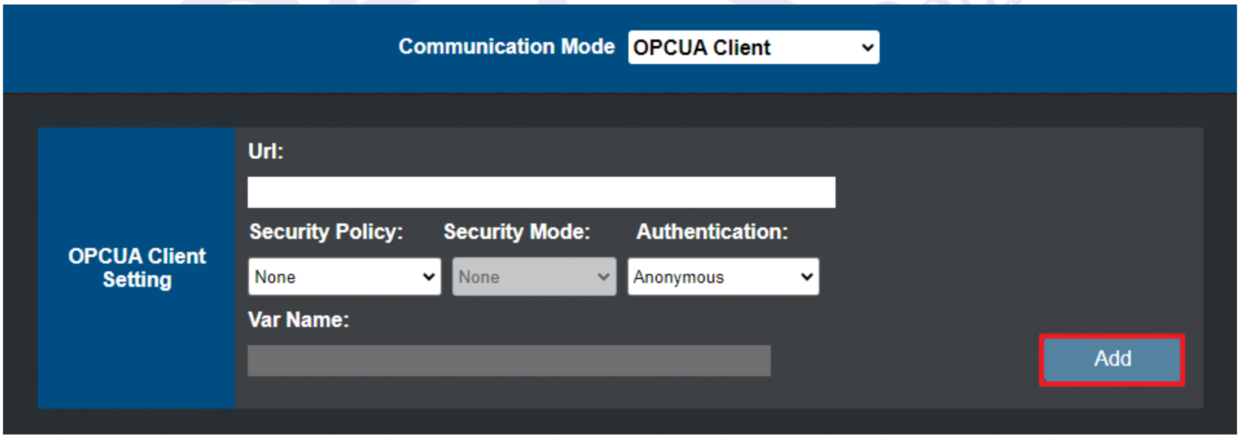


Figure 4-408 OPCUA connection configuration

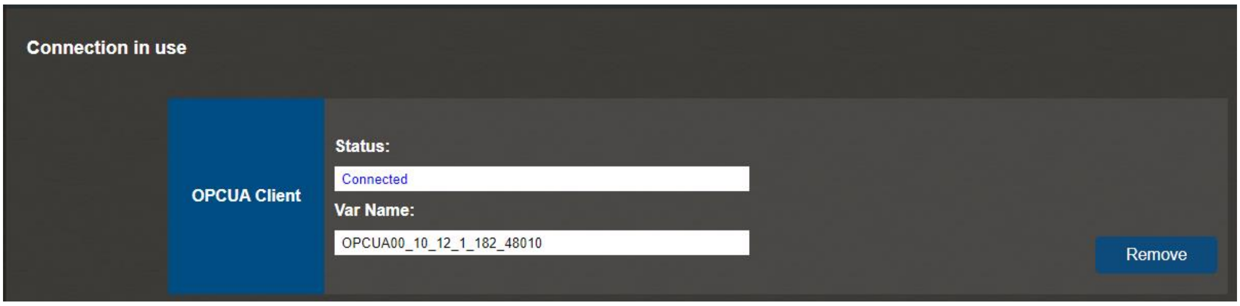


Figure 4-409 OPCUA Server configuration completed and successfully connected

Figure 4-410 OPCUA Server configuration completed but connection failed

4.5 RESTful

4.5.1 nDAS RESTful API

This software provides a RESTful API with JSON output format. To use the API, you need to enter the IP address of nDAS in the URL bar and add the following commands to view the data returned by nDAS. The explanation of the returned data content and format is provided in Table 4-45.

When using the RESTful API, you will need to input request authentication. The authentication method used is Basic Authentication. The parameters for RESTful API Basic Authentication are specified in Table 4-44 or can be referred to in section 4.1.1.1.2.

Table 4-44 RESTful API Basic Authentication

Parameter name	Content
Username	admin
Password	123456(default)

Table 4-45 RESTful API and format

/di_value/slot_0/ch_n		
Ch	R	Channel
En	R	Channel Mask
Md	R	Mode
Stat	R	DI Value
Cnting	R/W	DI Counter Switch
OvLch	R/W	Overflow Value
CtFq	R	DI Counter
Lch	R/W	Low to High Latch Value
Hch	R/W	High to Low Latch Value
/do_value/slot_0/ch_n		
Ch	R	Channel
En	R	Channel Mask
Md	R	Mode
Stat	R	DO Value
Val	R/W	DO Pulse
PsCtn	R/W	Pulse outputting is continuous or not
PsStop	R/W	DO Pulse Stop Status
PsIV	R/W	Incremental Pulse Output Value
/ai_value/slot_0/ch_n		

Ch	R	Channel
En	R	Channel Mask
Rng	R	Input Range
Val	R	AI Value
EgF	R	AI Enginnering Value
Evt	R	AI Status
LoA	R/W	Low Alarm Status
HiA	R/W	High Alarm Status
HVal	R	Maximum AI Value
HEgF	R	Maximum AI Enginnering Value
LVal	R	Minimum AI Value
LEgF	R	Minimum AI Enginnering Value
SVal	R	AI Value after Scaling
CLrL	R/W	Clear Minimum AI Value
CLrH	R/W	Clear Maximum AI Value
/expansion_bit/com_m/ch_n		
Ch	R	Channel
Val	R/W	Modbus Bit Value
Evt	R	Read Modbus Error Code
SID	R	Slave ID
Addr	R	Modbus Address
Prop	R	R, W, R/W mode
WEvt	R	Write Modbus Error Code
MAddr	R	Expansion Mapping address
/expansion_bit/tcp_m/ch_n		
Ch	R	Channel
Val	R/W	Modbus Bit Value
Evt	R	Read Modbus Error Code
SID	R	Slave ID
Addr	R	Modbus Address
Prop	R	R, W, R/W mode
WEvt	R	Write Modbus Error Code
MAddr	R	Expansion Mapping address
/expansion_word/com_m/ch_n		
Ch	R	Channel

Val	R/W	Modbus Word Value	
Evt	R	Read Modbus Error Code	
SID	R	Slave ID	
Addr	R	Modbus Address	
Prop	R	R, W, R/W mode	
WEvt	R	Write Modbus Error Code	
MAddr	R	Expansion Mapping address	
/expansion_word/tcp_m/ch_n			
Ch	R	Channel	
Val	R/W	Modbus Word Value	
Evt	R	Read Modbus Error Code	
SID	R	Slave ID	
Addr	R	Modbus Address	
Prop	R	R, W, R/W mode	
WEvt	R	Write Modbus Error Code	
MAddr	R	Expansion Mapping address	
/calc_value/ch_n			
CalcVal	R/W	Calc Operators Value	
/log_message			
PE	R	1	DI Event form
		2	DO Event form
		4	AI Event form
		8	AO Event form
		16	WDT Event form
		128	Periodic
UID	R	Unique Identifier	
MAC	R	MAC address	
TIM	R	Timestamp	
Record	R	Index	Recording I/O-type of the storage
		0	Invalid
		1	DI Logic status
		2	DI counter value
		3	DI Frequency value
		4	DO Logic status

		5	Reserved
		6	Reserved
		7	AI value
		8	Reserved
		9	Reserved
		10	Reserved
		11	Reserved
		12	Reserved
		13	Reserved
		14	Reserved
		15	Reserved
		16	Reserved
		17	Reserved
		18	Reserved
		19	Reserved
		20	Reserved
		21	Reserved
		22	Reserved
		23	Reserved
		30	Expansion bit data
		31	Expansion bit error code
		32	Expansion word data
		33	Expansion word error code

If you wish to view the data for individual DI channels using the RESTful API, you need to append "/ch_n" to the API URL, where "n" represents the channel number. In the context of the example provided (Figure 4-411), to observe the data for the first DI channel (channel 0), you should input the following URL into the address bar: "10.12.1.228/di_value/slot_0/ch_0". This action will allow you to access the returned data specifically for the first DI channel.

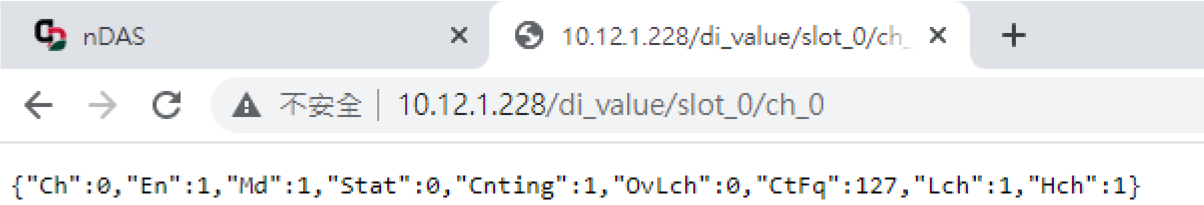


Figure 4-411 RESTful return data of DI channel

If you wish to retrieve data for all DI channels simultaneously using the RESTful API, you should exclude "/ch_n" from the API URL. Referring to the example from Figure 4-412, to access data for all DI channels, input the following URL into the address bar: "10.12.1.228/di_value/slot_0". This will grant you access to the data returned by all DI channels within the specified slot.

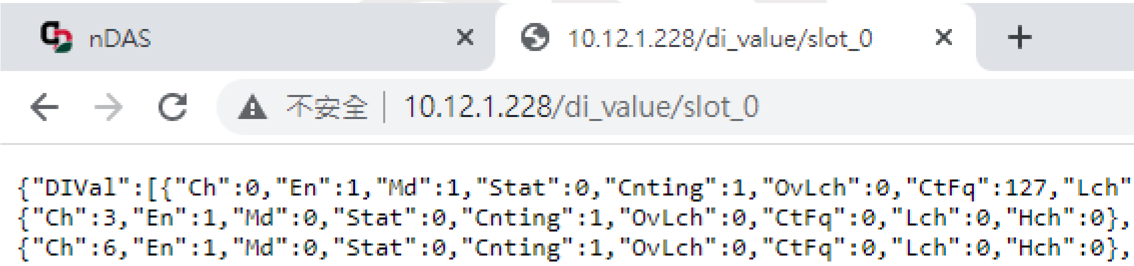


Figure 4-412 RESTful return all data of DI channel

If you intend to view data for various DO channels using the RESTful API, you need to append "/ch_n" to the API URL, with "n" representing the channel number. In the context of the example provided (Figure 4-413), to access data for the first DO channel (channel 0), you should enter the following URL into the address bar: "10.12.1.228/do_value/slot_0/ch_0". This will allow you to view the returned data specifically for the first DO channel.

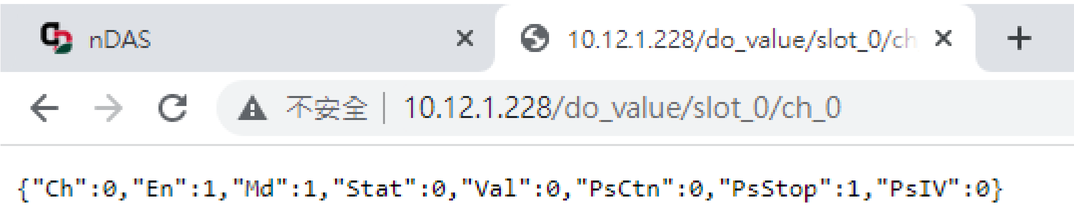


Figure 4-413 RESTful return data of DO channel

To retrieve data for all DO channels at once using the RESTful API, you should omit "/ch_n" from the API URL. In reference to the example shown in Figure 4-414, to access data for all DO channels, input the following URL into the address bar: "10.12.1.228/do_value/slot_0". This will allow you to view the data returned by all DO channels within the specified slot.

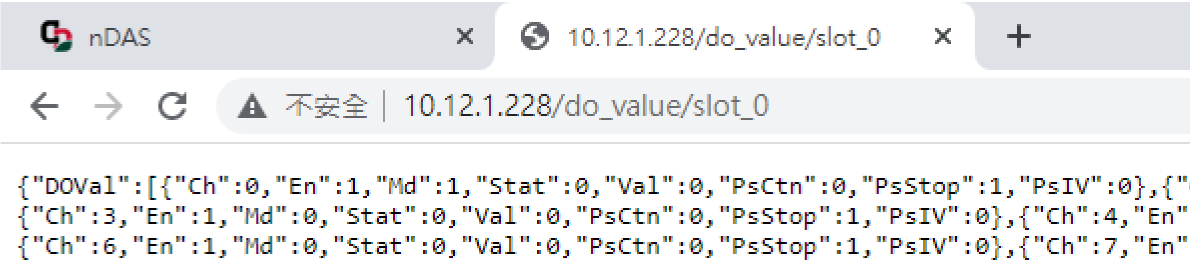


Figure 4-414 RESTful return all data of DO channel

To view data for individual AI channels using the RESTful API, you should add "/ch_n" to the API URL, where "n" represents the channel number. In the context of the example from Figure 4-415, to access data for the first AI channel (channel 0), you need to enter the following URL into the address bar: "10.12.1.53/ai_value/slot_0/ch_0". This will enable you to observe the data returned specifically for the first AI channel.

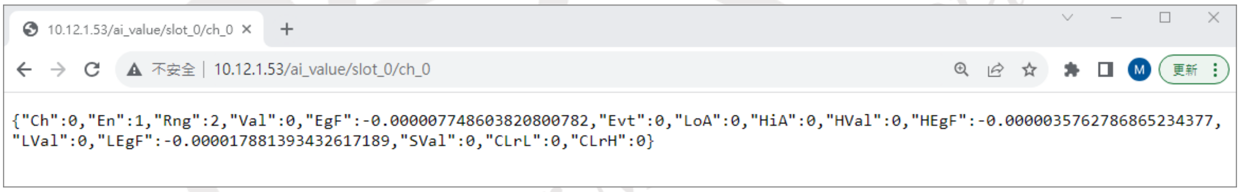


Figure 4-415 RESTful return data of AI channel

If you want to retrieve data for all AI channels simultaneously using the RESTful API, you should exclude "/ch_n" from the API URL. In reference to the example depicted in Figure 4-416, to access data for all AI channels, input the following URL into the address bar: "10.12.1.53/ai_value/slot_0". This will enable you to observe the data returned by all AI channels within the specified slot.

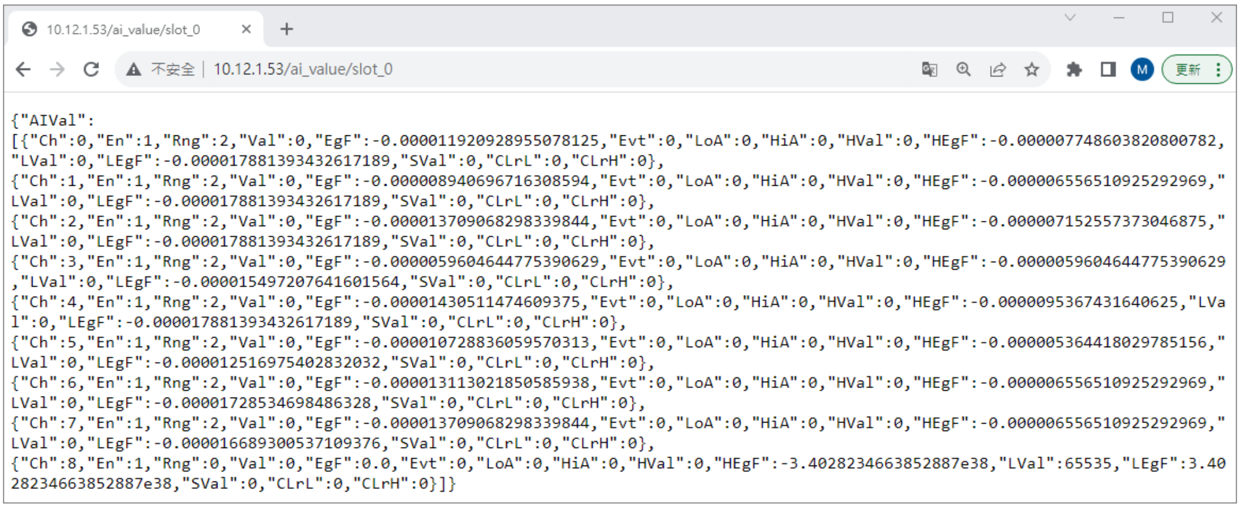


Figure 4-416 RESTful return all data of AI channel

To view data for individual Modbus TCP/RTU expansion bit channels using the RESTful API, you should add "/ch_n" to the API URL, where "n" represents the channel number. In the context of the example shown in Figure 4-417, to access data for the first Modbus RTU COM2 expansion bit channel (channel 0), you need to input the following URL into the address bar: "10.12.1.53/expansion_bit/com_2/ch_0". This will allow you to observe the data returned specifically for the first Modbus TCP/RTU expansion bit channel.

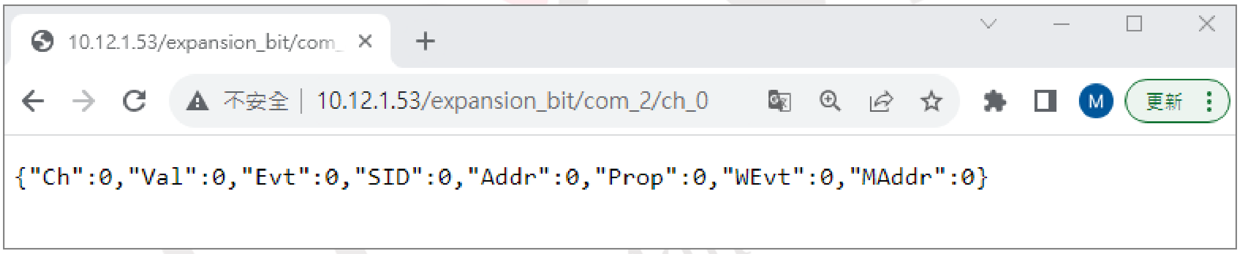


Figure 4-417 RESTful return data of Modbus TCP/RTU expansion bit channel

If you wish to retrieve data for all Modbus TCP/RTU expansion bit channels at once using the RESTful API, you should exclude "/ch_n" from the API URL. Referring to the example in Figure 4-418, to access data for all Modbus RTU COM2 expansion bit channels, input the following URL into the address bar: "10.12.1.53/expansion_bit/com_2". This will allow you to observe the data returned by all Modbus TCP/RTU expansion bit channels within the specified COM2 slot.

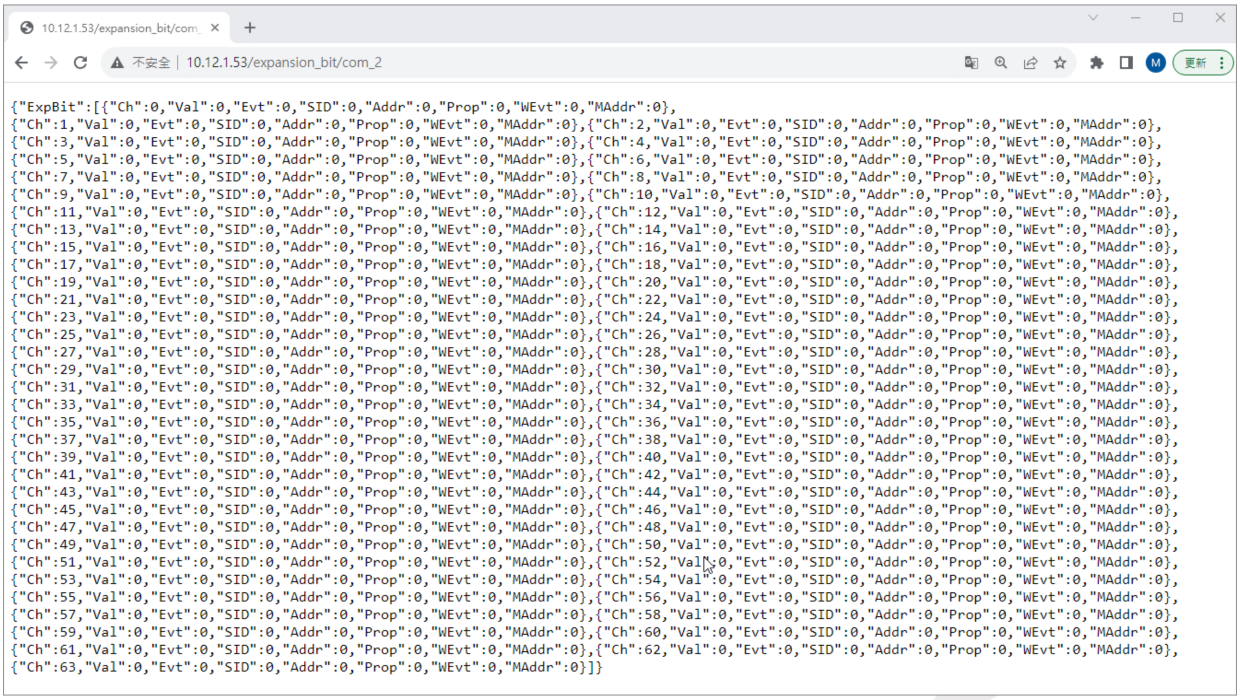


Figure 4-418 RESTful return all data of Modbus TCP/RTU expansion bit channel

Important Note: You can replace "com" with "tcp". Please note that you need to add a Modbus TCP Master first for this to take effect. Please refer to section 4.1.2.1 for detailed instructions.

Important Note: The value of "m" in "com_m" should be determined according to the Port numbers included in nDAS, as indicated in section 4.1.2.1.

To view data for individual Modbus TCP/RTU expansion word channels using the RESTful API, you should add "/ch_n" to the API URL, where "n" represents the channel number. In the context of the example from Figure 4-419, to access data for the first Modbus RTU COM2 expansion word channel (channel 0), you need to input the following URL into the address bar: "10.12.1.53/expansion_word/com_2/ch_0". This will allow you to observe the data returned specifically for the first Modbus TCP/RTU expansion word channel.

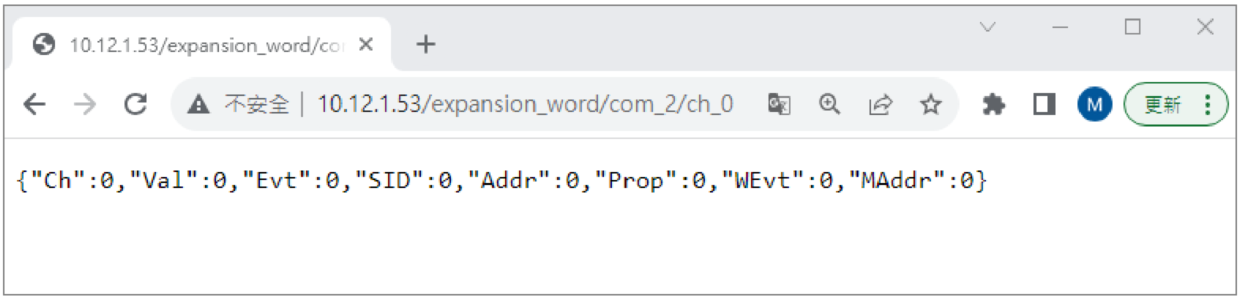


Figure 4-419 RESTful return data of Modbus TCP/RTU expansion word channel

To retrieve data for all Modbus TCP/RTU expansion word channels at once using the RESTful API, you should exclude "/ch_n" from the API URL. Referring to the example in Figure 4-420, to access data for all Modbus RTU COM2 expansion word channels, input the following URL into the address bar: "10.12.1.53/expansion_word/com_2". This will allow you to observe the data returned by all Modbus TCP/RTU expansion word channels within the specified COM2 slot.

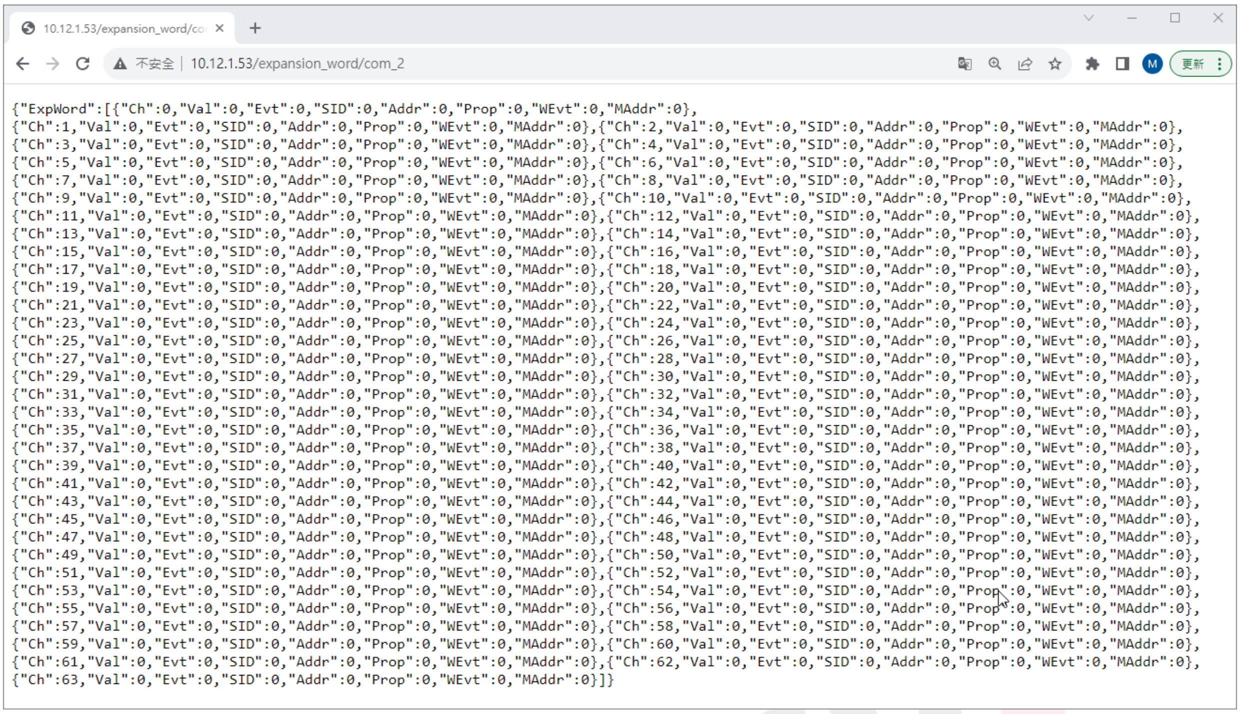


Figure 4-420 RESTful return all data of Modbus TCP/RTU expansion word channel

Important Note: You can replace "com" with "tcp". Please note that you need to add a Modbus TCP Master first for this to take effect. Please refer to section 4.1.2.1 for detailed instructions.

Important Note: The value of "m" in "com_m" should be determined according to the Port numbers included in nDAS, as indicated in section 4.1.2.1.

To view data for individual CALC channels using the RESTful API, you should add "/ch_n" to the API URL, where "n" represents the channel number. In the context of the example from Figure 4-421, to access data for the first CALC channel (channel 0), you need to input the following URL into the address bar: "10.12.1.219/calc_value/ch_0". This will allow you to observe the data returned specifically for the first CALC channel.

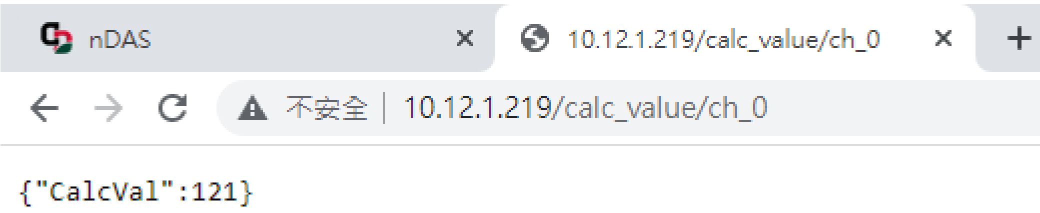


Figure 4-421 RESTful return data of CALC channel

The RESTful data structure of this software is delineated as follows, elaborating on the distinct RESTful requests for DI, DO, AI, and more.

4.5.1.1 Digital Input

Description	Retrieves information about the digital input value resource on specific slot.
URL Structure	http://ip/di_value/slot_index http://ip/di_value/slot_index/ch_num
HTTP Method	GET: Returns the representation of all of digital input value resource. PUT: Replace all of digital input value resource. PATCH: Apply partial modifications to digital input value resource.

GET	Multiple channel Request: **GET ip/di_value/slot_index** Single channel Request: **GET ip/di_value/slot_index/ch_num** [Example] Request: **GET 10.12.1.159/di_value/slot_0** Content-type: application/json;charset=utf-8 Response: 200 OK { "DIVal": [{ "Ch": 0, "En": 1, "Md": 0, "Stat": 1, "Cnting": 1, "OvLch": 0, "CtFq": 0, "Lch": 0, "Hch": 0 }, {

```
        "Ch": 1,
        "En": 1,
        "Md": 0,
        "Stat": 1,
        "Cnting": 1,
        "OvLch": 0,
        "CtFq": 0,
        "Lch": 0,
        "Hch": 0
    },
    {
        "Ch": 2,
        "En": 1,
        "Md": 0,
        "Stat": 1,
        "Cnting": 1,
        "OvLch": 0,
        "CtFq": 0,
        "Lch": 0,
        "Hch": 0
    },
    {
        "Ch": 3,
        "En": 1,
        "Md": 0,
        "Stat": 1,
        "Cnting": 1,
        "OvLch": 0,
        "CtFq": 0,
        "Lch": 0,
        "Hch": 0
    },
    {
        "Ch": 4,
        "En": 1,
        "Md": 0,
```

"Stat": 1,
"Cnting": 1,
"OvLch": 0,
"CtFq": 0,
"Lch": 0,
"Hch": 0
},
{
"Ch": 5,
"En": 1,
"Md": 0,
"Stat": 1,
"Cnting": 1,
"OvLch": 0,
"CtFq": 0,
"Lch": 0,
"Hch": 0
},
{
"Ch": 6,
"En": 1,
"Md": 0,
"Stat": 1,
"Cnting": 1,
"OvLch": 0,
"CtFq": 0,
"Lch": 0,
"Hch": 0
},
{
"Ch": 7,
"En": 1,
"Md": 0,
"Stat": 1,
"Cnting": 1,
"OvLch": 0,

	"CtFq": 0,
	"Lch": 0,
	"Hch": 0
	}
	]
	}
	
	Request: **GET 10.12.1.159/di_value/slot_0/ch_0**
	Content-type: application/json;charset=utf-8
	Response: 200 OK
	{
	"Ch": 0,
	"En": 1,
	"Md": 0,
	"Stat": 1,
	"Cnting": 1,
	"OvLch": 0,
	"CtFq": 0,
	"Lch": 0,
	"Hch": 0
	}

PUT	Multiple channel Request: **PUT ip/di_value/slot_index** Single channel Request: **PUT ip/di_value/slot_index/ch_num** [Example] Request: **PUT 10.12.1.159/di_value/slot_0** Content-type: application/json;charset=utf-8 Body: { "DIVal":[

{
  "Ch": 0,
  "En": 1,
  "Md": 0,
  "Stat": 0,
  "Cnting": 1,
  "OvLch": 0,
  "CtFq": 0,
  "Lch": 0,
  "Hch": 0
},
{
  "Ch": 1,
  "En": 1,
  "Md": 0,
  "Stat": 0,
  "Cnting": 1,
  "OvLch": 0,
  "CtFq": 0,
  "Lch": 0,
  "Hch": 0
},
{
  "Ch": 2,
  "En": 1,
  "Md": 0,
  "Stat": 0,
  "Cnting": 1,
  "OvLch": 0,
  "CtFq": 0,
  "Lch": 0,
  "Hch": 0
},
{
  "Ch": 3,
  "En": 1,

"Md": 0,
"Stat": 0,
"Cnting": 1,
"OvLch": 0,
"CtFq": 0,
"Lch": 0,
"Hch": 0
},
{
"Ch": 4,
"En": 1,
"Md": 0,
"Stat": 0,
"Cnting": 1,
"OvLch": 0,
"CtFq": 0,
"Lch": 0,
"Hch": 0
},
{
"Ch": 5,
"En": 1,
"Md": 0,
"Stat": 0,
"Cnting": 1,
"OvLch": 0,
"CtFq": 0,
"Lch": 0,
"Hch": 0
},
{
"Ch": 6,
"En": 1,
"Md": 0,
"Stat": 0,
"Cnting": 1,

"OvLch": 0,
"CtFq": 0,
"Lch": 0,
"Hch": 0
},
{
"Ch": 7,
"En": 1,
"Md": 0,
"Stat": 0,
"Cnting": 1,
"OvLch": 0,
"CtFq": 0,
"Lch": 0,
"Hch": 0
}
]
}

Response: 200 OK

	Request: **PUT 10.12.1.159/di_value/slot_0/ch_0**
	Content-type: application/json;charset=utf-8
	Body:
	{
	"Ch": 0,
	"En": 1,
	"Md": 0,
	"Stat": 0,
	"Cnting": 1,
	"OvLch": 0,
	"CtFq": 0,
	"Lch": 0,
	"Hch": 0
	}
	Response: 200 OK

PATCH	Multiple channel Request: **PATCH ip/di_value/slot_index** Single channel Request: **PATCH ip/di_value/slot_index/ch_num** [Example] Request: **PATCH 10.12.1.159/di_value/slot_0** Content-type: application/json;charset=utf-8 Body: { "DIVal": [{ "Ch":0, "Cnting": 1 }, { "Ch":3,

```
        "OvLch":0
        }
    ]
 }
```

Response: 200 OK

Request: **PATCH 10.12.1.159/di_value/slot_0/ch_3**

Content-type: application/json;charset=utf-8

Body:

```
{
    "Ch":3,
    "Cnting": 1
}
```

Response: 200 OK

4.5.1.2 Digital Output

Description	Retrieves information about the digital output value resource on specific slot.
URL Structure	http://ip/do_value/slot_index http://ip/do_value/slot_index/ch_num
HTTP Method	GET: Returns the representation of all of digital output value resource. PUT: Replace all of digital output value resource. PATCH: Apply partial modifications to digital output value resource.

GET	Multiple channel Request: **GET ip/do_value/slot_index** Single channel Request: **GET ip/do_value/slot_index/ch_num** [Example] Request: **GET 10.12.1.159/do_value/slot_0** Content-type: application/json;charset=utf-8 Response: 200 OK { "DOVal": [{ "Ch": 0, "En": 1, "Md": 1, "Stat": 0, "Val": 0, "PsCtn": 1, "PsStop": 0, "PsIV": 0 }, { "Ch": 1,

```
        "En": 1,
        "Md": 0,
        "Stat": 0,
        "Val": 0,
        "PsCtn": 0,
        "PsStop": 1,
        "PsIV": 0
    },
    {
        "Ch": 2,
        "En": 1,
        "Md": 0,
        "Stat": 0,
        "Val": 0,
        "PsCtn": 0,
        "PsStop": 1,
        "PsIV": 0
    },
    {
        "Ch": 3,
        "En": 1,
        "Md": 0,
        "Stat": 0,
        "Val": 0,
        "PsCtn": 0,
        "PsStop": 1,
        "PsIV": 0
    },
    {
        "Ch": 4,
        "En": 1,
        "Md": 0,
        "Stat": 0,
        "Val": 0,
        "PsCtn": 0,
        "PsStop": 1,
```

```
                "PsIV": 0
            },
            {
                "Ch": 5,
                "En": 1,
                "Md": 0,
                "Stat": 0,
                "Val": 0,
                "PsCtn": 0,
                "PsStop": 1,
                "PsIV": 0
            },
            {
                "Ch": 6,
                "En": 1,
                "Md": 0,
                "Stat": 0,
                "Val": 0,
                "PsCtn": 0,
                "PsStop": 1,
                "PsIV": 0
            },
            {
                "Ch": 7,
                "En": 1,
                "Md": 0,
                "Stat": 0,
                "Val": 0,
                "PsCtn": 0,
                "PsStop": 1,
                "PsIV": 0
            }
        ]
    }
```

	Request: **GET 10.12.1.159/do_value/slot_0/ch_0**
	Content-type: application/json;charset=utf-8
	Response: 200 OK
	{
	"Ch": 0,
	"En": 1,
	"Md": 1,
	"Stat": 0,
	"Val": 4294967295,
	"PsCtn": 1,
	"PsStop": 0,
	"PsIV": 0
	}

PUT	Multiple channel Request: **PUT ip/do_value/slot_index** Single channel Request: **PUT ip/do_value/slot_index/ch_num** [Example] Request: **PUT 10.12.1.159/do_value/slot_0** Content-type: application/json;charset=utf-8 Body: { "DOVal":[{ "Ch":0, "En":0, "Md":0, "Stat":0, "Val":0,

"PsCtn":0,
"PsStop":0,
"PsIV":0
},
{
"Ch":1,
"En":0,
"Md":0,
"Stat":0,
"Val":0,
"PsCtn":0,
"PsStop":0,
"PsIV":0
},
{
"Ch":2,
"En":0,
"Md":0,
"Stat":0,
"Val":0,
"PsCtn":0,
"PsStop":0,
"PsIV":0
},
{
"Ch":3,
"En":0,
"Md":0,
"Stat":0,
"Val":0,
"PsCtn":0,
"PsStop":0,
"PsIV":0
},
{
"Ch":4,

"En":0,
"Md":0,
"Stat":0,
"Val":0,
"PsCtn":0,
"PsStop":0,
"PsIV":0
},
{
"Ch":5,
"En":0,
"Md":0,
"Stat":0,
"Val":0,
"PsCtn":0,
"PsStop":0,
"PsIV":0
},
{
"Ch":6,
"En":0,
"Md":0,
"Stat":0,
"Val":0,
"PsCtn":0,
"PsStop":0,
"PsIV":0
},
{
"Ch":7,
"En":0,
"Md":0,
"Stat":0,
"Val":0,
"PsCtn":0,
"PsStop":0,

	"PsIV":0
	}
	]
	}
	Response: 200 OK
	
	Request: **PUT 10.12.1.159/do_value/slot_0/ch_0**
	Content-type: application/json;charset=utf-8
	Body:
	{
	"Ch":0,
	"En":0,
	"Md":0,
	"Stat":0,
	"Val":0,
	"PsCtn":0,
	"PsStop":0,
	"PsIV":0
	}
	Response: 200 OK

PATCH	Multiple channel Request: **PATCH ip/do_value/slot_index** Single channel Request: **PATCH ip/do_value/slot_index/ch_num** [Example] Request: **PATCH 10.12.1.159/do_value/slot_0** Content-type: application/json;charset=utf-8 Body: { "DOVal": [{

	"Ch":2,
	"Md": 2
	},
	{
	"Ch":3,
	"PsStop":1
	}
	]
	}
	Response: 200 OK
	
	Request: **PATCH 10.12.1.159/do_value/slot_0/ch_3**
	Content-type: application/json;charset=utf-8
	Body:
	{
	"Ch":3,
	"PsCtn": 1
	}
	Response: 200 OK

4.5.1.3 Analog Input

Description	Retrieves information about the analog input value resource on specific slot.
URL Structure	http://ip/ai_value/slot_index http://ip/ai_value/slot_index/ch_num
HTTP Method	GET: Returns the representation of all of analog input value resource. PUT:None PATCH: Apply partial modifications to analog input value resource.

GET	Multiple channel Request: **GET ip/ai_value/slot_index** Single channel Request: **GET ip/ai_value/slot_index/ch_num** [Example] Request: **GET 10.12.1.159/ai_value/slot_0** Content-type: application/json;charset=utf-8 Response: 200 OK { "AIVal": [{ "Ch": 0, "En": 1, "Rng": 2, "Val": 0, "EgF": -0.0000881092071533203, "Evt": 0, "LoA": 0, "HiA": 0, "HVal": 0, "HEgF": -0.0000816583633422851 6, "LVal": 0, "LEgF": -0.0000983476638793945 3, "SVal": 0, "CLrL": 0, "CLrH": 0 }, { "Ch": 1, "En": 1, "Rng": 2, "Val": 0,

"EgF": -0.00009059906005859375,
"Evt": 0,
"LoA": 0,
"HiA": 0,
"HVal": 0,
"HEgF": -0.00008046627044677734,
"LVal": 0,
"LEgF": -0.00009715557098388672,
"SVal": 0,
"CLrL": 0,
"CLrH": 0
},
{
"Ch": 2,
"En": 1,
"Rng": 2,
"Val": 0,
"EgF": -0.00008404254913330078,
"Evt": 0,
"LoA": 0,
"HiA": 0,
"HVal": 0,
"HEgF": -0.00008165836334228516,
"LVal": 0,
"LEgF": -0.00009775161743164063,
"SVal": 0,
"CLrL": 0,
"CLrH": 0
},
{
"Ch": 3,
"En": 1,
"Rng": 2,
"Val": 0,
"EgF": -0.00008940696716308594,
"Evt": 0,

```
            "LoA": 0,
            "HiA": 0,
            "HVal": 0,
            "HEgF": -0.000081658363342285156,
            "LVal": 0,
            "LEgF": -0.00009655952453613281,
            "SVal": 0,
            "CLrL": 0,
            "CLrH": 0
        },
        {
            "Ch": 4,
            "En": 1,
            "Rng": 2,
            "Val": 0,
            "EgF": -0.000092387199401855547,
            "Evt": 0,
            "LoA": 0,
            "HiA": 0,
            "HVal": 0,
            "HEgF": -0.000084042549133300078,
            "LVal": 0,
            "LEgF": -0.000099539756774902344,
            "SVal": 0,
            "CLrL": 0,
            "CLrH": 0
        },
        {
            "Ch": 5,
            "En": 1,
            "Rng": 2,
            "Val": 0,
            "EgF": -0.000090003013610839844,
            "Evt": 0,
            "LoA": 0,
            "HiA": 0,
```

```
        "HVal": 0,
        "HEgF": -0.00007987022399902344,
        "LVal": 0,
        "LEgF": -0.00010132789611816406,
        "SVal": 0,
        "CLrL": 0,
        "CLrH": 0
    },
    {
        "Ch": 6,
        "En": 1,
        "Rng": 2,
        "Val": 0,
        "EgF": -0.0000864267349243164,
        "Evt": 0,
        "LoA": 0,
        "HiA": 0,
        "HVal": 0,
        "HEgF": -0.00008165836334228516,
        "LVal": 0,
        "LEgF": -0.0000977516174316406З,
        "SVal": 0,
        "CLrL": 0,
        "CLrH": 0
    },
    {
        "Ch": 7,
        "En": 1,
        "Rng": 2,
        "Val": 0,
        "EgF": -0.00009059906005859375,
        "Evt": 0,
        "LoA": 0,
        "HiA": 0,
        "HVal": 0,
        "HEgF": -0.00007867813110351563,
```

```
            "LVal": 0,
            "LEgF": -0.00009775161743164063,
            "SVal": 0,
            "CLrL": 0,
            "CLrH": 0
        },
        {
            "Ch": 8,
            "En": 1,
            "Rng": 0,
            "Val": 0,
            "EgF": 0.0,
            "Evt": 0,
            "LoA": 0,
            "HiA": 0,
            "HVal": 0,
            "HEgF": -3.4028234663852887e38,
            "LVal": 65535,
            "LEgF": 3.4028234663852887e38,
            "SVal": 0,
            "CLrL": 0,
            "CLrH": 0
        }
    ]
}
```

	Request: **GET 10.12.1.159/ai_value/slot_0/ch_0**
	Content-type: application/json;charset=utf-8
	Response: 200 OK
	{
	"Ch": 0,
	"En": 1,
	"Rng": 2,
	"Val": 0,
	"EgF": -0.00008761882781982422,
	"Evt": 0,
	"LoA": 0,
	"HiA": 0,
	"HVal": 0,
	"HEgF": -0.00008165836334228516,
	"LVal": 0,
	"LEgF": -0.00009834766387939453,
	"SVal": 0,
	"CLrL": 0,
	"CLrH": 0
	}

PATCH	Multiple channel Request: **PATCH ip/ai_value/slot_index** Single channel Request: **PATCH ip/ai_value/slot_index/ch_num** [Example] Request: **PATCH 10.12.1.159/ai_value/slot_0** Content-type: application/json;charset=utf-8 Body: { "AIVal": [{ "Ch":2, "LoA": 2

	},
	 {
	 "Ch":3,
	 "HiA":1
	 }
	]
	 }
	 Response: 200 OK
	
	 Request: **PATCH 10.12.1.159/ai_value/slot_0/ch_3**
	 Content-type: application/json;charset=utf-8
	 Body:
	 {
	 "LoA": 1
	 }
	 Response: 200 OK

4.5.1.4 Modbus Expansion Bit Data(Modbus RTU/TCP)

Description	Retrieves information about the expansion tag bit data resource, the data information is defined by user configuration.
URL Structure	http://ip/expansion_bit/com_x http://ip/expansion_bit/tcp_x http://ip/expansion_bit/com_x/ch_num http://ip/expansion_bit/tcp_x/ch_num
HTTP Method	GET: Returns the representation of all of expansion bit data resource. PUT: None PATCH: Apply partial modifications to expansion bit data resource.

GET	Multiple channel Request: **GET ip/expansion_bit/com_x** Single channel Request: **GET ip/expansion_bit/com_x/ch_num** [Example] Request: **GET 10.12.1.159/expansion_bit/com_2** Content-type: application/json;charset=utf-8 Response: 200 OK { "ExpBit": [{ "Ch": 0, "Val": 0, "Evt": 0, "SID": 0, "Addr": 0, "Prop": 0, "WEvt": 0, "MAddr": 0 }, { "Ch": 1, "Val": 0, "Evt": 0, "SID": 0, "Addr": 0, "Prop": 0, "WEvt": 0, "MAddr": 0 }, ... { "Ch": 127,

	"Val": 0,
	"Evt": 0,
	"SID": 0,
	"Addr": 0,
	"Prop": 0,
	"WEvt": 0,
	"MAddr": 0
	}
	]
	}
	
	Request: **GET 10.12.1.159/expansion_bit/com_2/ch_0**
	Content-type: application/json;charset=utf-8
	Response: 200 OK
	{
	"Ch": 0,
	"Val": 0,
	"Evt": 0,
	"SID": 0,
	"Addr": 0,
	"Prop": 0,
	"WEvt": 0,
	"MAddr": 0
	}

PATCH	Single channel Request: **PATCH ip/expansion_bit/com_x/ch_num** [Example]

	Request: **PATCH 10.12.1.159/expansion_bit/com_2/ch_0**
	Content-type: application/json;charset=utf-8
	Body:
	{
	"Val": 1
	}
	Response: 200 OK

4.5.1.5 Modbus Expansion Word Data(Modbus RTU/TCP)

Request: /expansion_word/com_x/ch_num

Request: /expansion_word/tcp_x/ch_num

Description	Retrieves information about the expansion tag word data resource, the data information is defined by user configuration.
URL Structure	http://ip/expansion_word/com_x http://ip/expansion_word/tcp_x http://ip/expansion_word/com_x/ch_num http://ip/expansion_word/tcp_x/ch_num
HTTP Method	GET: Returns the representation of all of expansion word data resource. PUT: None PATCH: Apply partial modifications to expansion word data resource.

GET	Multiple channel Request: **GET ip/expansion_word/com_x** Single channel Request: **GET ip/expansion_word/com_x/ch_num** [Example] Request: **GET 10.12.1.159/expansion_word/com_2** Content-type: application/json;charset=utf-8 Response: 200 OK { "ExpWord": [{

```
                "Ch": 0,
                "Val": 0,
                "Evt": 0,
                "SID": 0,
                "Addr": 0,
                "Prop": 0,
                "WEvt": 0,
                "MAddr": 0
            },
            {
                "Ch": 1,
                "Val": 0,
                "Evt": 0,
                "SID": 0,
                "Addr": 0,
                "Prop": 0,
                "WEvt": 0,
                "MAddr": 0
            },
            ...
            {
                "Ch": 127,
                "Val": 0,
                "Evt": 0,
                "SID": 0,
                "Addr": 0,
                "Prop": 0,
                "WEvt": 0,
                "MAddr": 0
            }
        ]
    }
```


Request: **GET 10.12.1.159/expansion_word/com_2/ch_0**

Content-type: application/json;charset=utf-8

Response: 200 OK

	{
	 "Ch": 0,
	 "Val": 0,
	 "Evt": 0,
	 "SID": 0,
	 "Addr": 0,
	 "Prop": 0,
	 "WEvt": 0,
	 "MAddr": 0
	 }

PATCH	Single channel Request:
	 PATCH ip/expansion_word/com_x/ch_num
	
	 [Example]
	
	 Request: **PATCH 10.12.1.159/expansion_word/com_2/ch_0**
	 Content-type: application/json;charset=utf-8
	 Body:
	 {
	 "Val": 255
	 }
	 Response: 200 OK

4.5.1.6 Calc Operator

Description	Retrieves information about the Calc operator value resource.
URL Structure	http://ip/calc_value/ch_num
HTTP Method	GET: Returns the representation of all of Calc operator value resource. PUT: Replace all of Calc operator value resource. PATCH: Apply partial modifications to Calc operator value resource.

GET	Multiple channel Request: **GET ip/calc_value** Single channel Request:

GET ip/calc_value/ch_num

[Example]

Request: **GET 10.12.1.159/calc_value**

Content-type: application/json;charset=utf-8

Body: {"Ch":[0,1,2,3,4]}

Response: 200 OK


```
{
    "CalcVal": [
        {
            "Ch": 0,
            "Val": 0
        },
        {
            "Ch": 1,
            "Val": 0
        },
        {
            "Ch": 2,
            "Val": 0
        },
        {
            "Ch": 3,
            "Val": 0
        },
        {
            "Ch": 4,
            "Val": 0
        }
    ]
}
```

	Request: **GET ip/calc_value/ch_0**
	Content-type: application/json;charset=utf-8
	Response: 200 OK
	{
	"CalcVal":0
	}

PUT	Multiple channel Request:
	PUT ip/calc_value
	Single channel Request:
	PUT ip/calc_value/ch_num
	
	[Example]
	
	Request: **PUT 10.12.1.159/calc_value**
	Content-type: application/json;charset=utf-8
	Body:
	{
	"CalcVal": [
	{
	"Ch": 0,
	"Val": 1.0
	},
	{
	"Ch": 1,
	"Val": 2.0
	}
	]
	}
	Response: 200 OK

	Request: **PUT 10.12.1.159/calc_value/ch_0**
	Content-type: application/json;charset=utf-8
	Body:
	{
	"Val":1.0
	}
	Response: 200 OK

PATCH	Multiple channel Request:
	PATCH ip/calc_value
	Single channel Request:
	PATCH ip/calc_value/ch_num
	
	[Example]
	
	Request: **PATCH 10.12.1.159/calc_value**
	Content-type: application/json;charset=utf-8
	Body:
	{
	"CalcVal": [
	{
	"Ch": 0,
	"Val": 1.0
	},
	{
	"Ch": 1,
	"Val": 2.0
	}
	]
	}
	Response: 200 OK

	Request: **PATCH 10.12.1.159/calc_value/ch_0**
	Content-type: application/json;charset=utf-8
	Body:
	{
	"Val":1.0
	}
	Response: 200 OK

4.5.1.7 Data Log

Request: /log_message

Description	Retrieves the log data in system memory.
URL Structure	http://ip/log_message
HTTP Method	GET: According to the setting of filtering, server returns the all/partial of logged data.

GET	**GET ip/log_message** [Example] Request: **GET ip/log_message** Content-type: application/json;charset=utf-8 Response: 200 OK { "LogMsg": [{ "MAC": "80-D2-1D-E8-31-C7", "PE": "128", "Record": [[0,0,7,-9.0003e-05], [0,0,11,0

```
                ],
                [
                    0,1,7,-9.0003e-05
                ],
                [
                    0,1,11,0]
            ],
            "TIM": "1688029844461",
            "UID": "nDAS6000_80D21DE831C7"
        },
        {
            "MAC": "80-D2-1D-E8-31-C7",
            "PE": "128",
            "Record": [
                [
                    0,0,7,-8.76188e-05
                ],
                [
                    0,0,11,0
                ],
                [
                    0,1,7,-8.88109e-05
                ],
                [
                    0,1,11,0
                ]
            ],
            "TIM": "1688029845932",
            "UID": "nDAS6000_80D21DE831C7"
        },
    ...
        {
            "MAC": "80-D2-1D-E8-31-C7",
            "PE": "128",
            "Record": [
                [
```

```
                    0,0,7,-9.05991e-05
                ],
                [
                    0,0,11,0
                ],
                [
                    0,1,7,-8.58307e-05
                ],
                [
                    0,1,11,0
                ]
            ],
            "TIM": "1688029872471",
            "UID": "nDAS6000_80D21DE831C7"
        }
    ]
}
```


4.5.2 RESTful Trigger Script and VIC Flow

This software provides software control through the RESTful API, offering two types of control actions: triggering a custom function using a Python script (req_call_python_function) and triggering an operator within VIC Flow (req_call_flow). The required parameters are detailed in Table 4-46.

Table 4-46 Parameters of RESTful

Parameter name	Content
cmd	In accordance with the provided control content input, the placeholder "A" is utilized.
func	The triggered function name is represented by the placeholder "B" in the following examples.
trigger_id	The ID of the operator to be triggered is represented by the placeholder "C" in the following examples.
params	The arguments passed into the triggered function are represented by the placeholder "D" in the following examples.

The data structure of RESTful:

```
Request:PUT IP/restful/put

Headers
Content-type:  application/x-www-form-urlencoded

Body
x-www-form-urlencoded
  {
     "cmd" : "A",
     "func" : "B",
     "trigger_id" : "C",
     "params" : "D"
  }
```

The example of trigger Python Script:

```
Request:PUT IP/restful/put

Headers
Content-type:  application/x-www-form-urlencoded

Body
x-www-form-urlencoded
  {
     "cmd" : "req_call_python_function",
     "func" : "python function definition name",
     "parames" : "parameters passed to function"
  }
```

The example of trigger VIC Flow:

```
Request:PUT IP/restful/put

Headers
Content-type:  application/x-www-form-urlencoded

Body
```

x-www-form-urlencoded

{

"cmd"："req_call_flow",

"trigger_id"："the ID of ON_RESTFUL operator",

" parames "："parameters passed to ON_RESTFUL operator"

}

4.6 Toolbar

The toolbar of this software is located in the upper right corner, as shown in Figure 4-422. The introduction of each buton will be presented in the following section.

Figure 4-422 Toolbar of this software

4.6.1 I/O Chart

You can switch to the Chart by clicking on the chart icon (as shown in Figure 4-423), where you can view the status of all open channels, as shown in Figure 4-424.

Figure 4-423 Click Chart icon

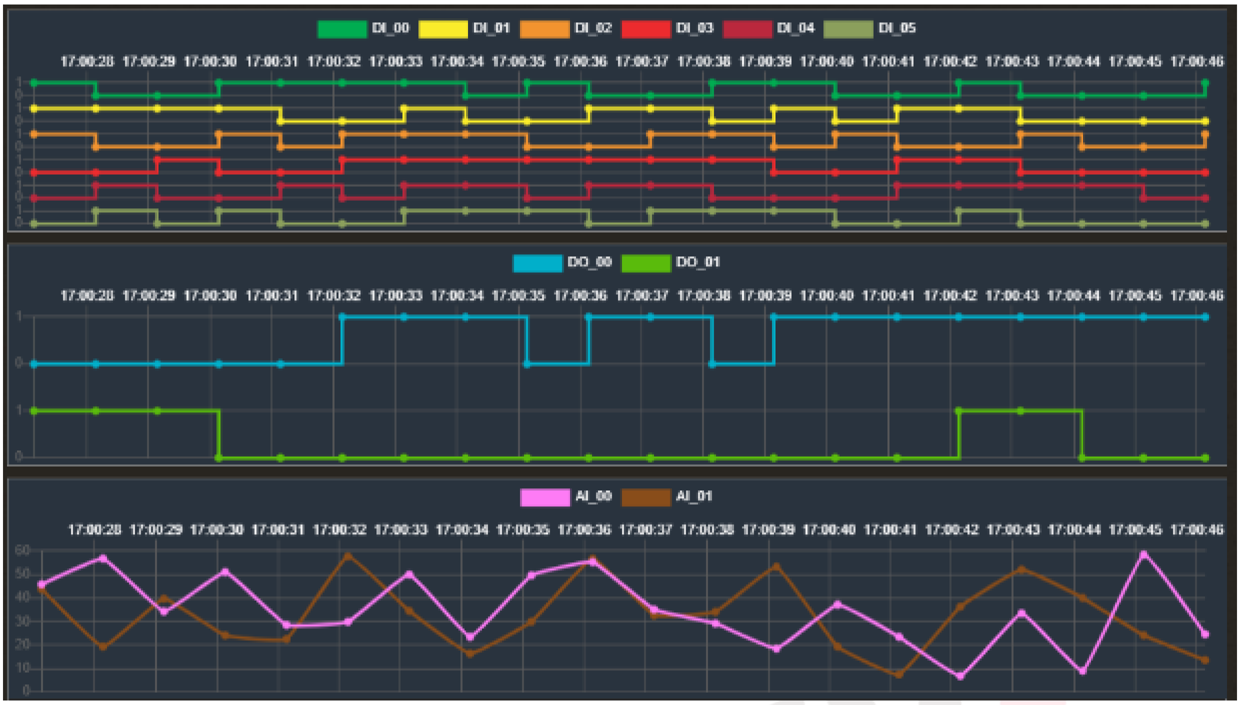


Figure 4-424 Chart

4.6.2 Modbus Chart

By clicking on the Modbus Channel icon (as shown in Figure 4-425), you can switch to the Modbus Channel Chart. After setting the Modbus Rule and checking the Chart option, corresponding drawing data will be displayed (for settings, please refer to section 4.1.2.4.1). Here, you can view the status of the Modbus Channel, as shown in Figure 4-426.

Figure 4-425 Click Modbus Channel icon

Note: A maximum of 9 channels can be displayed.

Figure 4-426 Modbus Channel Chart

4.6.3 Python Output

By clicking on the Python output icon (as shown in Figure 4-427), a Python dialog will pop up, where one can view the output information of the Python script, as shown in Figure 4-428.

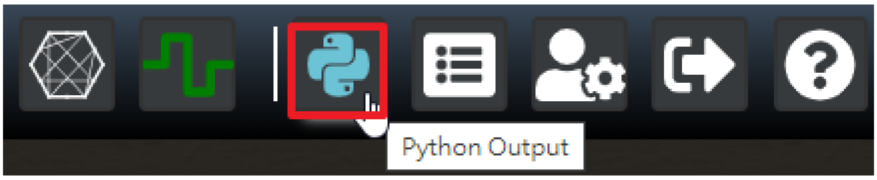


Figure 4-427 Click Python Output icon

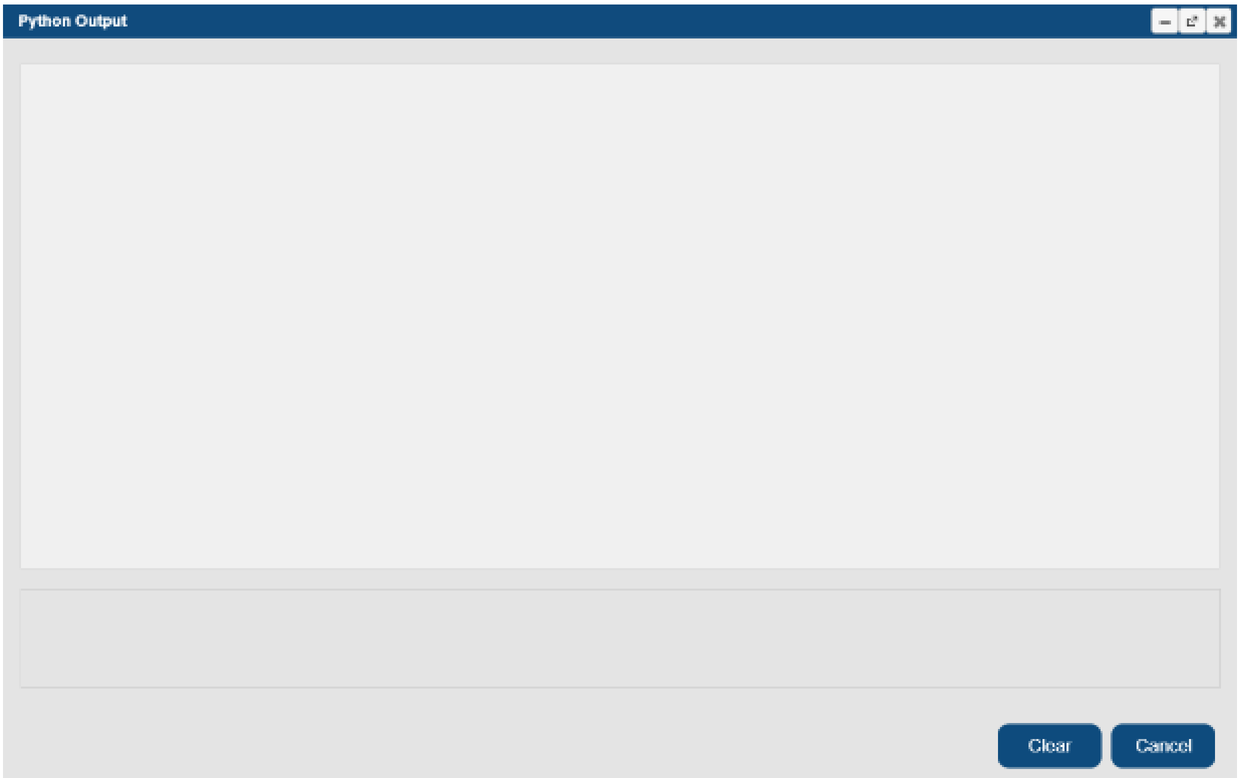


Figure 4-428 Python Output dialog

4.6.4 SECS/GEM Log

If the SECS/GEM connection is enabled, the toolbar will display the SECS/GEM log icon, as indicated by the red box in Figure 4-429. The “E” indicates passive mode, while “H” indicates active mode. Clicking the icon will display the SECS/GEM log dialog, as shown in Figure 4-430.

To write the log to a file, the “Use Log File” field must be checked, and the file will be stored according to rhe date.

Figure 4-429 SECS/GEM Log icon

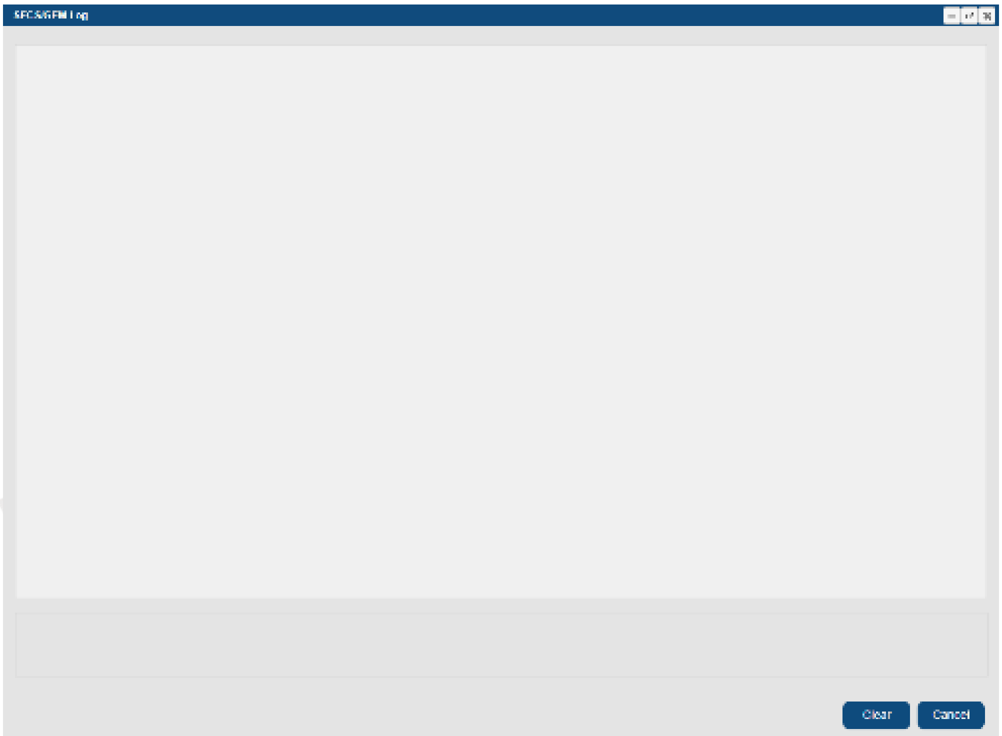


Figure 4-430 SECS/GEM Log dialog

Note：When lacking administrative privileges, it is not possible to browse SECS/GEM and SECS/GEM Host Logs.

Note：nDAS just support SECS/GEM Equipment

Moving the mouse cursor over the icon will display the current SECS/GEM status of the nDAS (as shown in Figure 4-431). The status descriptions are presented in Table 4-47.

Figure 4-431 Hover over the icon

Table 4-47 Corresponding SECS/GEM status table for the icon

SECS/GEM Status	Corresponding Icon	
	passive mode	active mode
DICONNECT	E	H
PORT_NOT_AVAILABLE		
CONNECT	E	H
SELECT		
WAIT_DELAY		
WAIT_CRA		
COMMUNICATING	E	H

4.6.5 System Log

This software includes a system log query function, which can record important software messages such as login/logout times and project loading times. Clicking on the system log icon (as shown in Figure 4-432) will display the system log dialog, as show in Figure 4-433. The system log can be exported as a txt file by clicking on the “Export” button in the dialog.

Note: The system log retains records for a maximum of 31 days, after which information beyond 31 days will be deleted.

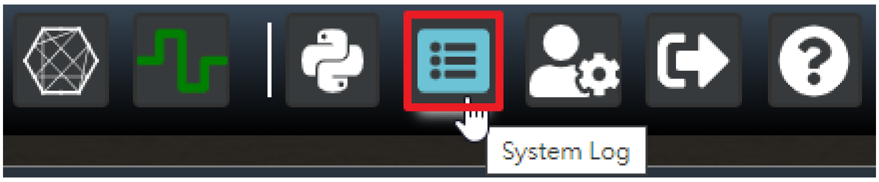


Figure 4-432 Click System Log icon

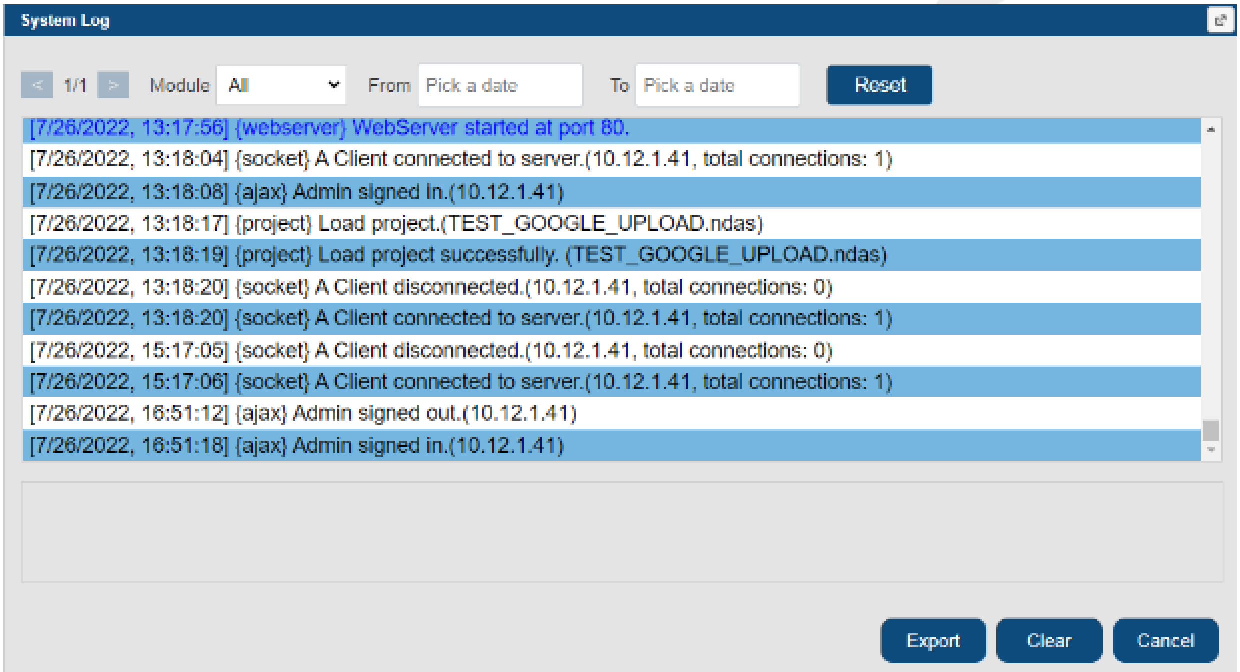


Figure 4-433 System Log dialog

4.6.6 Mode Switch

By clicking on the mode switch icon, it is possible to switch between user and administrator mode. In user mode (with administrative privileges), clicking on the mode switch (as shown in Figure 4-434) will switch to the administrator mode of the software, as shown in Figure 4-435. If you click on the mode switch in administrator mode (as shown in Figure 4-436), it will switch to the user mode (with administrative privileges) of the software, as shown in Figure 4-437.

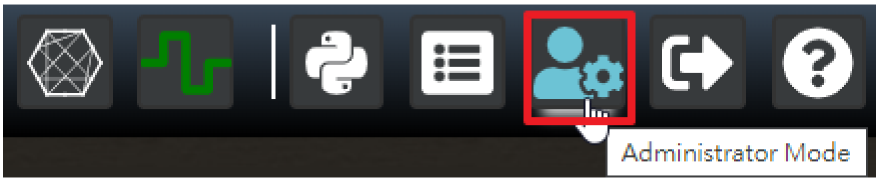


Figure 4-434 Clicking the mode switch in user mode (with administrative privileges)

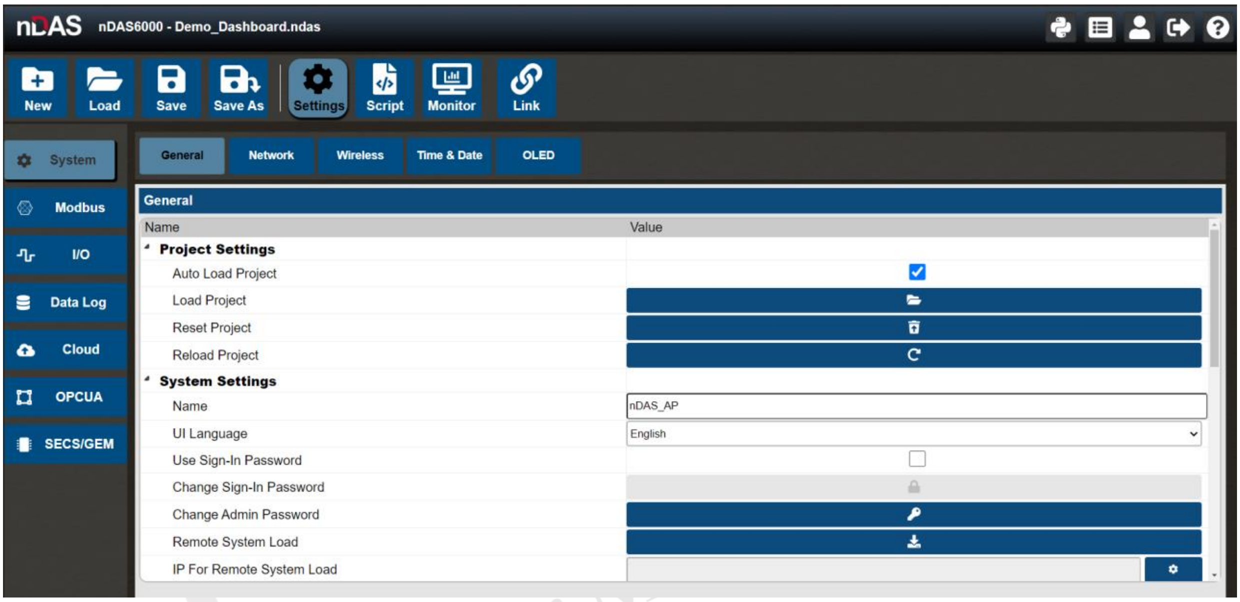


Figure 4-435 Administrator mode

Figure 4-436 Clicking the mode switch in administrator mode

Figure 4-437 User mode (with administrative privileges)

4.6.7 Login/Logout

Clicking on the login icon, as shown in Figure 4-438, will display the login dialog, as shown in Figure 4-439.

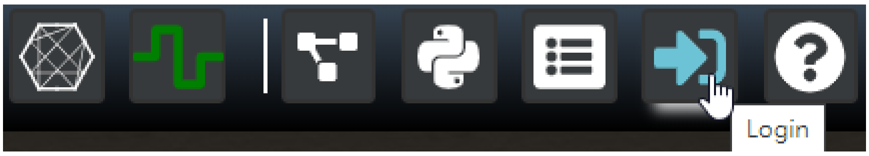


Figure 4-438 Click the login icon

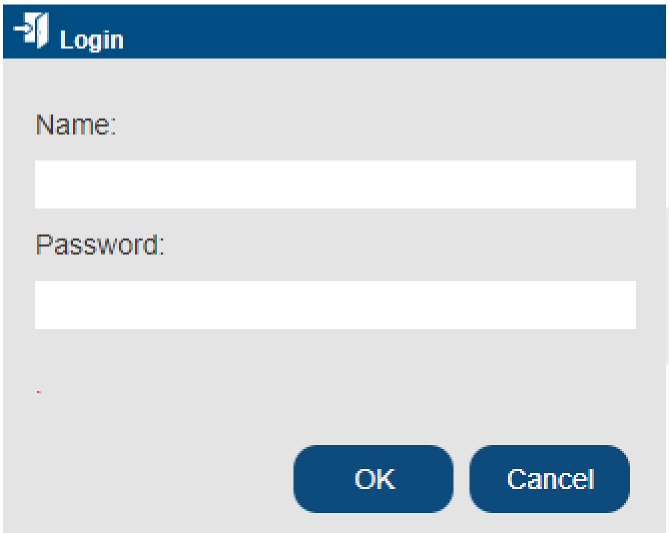


Figure 4-439 Login dialog

If the logout icon is clicked (as shown in Figure 4-440), a warning dialog (as shown in Figure 4-441) will first appear to confirm whether the user really wants to log out. After clicking “OK”, the user will be logged out.

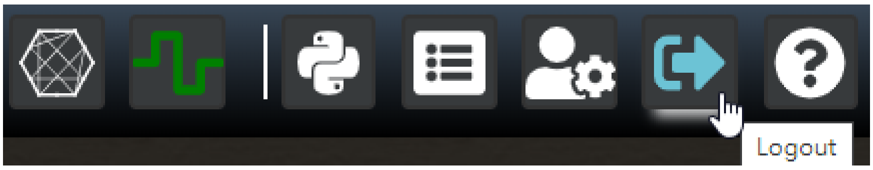


Figure 4-440 Click Logout Icon

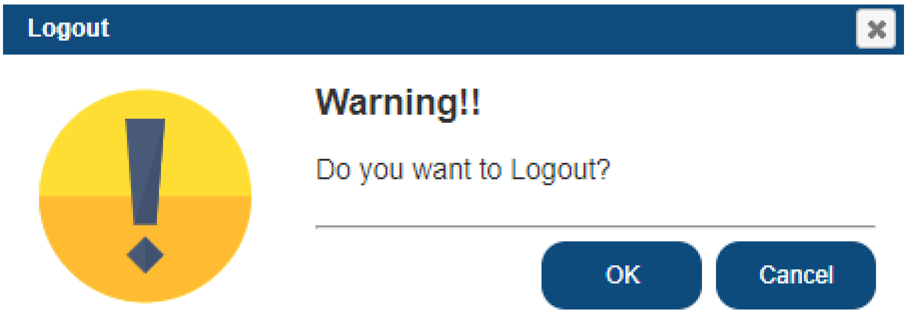


Figure 4-441 Logout Warning Dialog

4.6.8 About

Clicking on the About icon (as shown in Figure 4-442) will display the About dialog, which contains information such as the software version, user manual, software license, and nDAS storage information (as shown in Figure 4-443). The user manual will be displayed in the corresponding language based on the interface language, for example, the manual language corresponding to Japanese and English is English.

Figure 4-442 Click About Icon

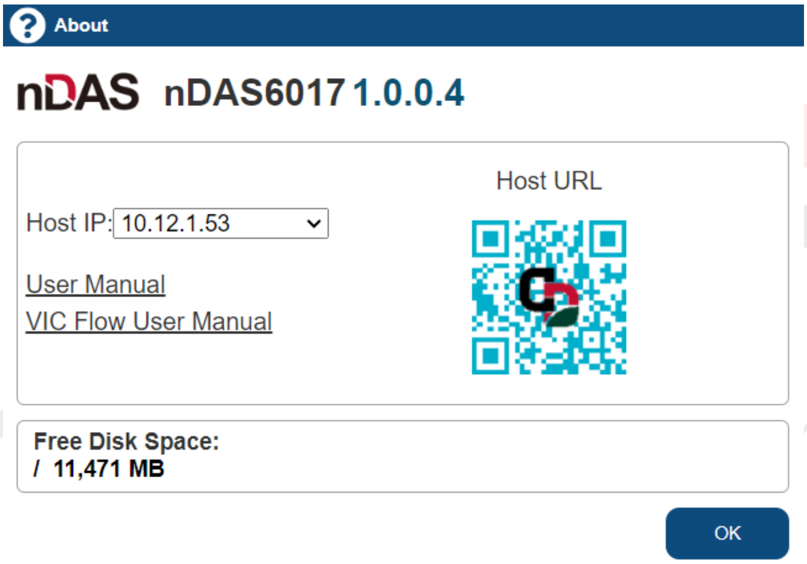
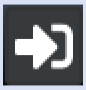
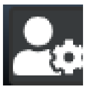
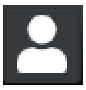
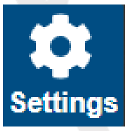
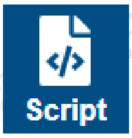
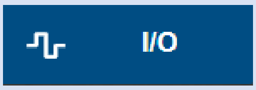


Figure 4-443 About Dialog

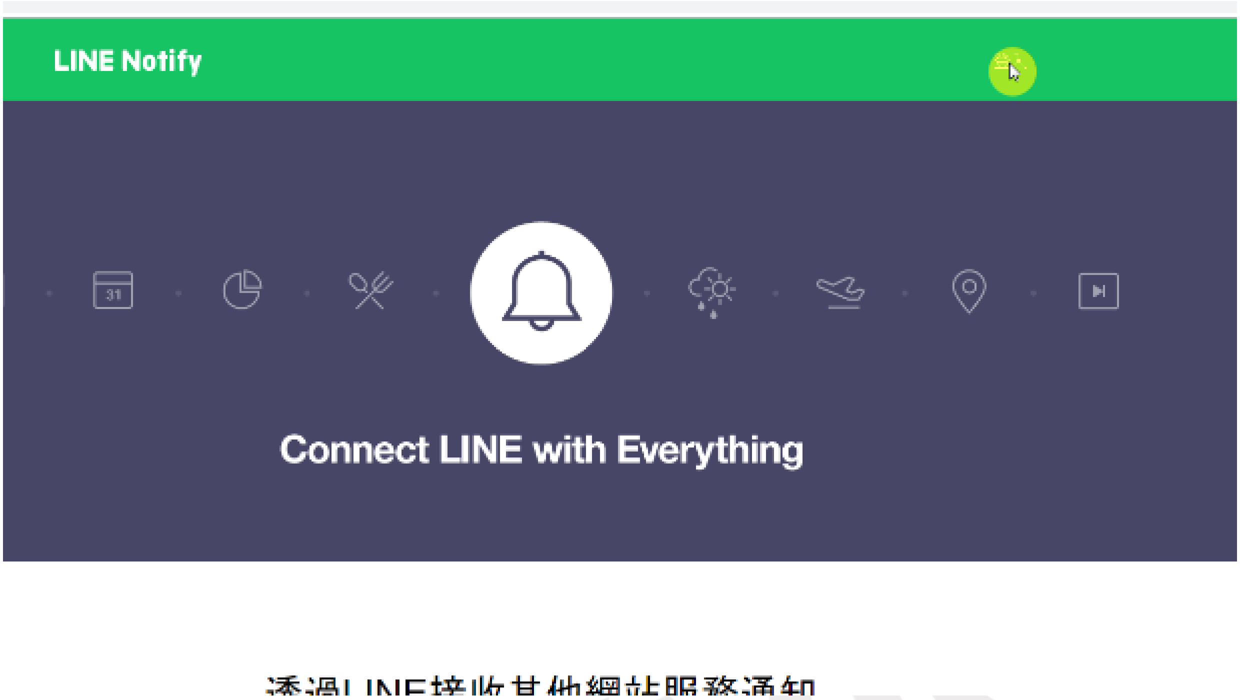
Chapter 5. Appendix

5.1 Software Icon Table

Icon	Introduction	Icon	Introduction
	Login		Logout
	Switch to Administrator Mode		Switch to User Mode
	System Log		Python Output
	I/O Chart		Modbus Chart
	About		
	New Project		Load Project
	Save Project		Save As Project
	Settings		Script
	Monitor		Link
	System Settings		Modbus
	I/O Settings		Database
	Cloud Upload		OPCUA Server
	SECS/GEM		

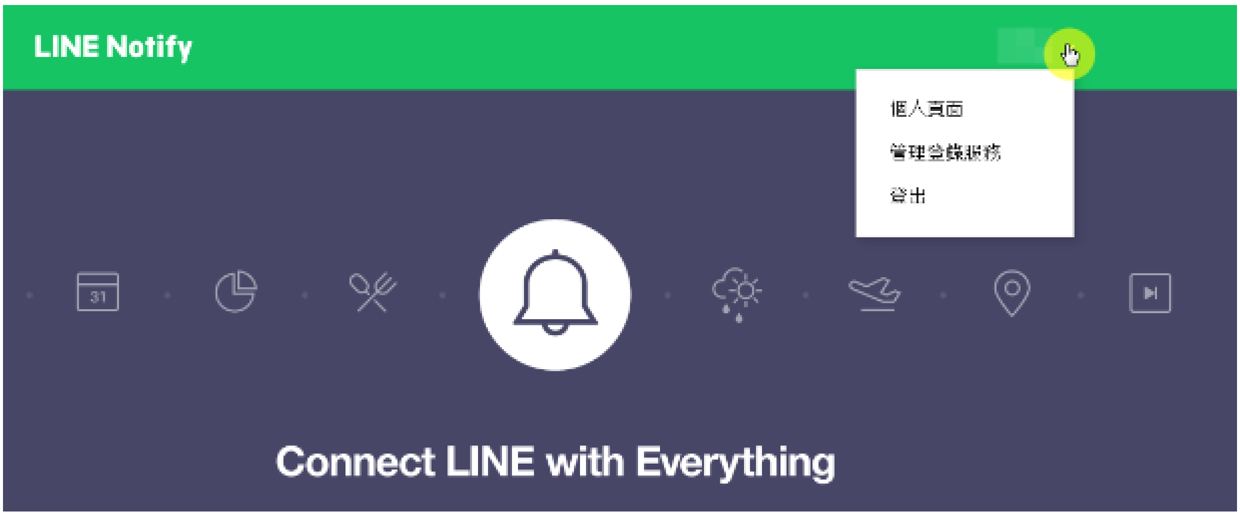
5.2 Application process for LINE Notify

1. Go to the Line Notify website (https://notify-bot.line.me/zh_TW/) and click on “Log in”.

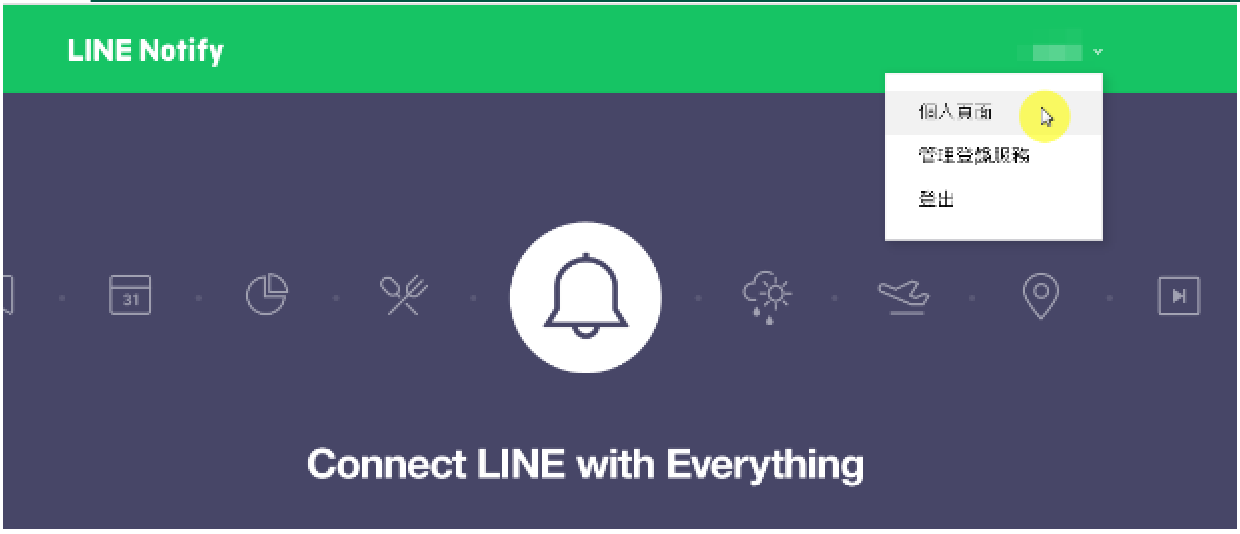


2. Please input your username and password to process with the login.

3. Once logged in, please click on the account name located on the upper left corner.



4. Click on “Personal Profile”.

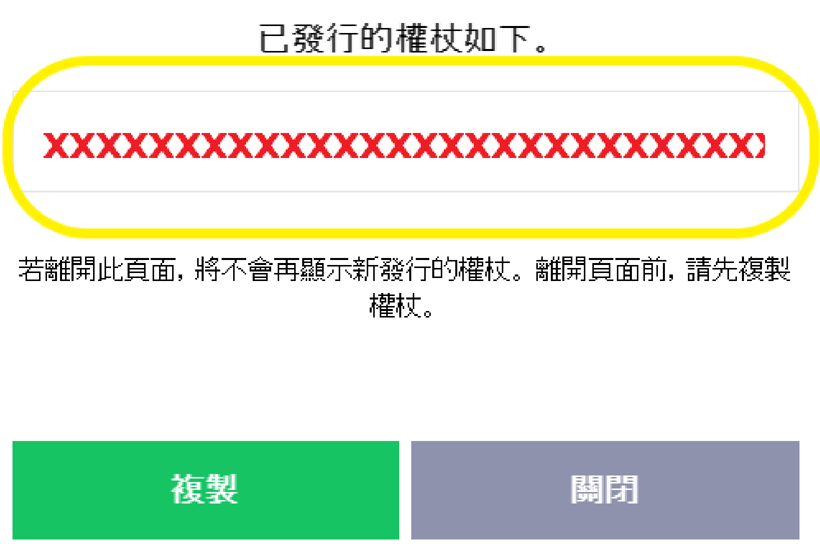


5. Click on “Issue Token”.

6. Enter the name of the token and the chat room or group where you wish to receive the message, then click on “Publish”. In this example, the token name is “VIC7000” and it uses the method of “receiving LINE Notify notifications in 1-on-1 chats”.

7. Please find and record the token as indicated by the yellow box in the figure. This token is required for setting the value of the “Token” field in the LINE Notify settings on the link.

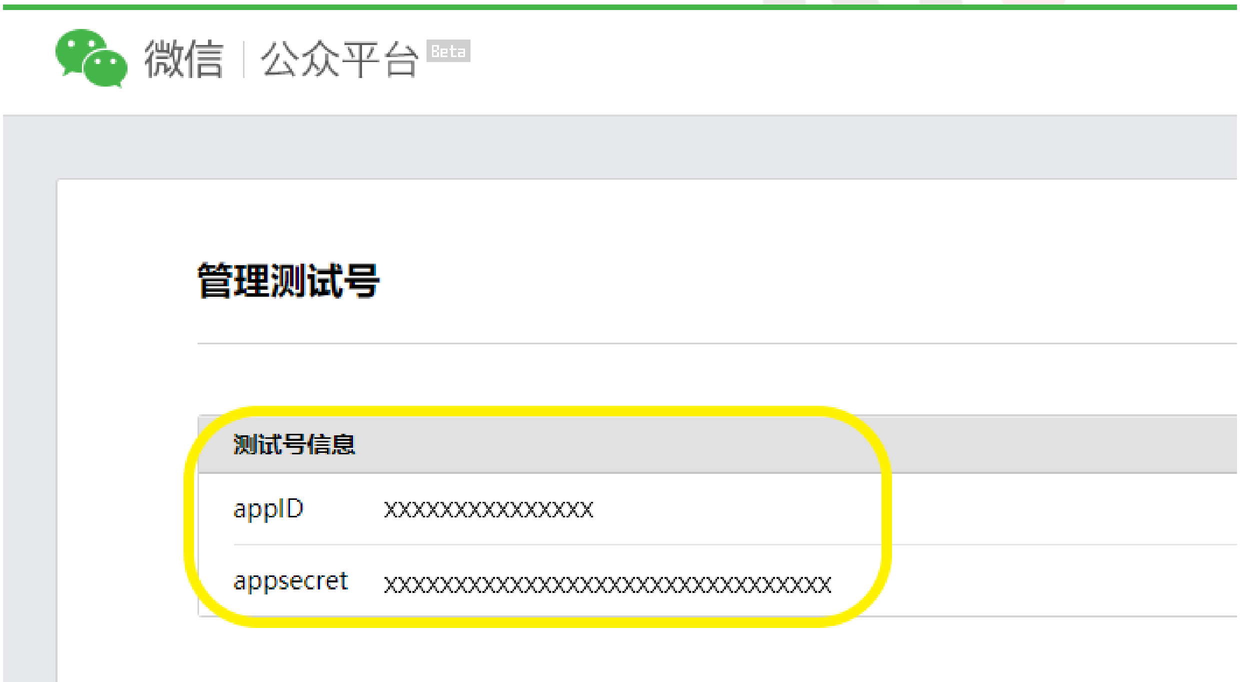
Note：This token must be recorded carefully as it will not be displayed again once you leave this page.



5.3 Application Process for Wechat Test Account

1. Go to the WeChat Offical Accounts Platform website and click on the login button. The website URL is https://mp.weixin.qq.com/debug/cgi-bin/sandbox?t=sandbox/login.

2. After logging in, record the appID and appsecret (as indicated in the yellow box in the figure), and fill in the corresponding fields in the WeChatSetting.



5.4 Application process for Wechat templateID and openID

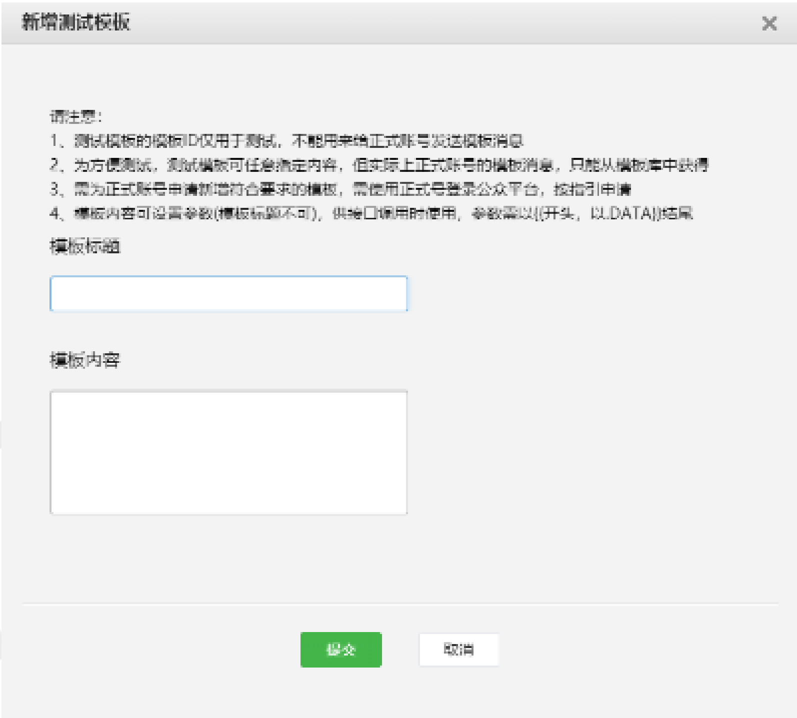
5.4.1 Setting templateID

(1) Go to the WeChat Offical Account Platform website and click on “Login”. The URL is https://mp.weixin.qq.com/debug/cgi-bin/sandbox?t=sandbox/login.

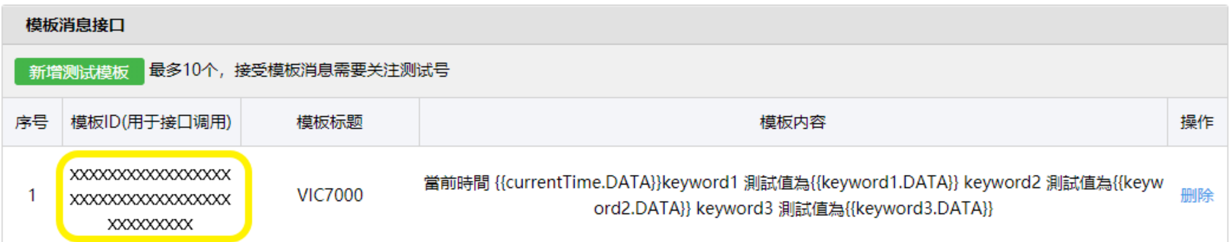
(2) Navigate to the "Template Message API" and click on "Add Test Template".

(3) Set the template title and content, then click submit. The output data format for the template is fixed and shown in the following table.

Script Parameter	Template Corresponding Content
keyword1	{{keyword1.DATA}}
keyword2	{{keyword2.DATA}}
keyword3	{{keyword3.DATA}}
Non-script Parameter	**Template Corresponding Content**
Current Time	{{currentTime.DATA}}



(4) Copy the template ID (as highlighted in the yellow box in the figure) to the "templateID" parameter in the script SEND.WECHAT or SEND.WECHAT_P function.

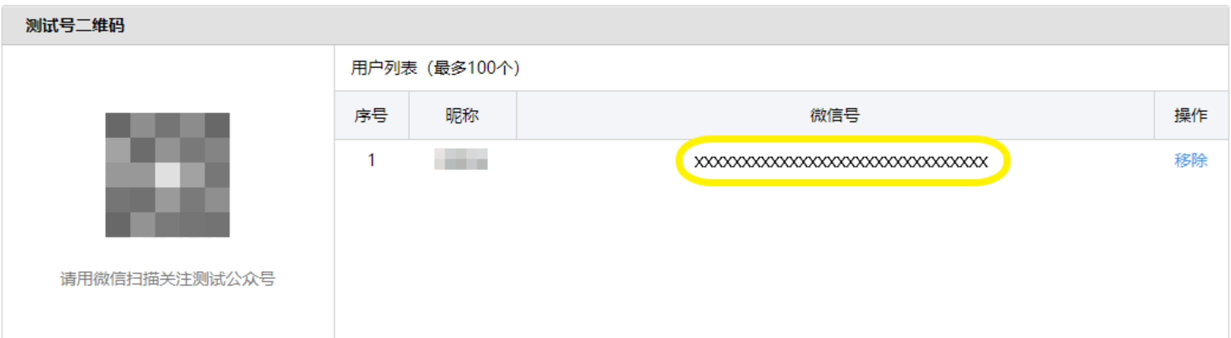


5.4.2 Setting openID

(1) Go to the WeChat Offical Account Platform website and click on “Login”. The URL is https://mp.weixin.qq.com/debug/cgi-bin/sandbox?t=sandbox/login。

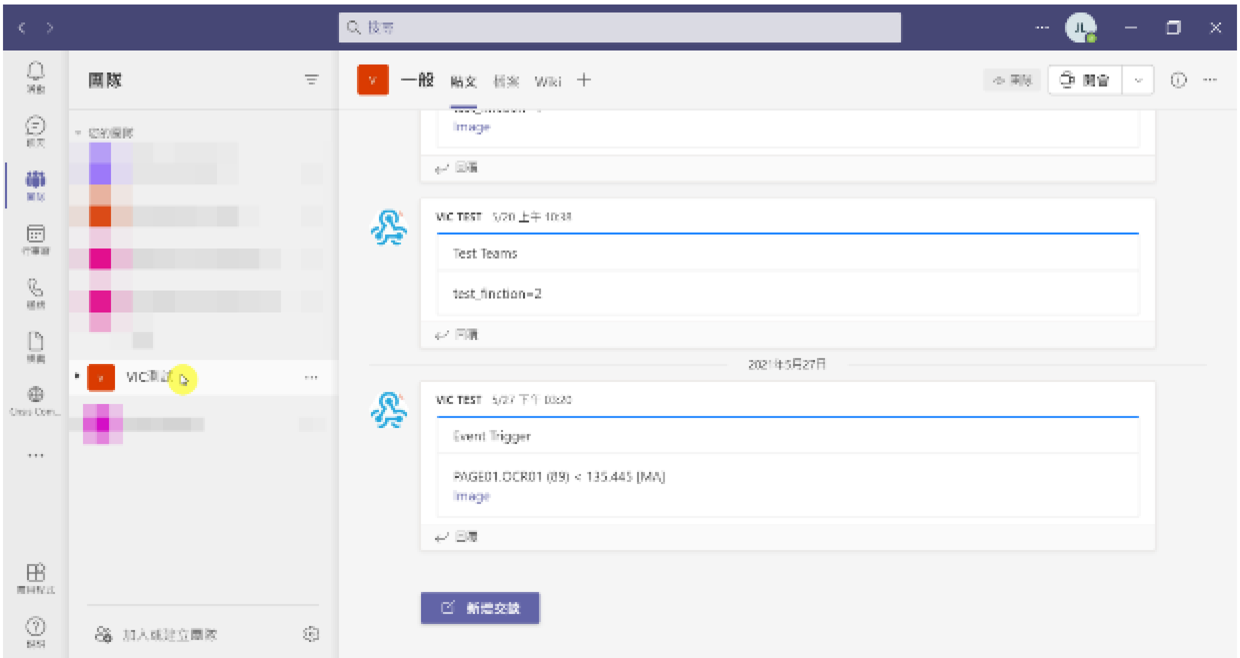
(2) Proceed to the QR Code for Test Account and scan the QR code using WeChat to subscribe to the test public account for evaluation purposes.

(3) If a user follows the testing public account, their corresponding "WeChat ID" will be displayed (as shown in the yellow box in the figure). Copy this ID and paste it into the "openID" parameter of the SEND.WECHAT_P function in the script.

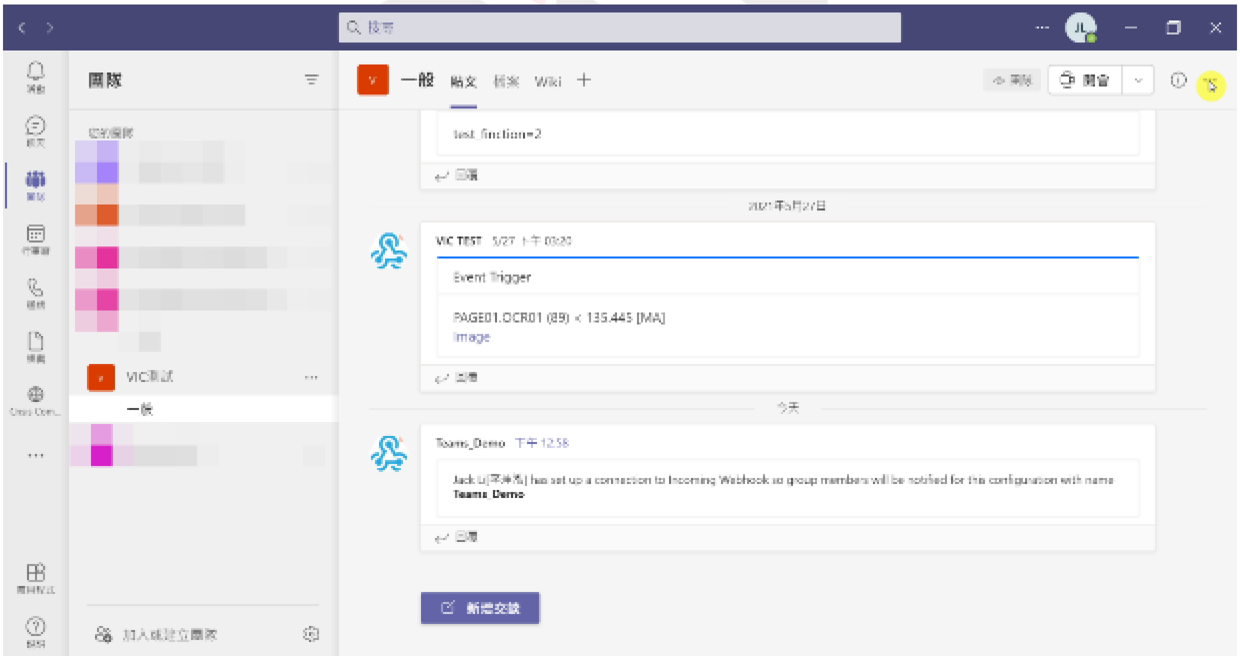


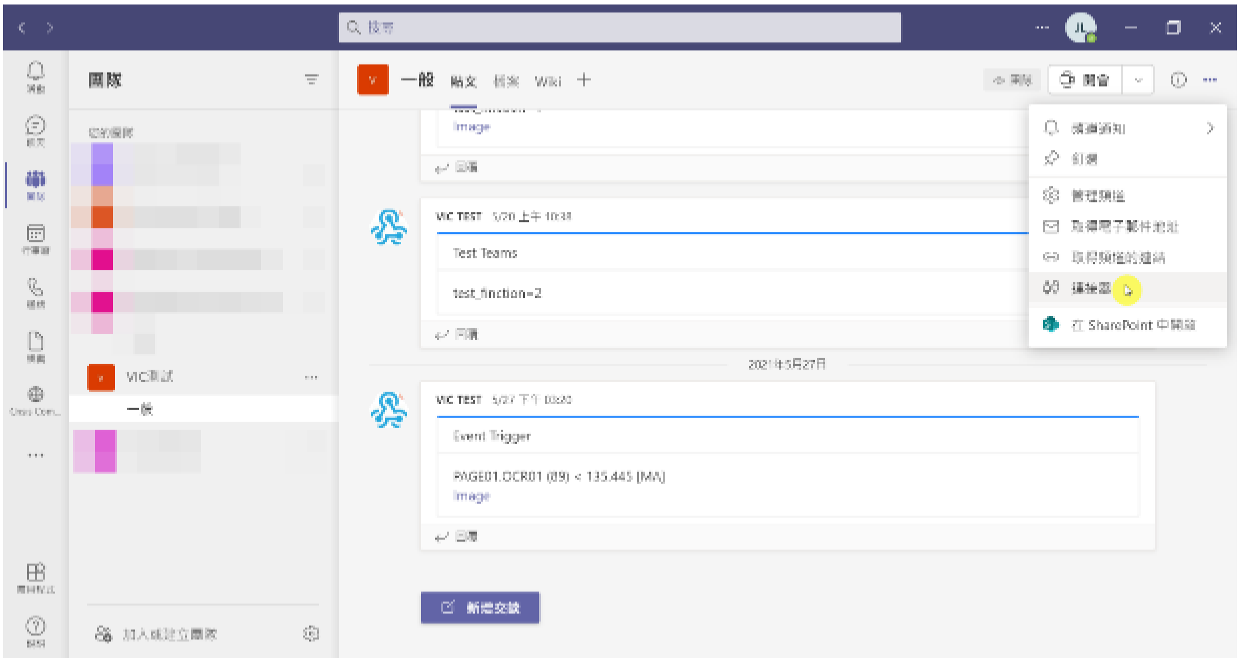
5.5 Process for Applying Teams Webhook

1. Go to the Microsoft Teams desktop application (or web verion) and click on the team you want to use.

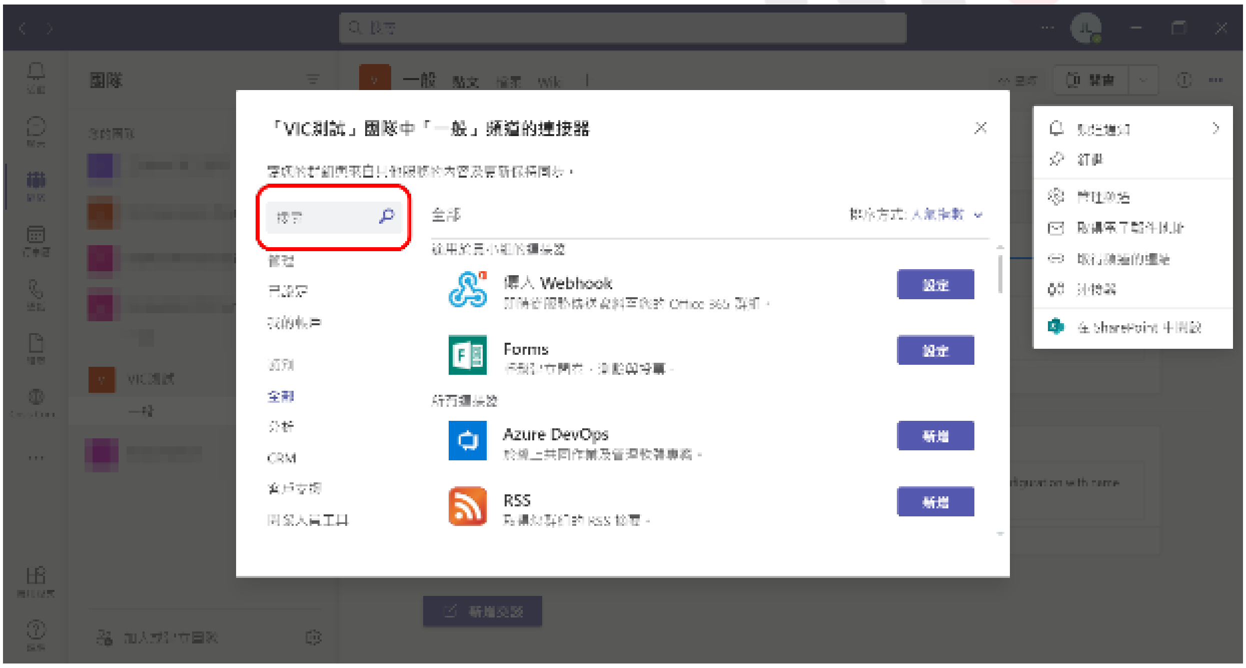


2. Click on the upper right corner and select “Connector”.





3. Enter “Webhook” into the search field.



4. Click on settings in the “Incoming Webhook” section.

5. Set the name of the webhook on the configuration, as indicated in the red box in the figure below. For example, name it “Teams_Demo”.

6. After setting the name, click on “Create”.

7. Once the webhook is created, a URL will appear (as indicated in the red box in the figure below). Copy the URL.

8. Finally, click on "Finish" and enter the URL in the Teams webhook field to complete the Teams setup.

5.6 The communication ports required for nDAS

The currently utilized communication ports by nDAS are presented in Table 5-1 as follows.

Table 5-1 Communication ports used by nDAS

Port	Content
80	Default communication port for nDAS network server.
502	Default communiction port for Modbus TCP.
5000	Default communiction port for nDAS SECS/GEM.
8001	Default communiction port for nDAS UDP.
8002	Default communiction port for nDAS UDP multicast.
8004	Default communiction port for nDAS TCP command.