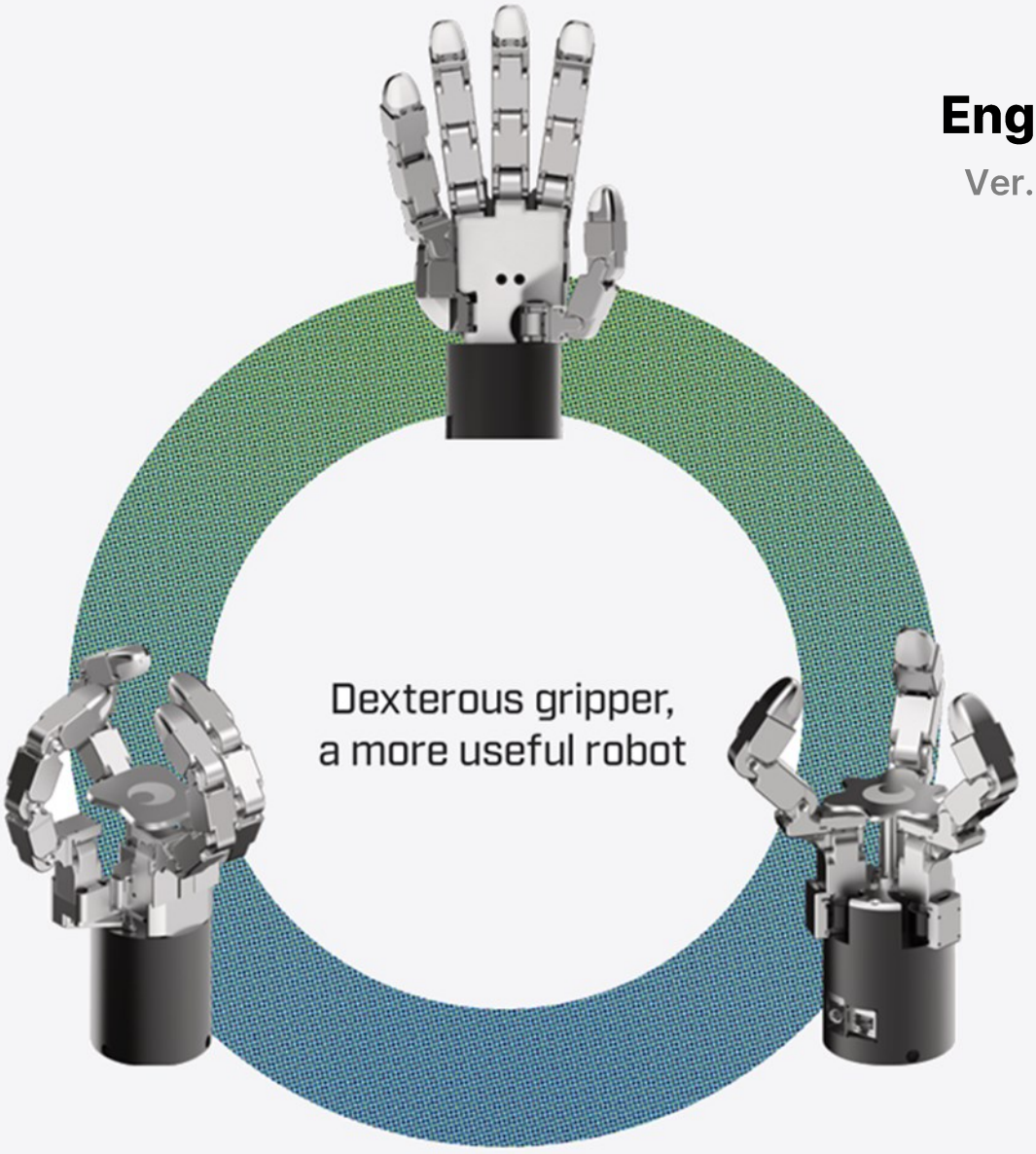
Delto Gripper

TMflow2 Component

User Manual



English

Ver. 1.0.2

INFORMATION

Copyright

Notice of revisions to the user guide

This documentation is subject to revision for technical improvements.

Manual Version: 1.0.2 English (2025.09)

Tesollo Corporation

Headquarters: 26th floor, POSCO Tower, 6-10 Songdo-dong, Yeonsu-gu, Incheon, Korea
Office: 26F, GIDC Tower A, 514, Iljik-dong, Gwangmyeong-si, Gyeonggi-do, Republic of Korea
Support T: +82 02 6914 6620 E: support@tesollo.com
H: http://en.tesollo.com/

Table of Contents

Chapter 1 Getting started

This document is designed to guide you through the installation and use of the Delto Gripper M in TM Robot. The test environment in this document is based on TMflow version 2.20. Depending on the version you are using, the user interface (UI) may differ.

Although basic protocol compatibility is maintained among DG series products, any input values exceeding the number of valid joints supported by each product are ignored.

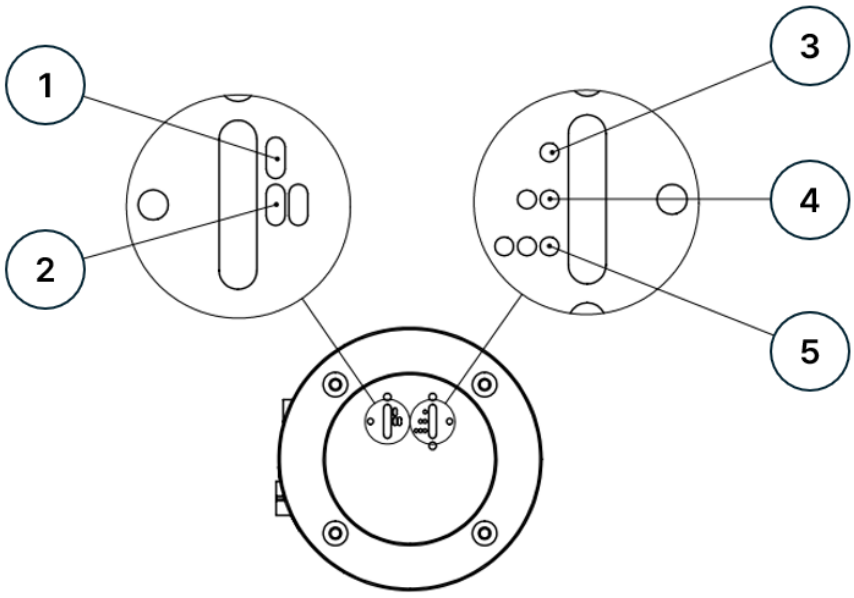
- DG-5F: supports up to 20 joint values.
- DG-4F: supports up to 18 joint values. Joints 17 and 18 are used as base joints 1 and 2.
- DG-3F: supports up to 12 joint values.
- DG-2F: supports up to 6 joint values.
- DG-1F: supports up to 3 joint values.

If you have any questions, please contact the email below.

Email: support@tesollo.com

1.1 Preparation Before Gripper Engagement

Set the switches on the bottom of the gripper to ①-④ before attaching it to the collaborative robot.



< Bottom view of the gripper>

No.	Control Modes	
①	Operator Mode	🔴🟡🔵 x 1 [All LEDs Blink Once]
②	Developer Mode	🔴🟡🔵 x 2 [All LEDs Blink Twice]
No.	**Communication Modes**	
③	RS485	○🟡○ [Yellow LED is On]
④	EtherNET	○○🔵 Socket Connect : Blue LED is On Socket Disconnect : Blue LED Blinks
⑤	Not used	Not used

① **Operator Mode:** This mode uses the internal controller of the product for operation. It supports the Modbus Protocol, allowing users to easily control the robotic hand by simply modifying the register map or using the provided UI.

② **Developer Mode:** This mode allows the user to control the robotic hand using a custom-developed controller. It enables the reception of information from each joint of the robotic hand, allowing direct control of each joint.

* When power is first applied, the control mode LED blinks, followed by the operation of the communication mode LED.
* Developer Mode is only available via Ethernet communication.
* If there is an issue with switch operation or communication, the red LED blinks.

*** Operator Mode and SDK: stall torque only available with 'Torque limit mode' off**

*** Developer Mode: Always use stall torque**

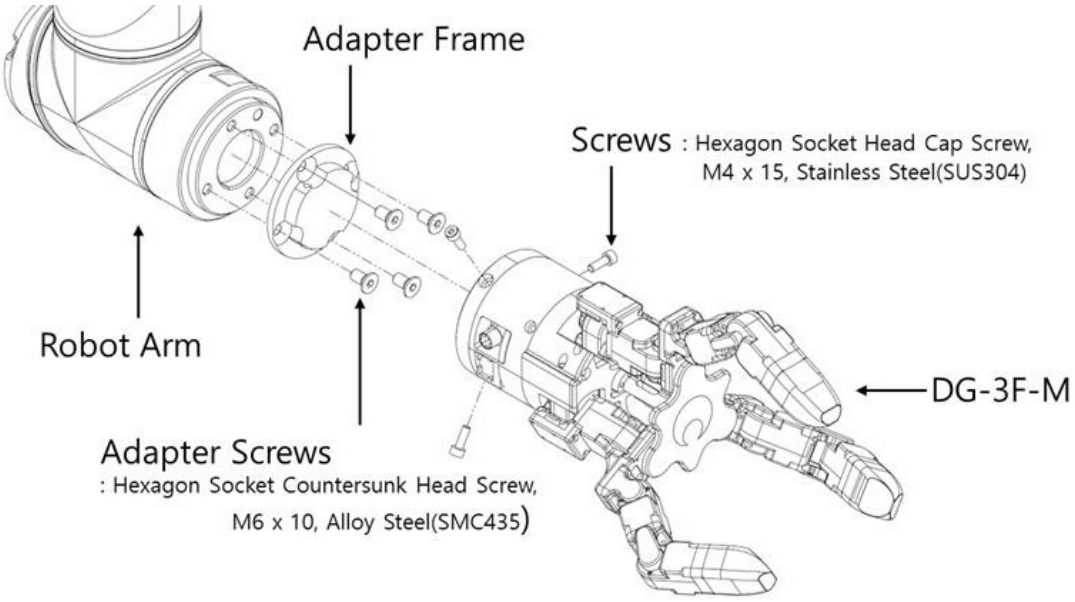
*** Caution**

Using 'Torque Limit Mode' off in Por using stall torque in developer mode may cause a rapid increase in product temperature and potential product damage. Please note that repairs (A/S) may be difficult in case of failure.

1.2 Gripper engagement

Attach the included Adapter frame to the end of the Collaborative Robot.

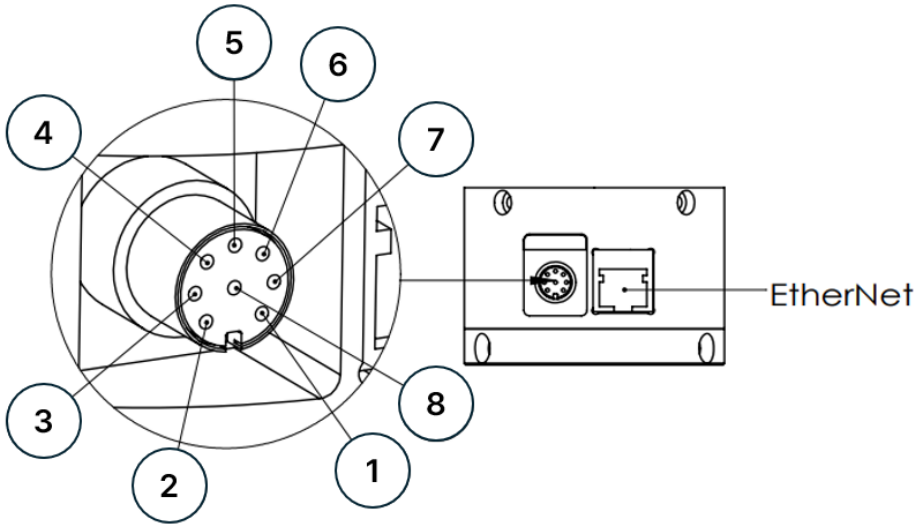
Attach the Adapter frame and the Delto Gripper.



<Example image of DG-3F-M collaborative robot fastening>

1.3 Power supply

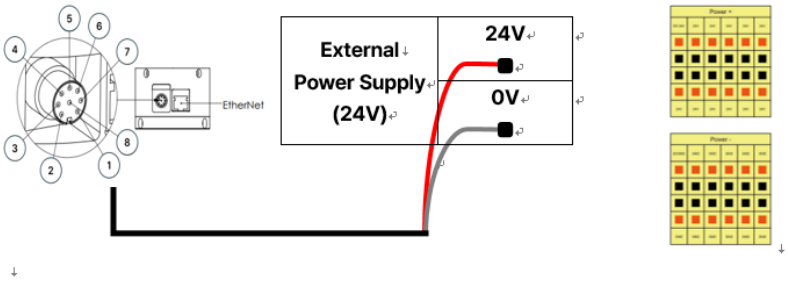
Connect the GPIO 8-pin cable to the gripper and then connect the power according to the table below.



< Gripper connector image >

No.	Color	Connection
1	White	RS485 A
2	Brown	RS485 B
3	Green	DI1 (Gripper Output)
4	Yellow	DI0 (Gripper Output)
5	Gray	GND
6	Pink	DO1 (Gripper Input)
7	Blue	DO0 (Gripper Input)
8	Red	24V DC
Input power DC 24V Requires use of a 5A or higher power supply Check the manual provided with your purchase and the labeling on the GPIO cable		

The TM control box provides internal power (24V/2A), and the maximum allowable current when using the external input port is 3.5 A. Since the maximum current draw of the gripper exceeds 3.5 A, it is recommended to power the gripper directly using a separate external power source.

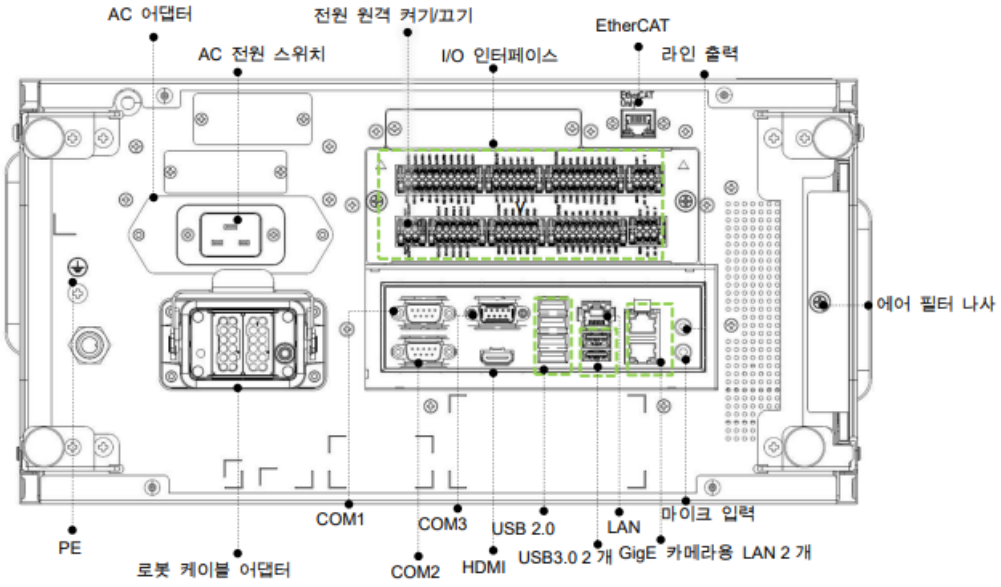


< TM Control Box >

(Image source: TM5-Hardware-Installation-anual_HW3.2_Rev1.08_EN.pdf)

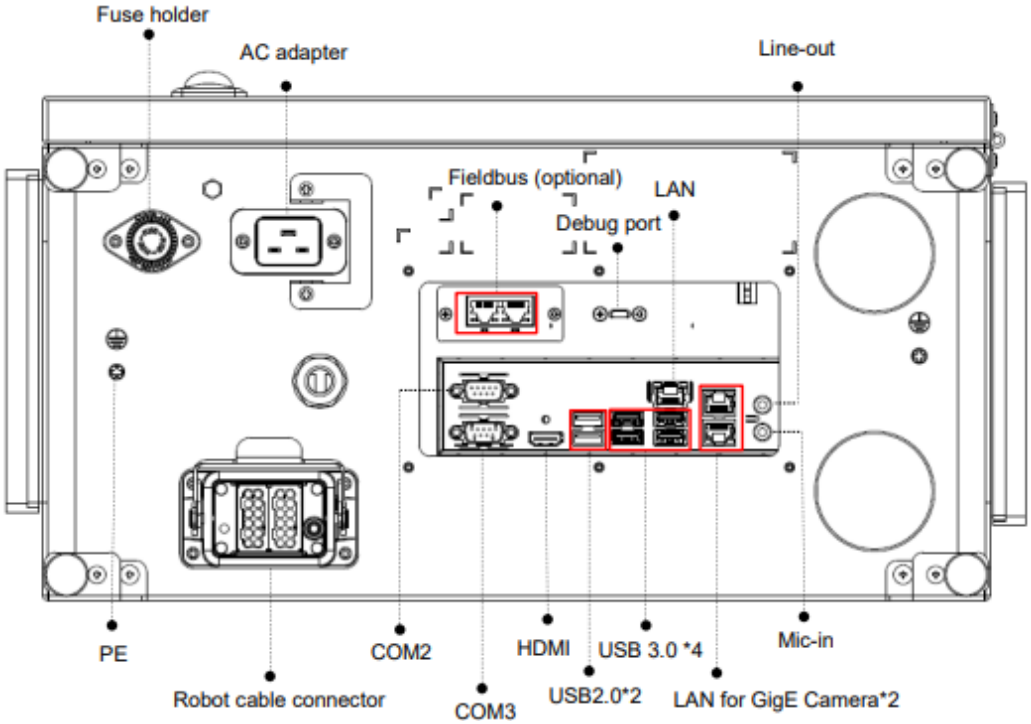
1.4 Gripper Communication

Connect the gripper to the LAN port on TM Robot's control box.



< TM Control Box Bottom >

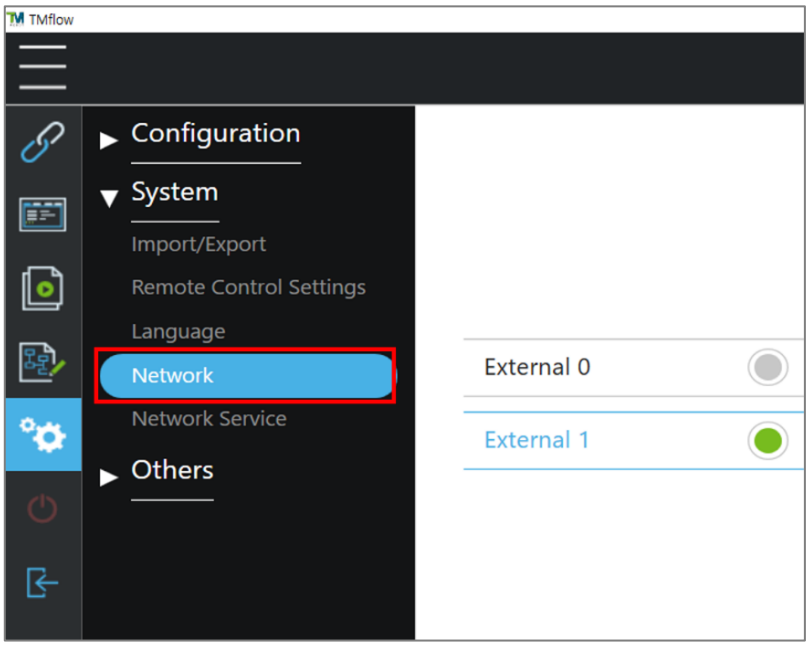
(Source: Hardware_TM5-Hardware-Installation-Manual_HW3.2_Rev1.06_EN.pdf)



< TM Control Box Bottom >

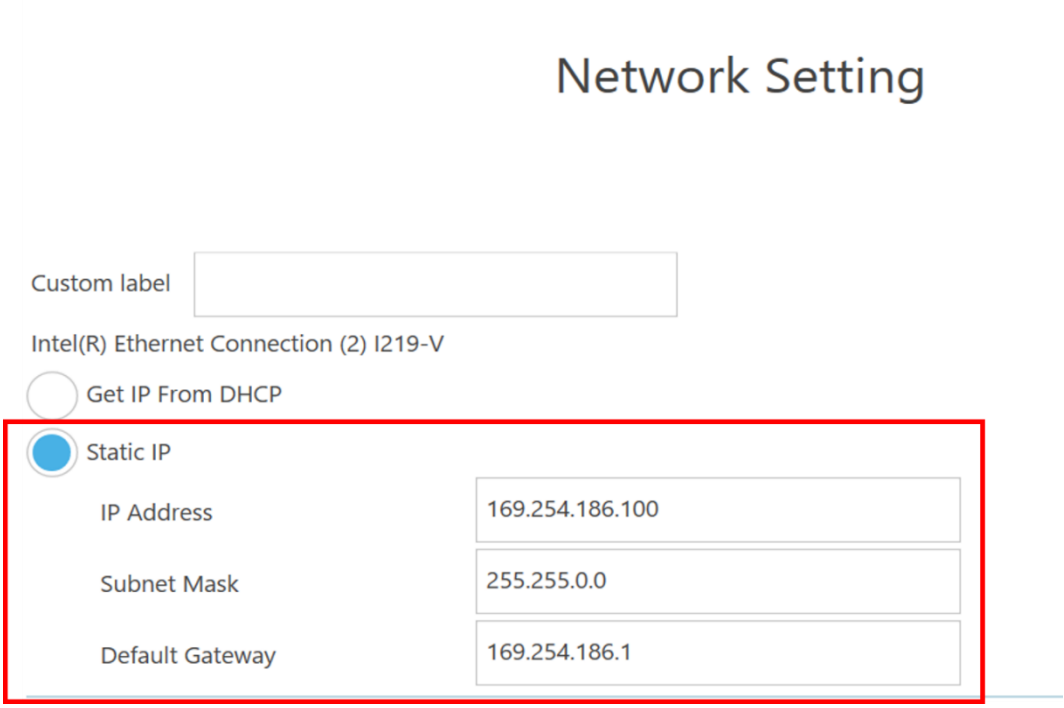
(Source: Hardware_TM5S-TM7S-Installation-Manual_HW5.02_1.04_EN.pdf)

In the menu of the TM control box, go to [System] → [Network].



< Gripper Communication - Network Setting Example 1 >

The default IP for the gripper is 169.254.186.72. Set the IP and Subnet Mask of the TM Control Box to the same band, using the example in the figure below:

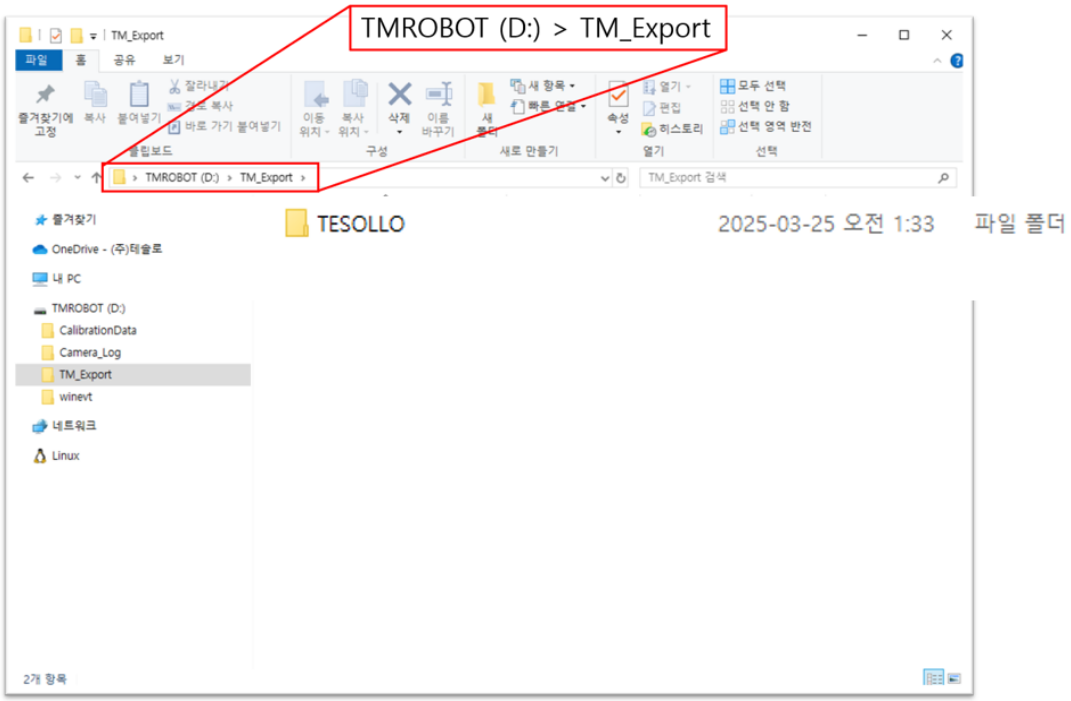


< Gripper Communication - Network Settings Example 2 >

1.5 Download and apply TM Plug & Play

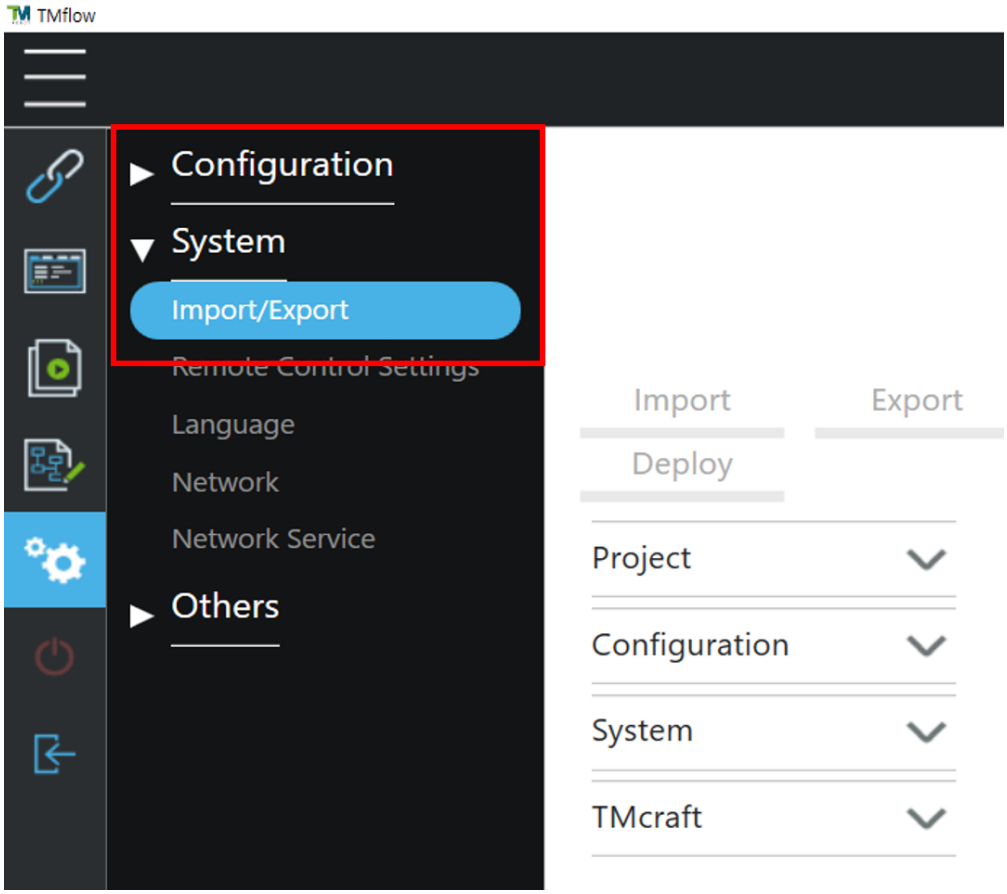
Download the [Delto Gripper TM Plug & Play] archive from the official website under [Support] → [Resources].

Place the downloaded file in the TM_Export folder on the USB drive named TMROBOT.

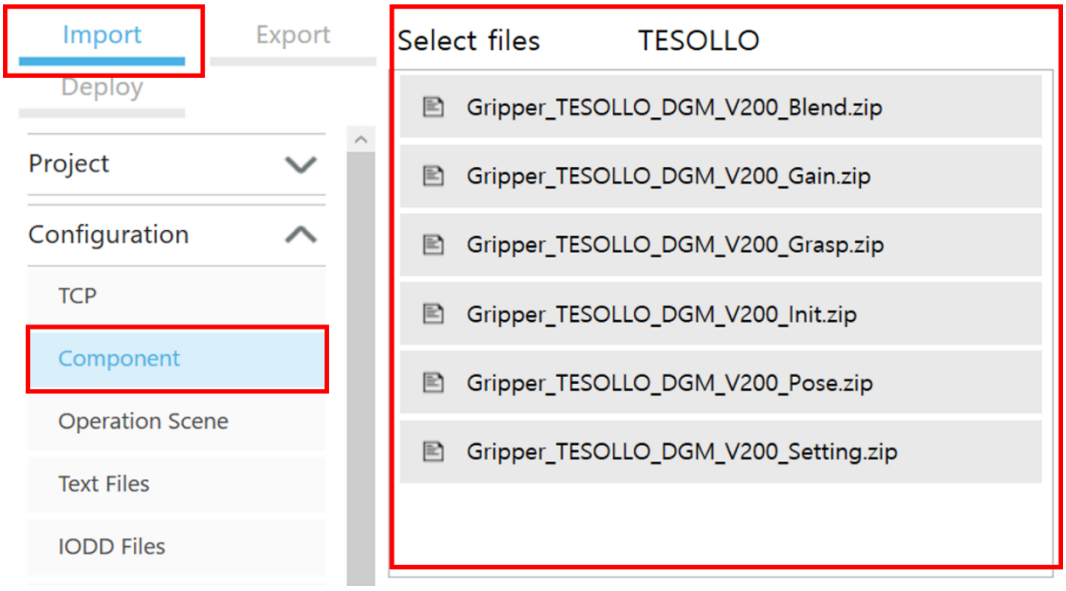


< How to apply Plug & Play 1 >

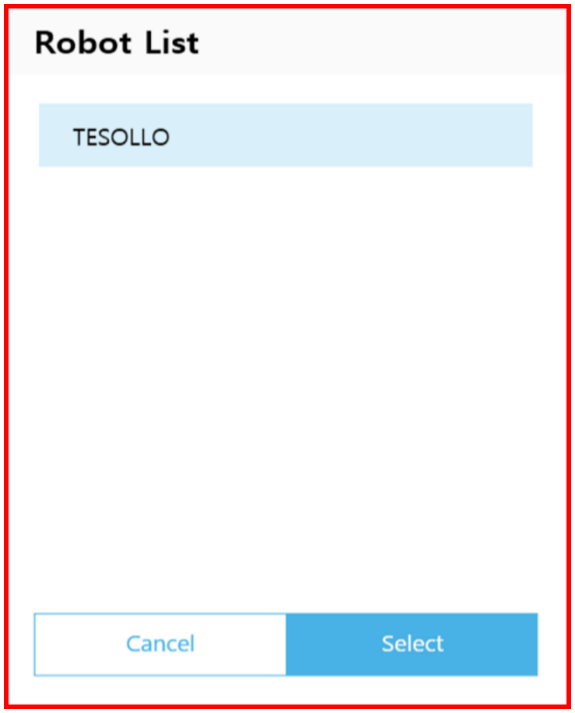
Insert the USB into the control box and go to, System -> Import/Export to import the Component and Project.



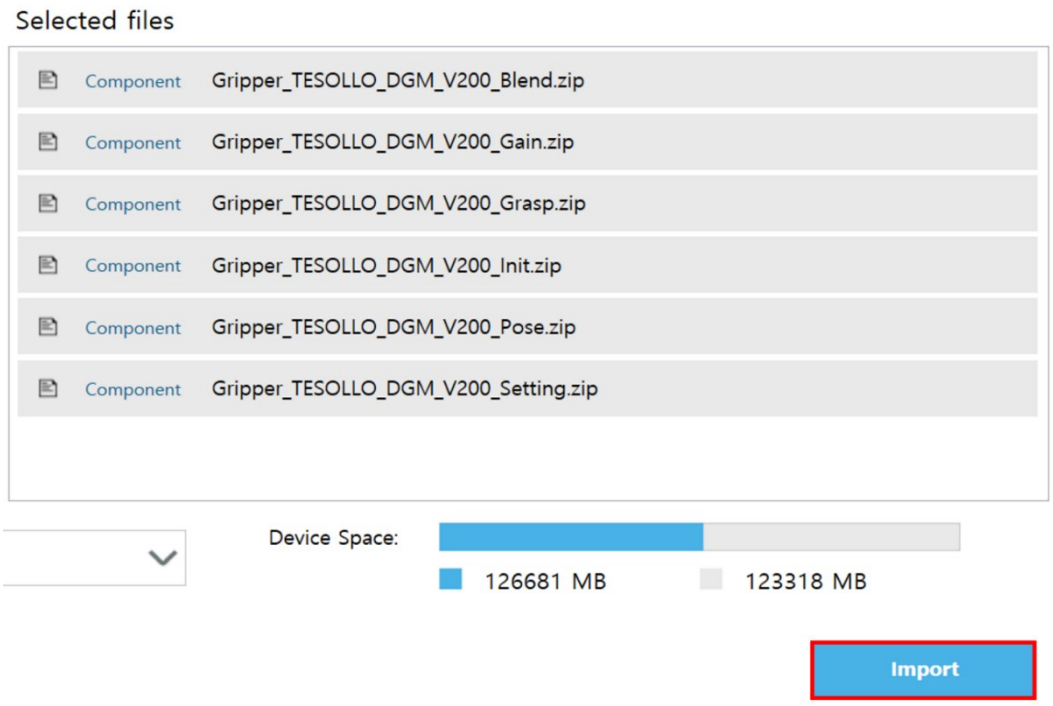
< How to apply Plug & Play 2 >



< How to apply Plug & Play 3 >

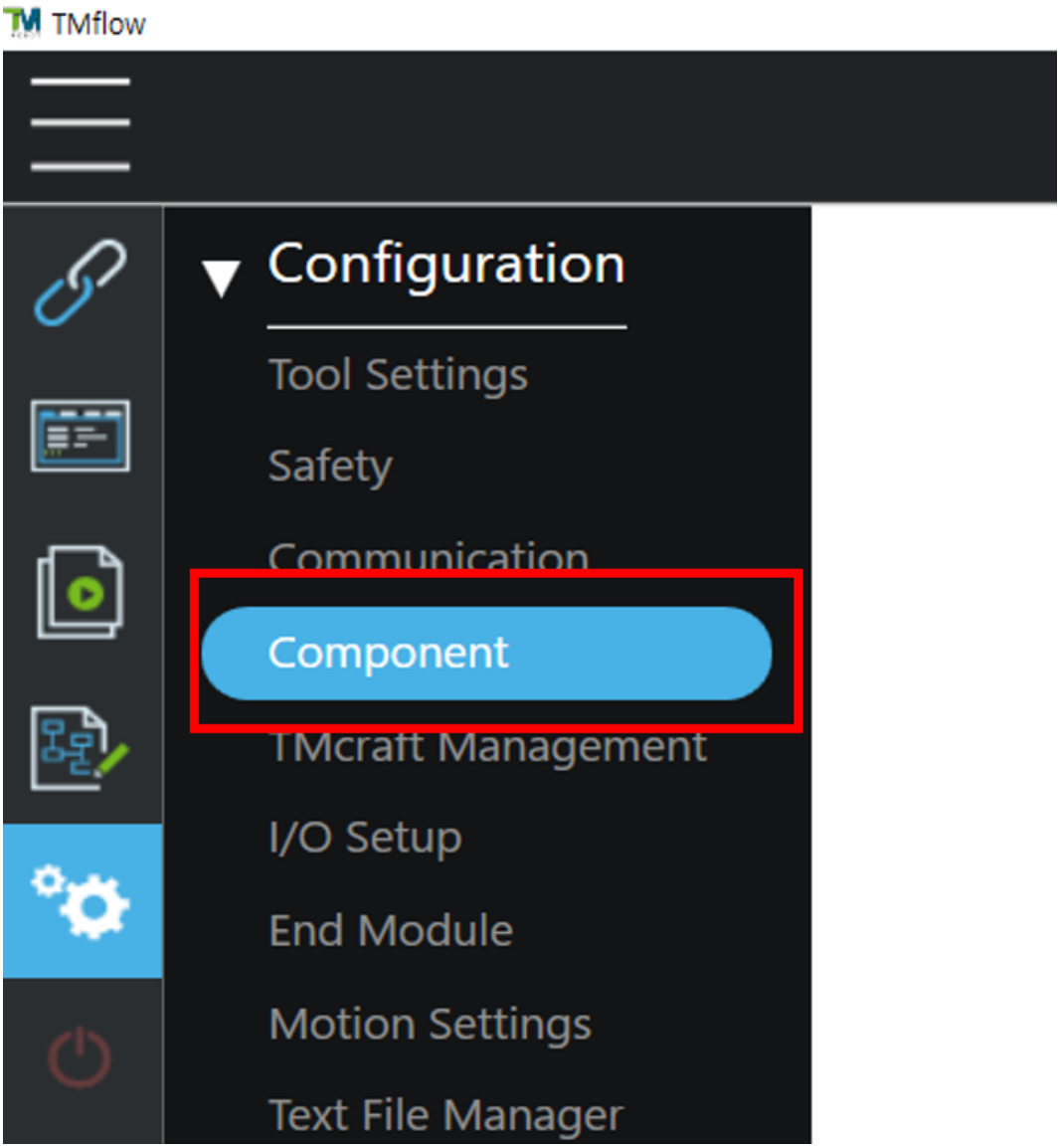


< How to apply Plug & Play 4 >

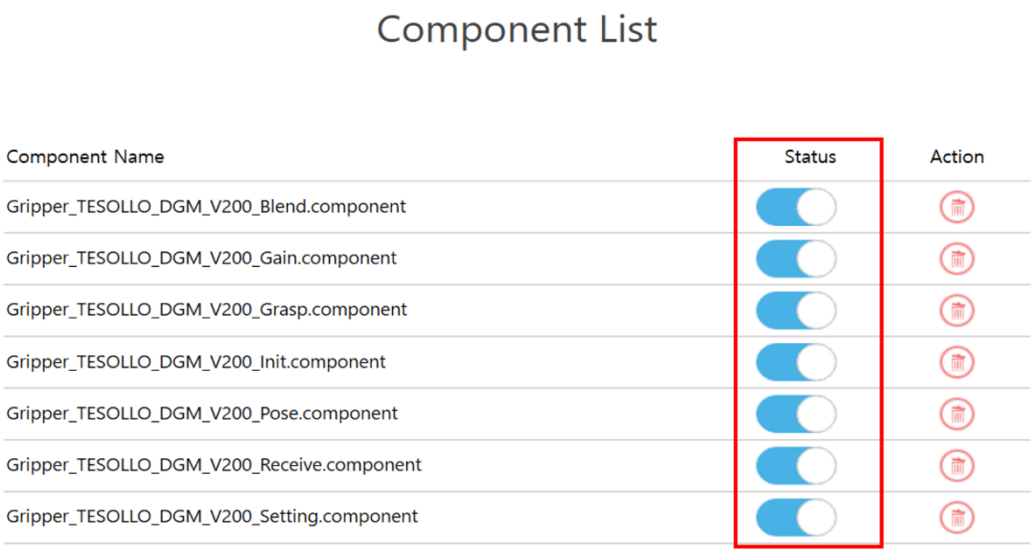


< How to apply Plug & Play 5 >

After importing the Component, proceed to activate it. ☰ Go to -> System -> Component and check the items you want to use and save them.



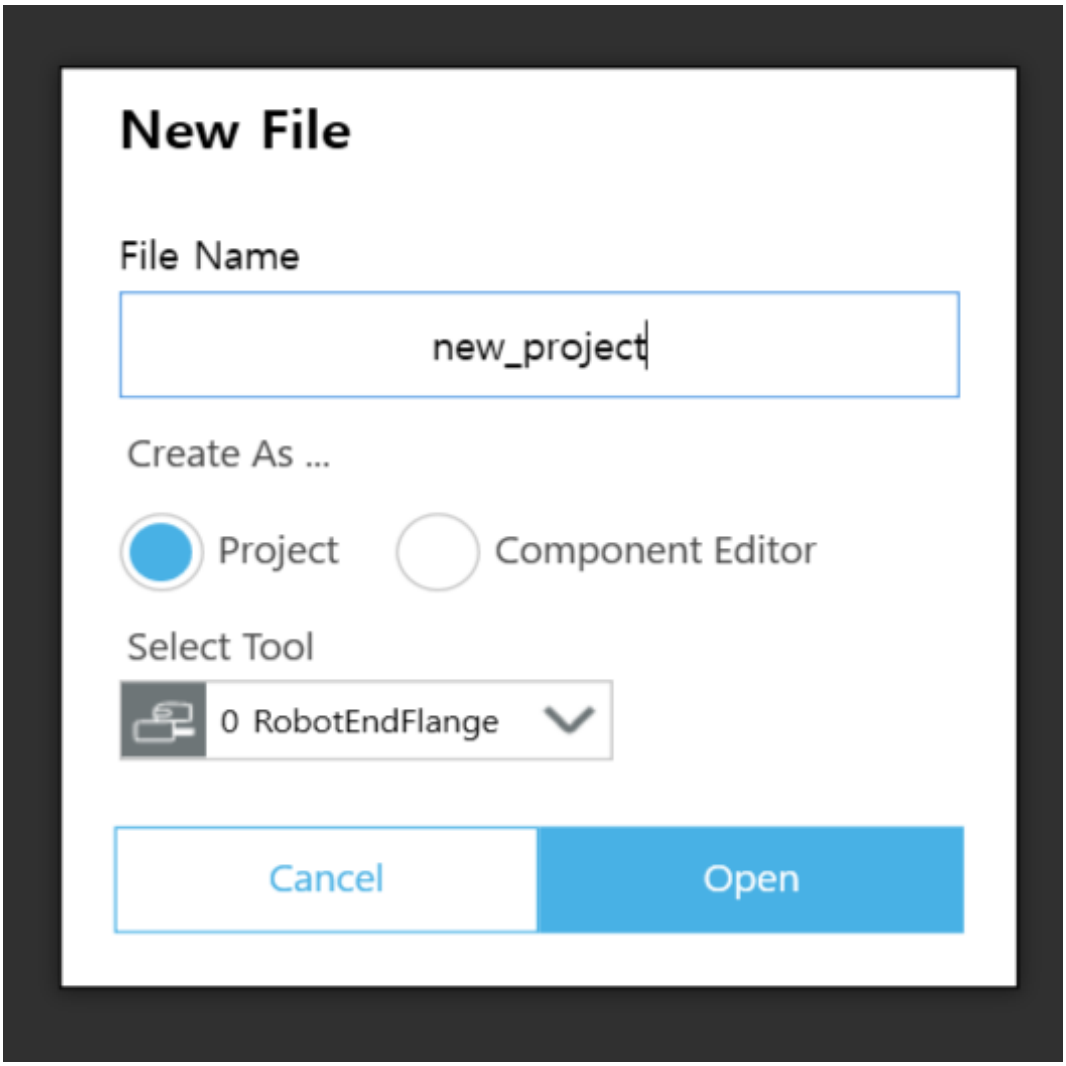
< How to apply Plug & Play 6 >



< How to apply Plug & Play 7 >

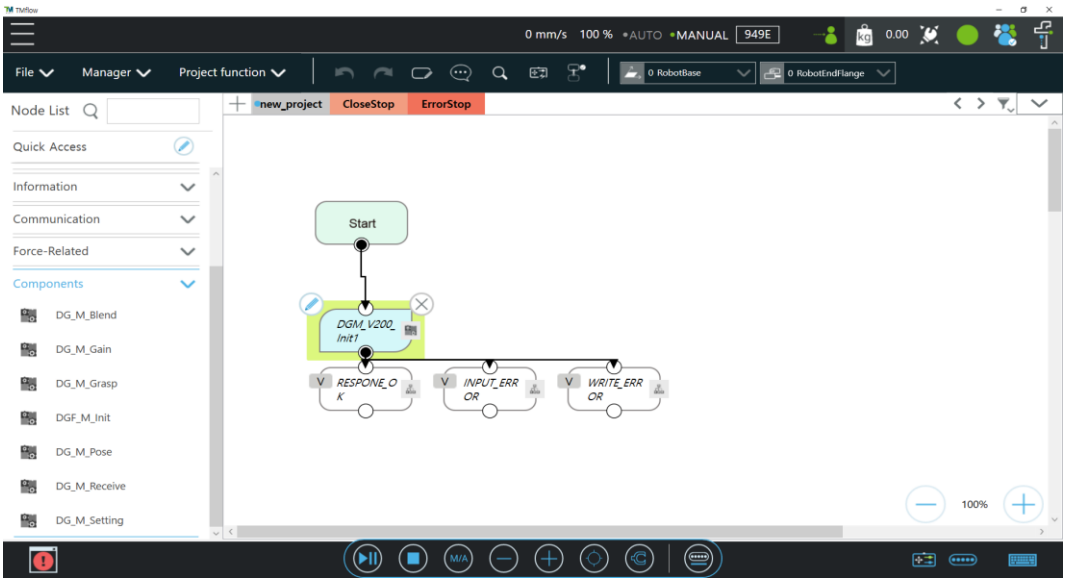
Chapter 2 How to use components

Create a project



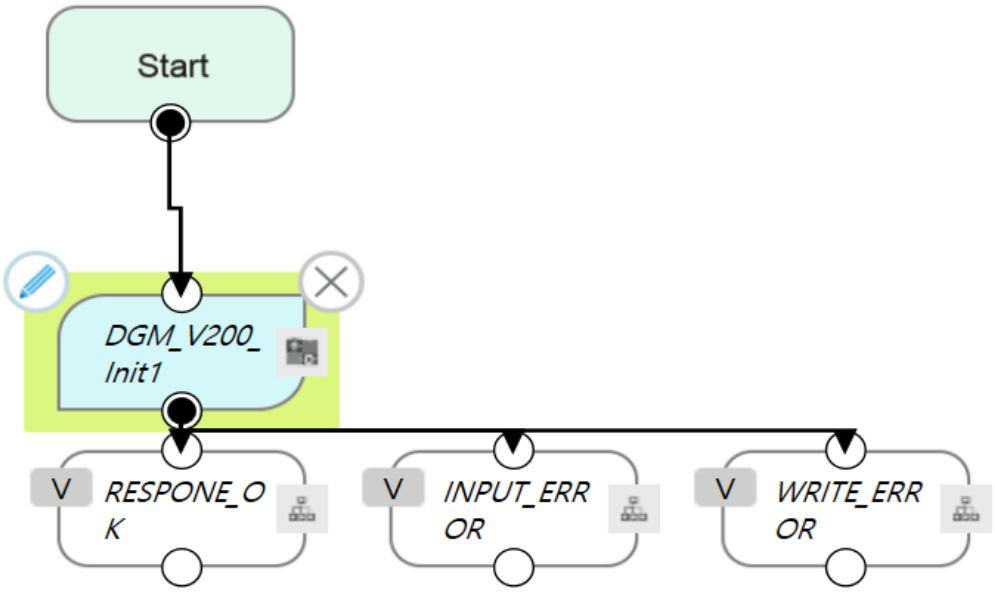
< How to Use Component 1 >

Drag the module you want to use and insert it into your project

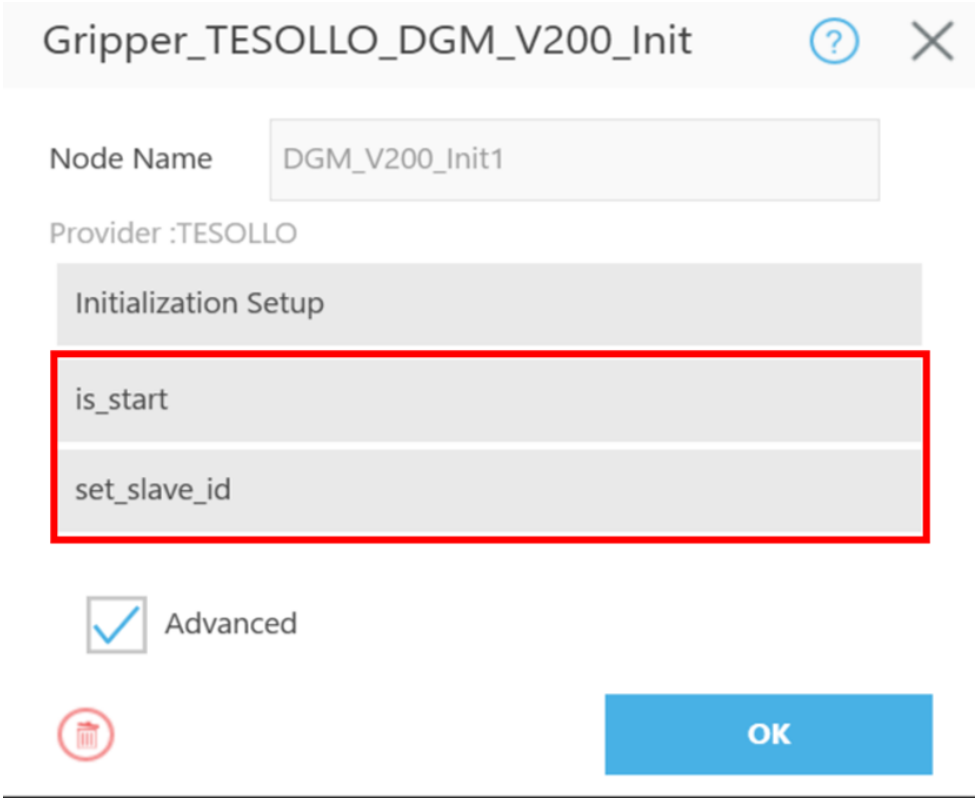


< How to use Components 2 >

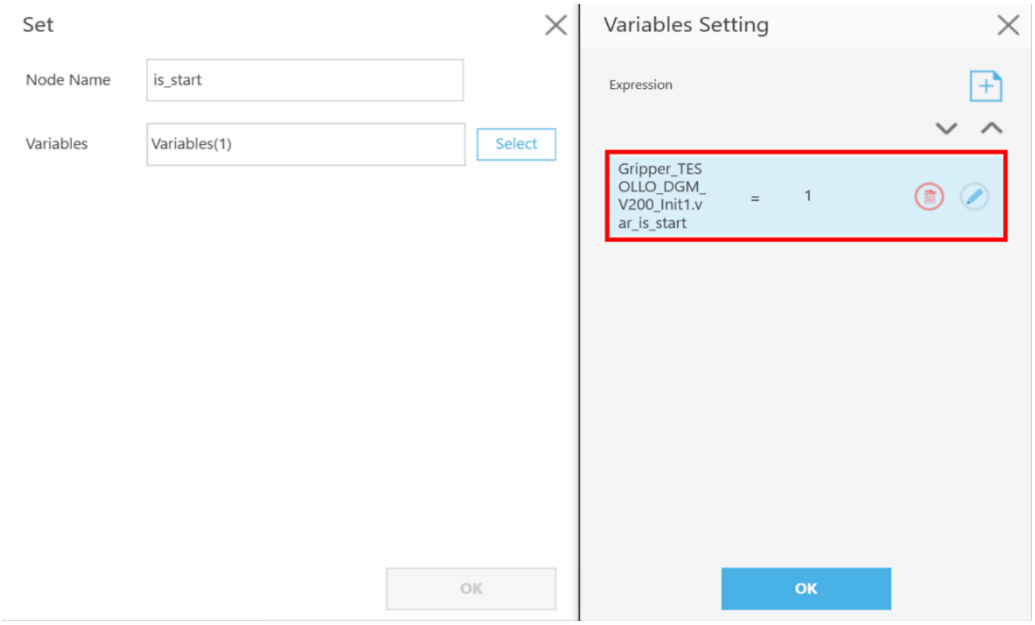
Enter the values required for the gripper behavior



< How to use Components 3 >



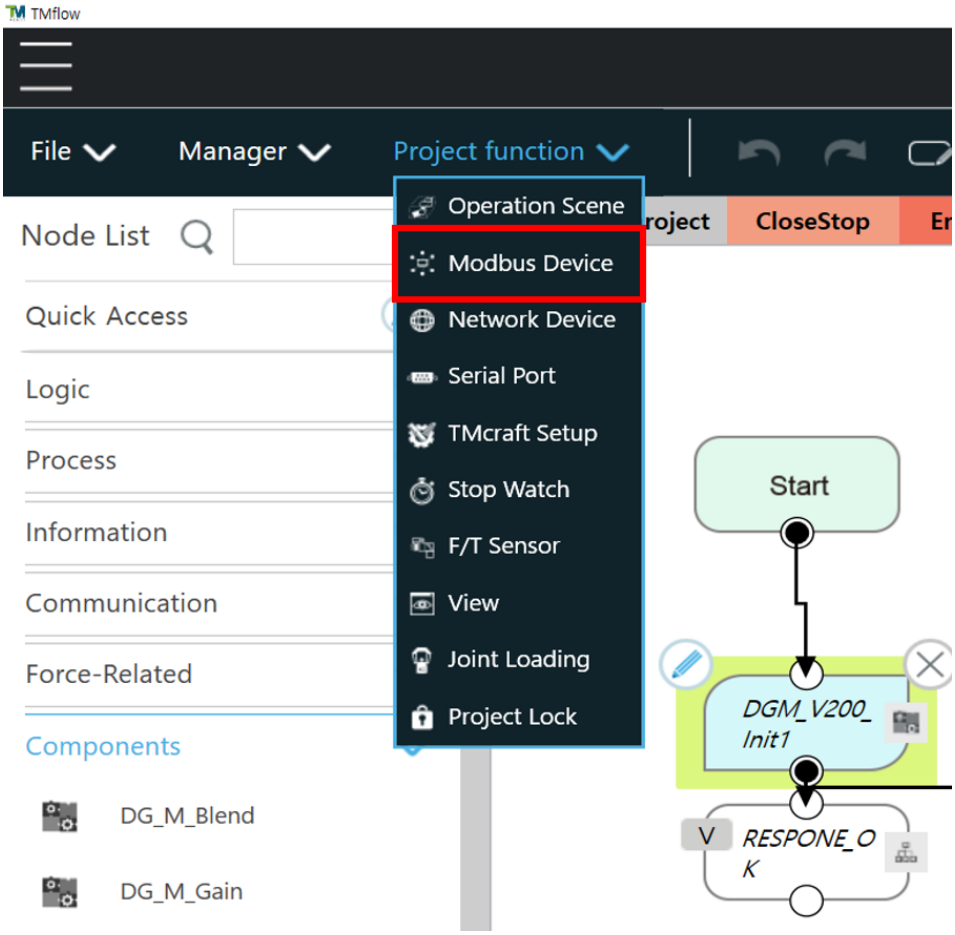
< How to Use Components 4 >



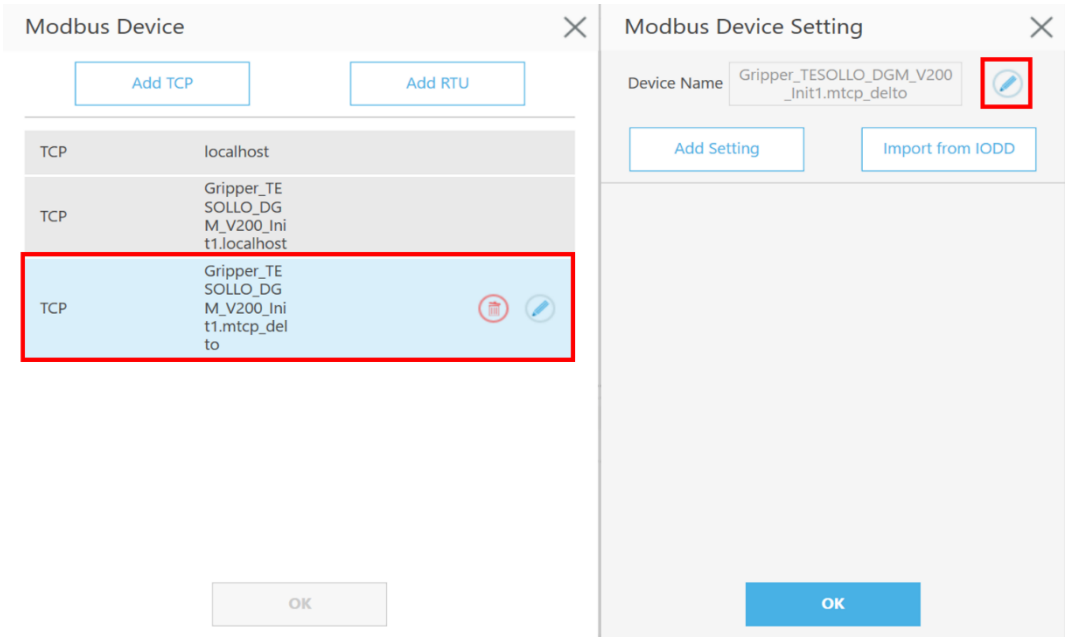
< How to Use Components 5 >

Note: If you change the IP or PORT of the gripper, you must also modify the TCP device ("mtcp_delto") setting for each component in the Modbus Device settings to do the same.

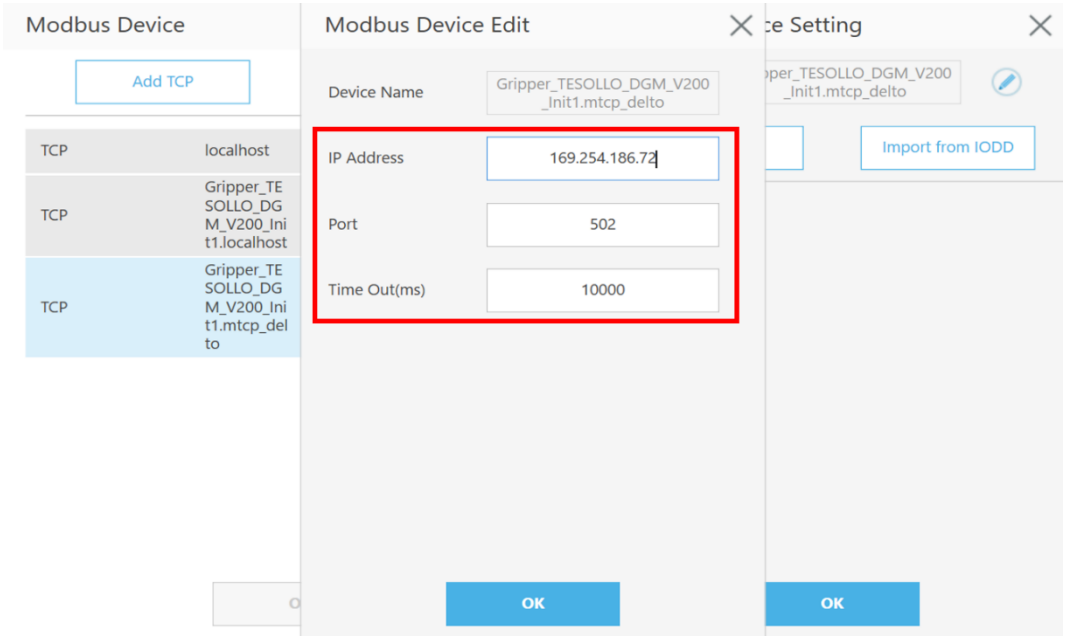
- Default IP: 169.254.186.72, Default port number: 502



<How to Use Components 6 >



< How to Use Components 7 >



< How to Use Components 8 >

Chapter 3 Component description

3.1 NOTE

3.1.1 ModbusTCP

This component is developed based on the Modbus TCP protocol.

If you need a more detailed description or additional features, please refer to the Modbus topic in the separately available product manual.

If you want to use a user-set Slave ID that is not the default Slave ID,
You can set the Slave ID of the gripper you want to connect to in the Advanced Options of each Node.

Only if the Flag value is True in each setting item will the set Value value actually be sent to the gripper.
If the data you set is not saved to EEPROM, it will be restored to the previous value when the device restarts.
To keep the changed settings permanently, be sure to execute SET_EEPROM_WRITE to save them to EEPROM.

When data is entered for each finger to be transmitted to the gripper, the values are merged into a single data sequence and transmitted in one operation. Accordingly, in the following cases, the gripper executes the same operation:

e.g.

Case 1. finger1 = {1,1,1}, finger2 = {1,1,1} → transmitted data = {1,1,1,1,1,1}

Case 2. finger1 = {1,1,1,1}, finger2 = {1,1} → transmitted data = {1,1,1,1,1,1}

Data Types	Variables	Description
int	slave_id	Setting the Modbus Connection Slave ID

3.2 Init

3.2.1 Is_start

Start controlling the gripper

Data Types	Variables	Description
int	is_start	0: Control off 1: Control on

3.3 Gain

Components that allow you to manipulate functions related to the gain value of the Delto Gripper.

3.3.1 set_p_gain

Data Types	Variables	Description
bool	send_p_gain_flag	Set whether to send data to the gripper
float[]	finger1_p_gain	Set the p-gain for finger 1
float[]	finger2_p_gain	Set the p-gain for finger 2
float[]	finger3_p_gain	Set the p-gain for finger 3
float[]	finger4_p_gain	Set the p-gain for finger 4
float[]	finger5_p_gain	Set the p-gain for finger 5

Note

* In DG-4F, set the base joint using the variable for finger 5

3.3.2 set_d_gain

Data Types	Variables	Description
bool	send_d_gain_flag	Set whether to send data to the gripper
float[]	finger1_d_gain	Set the d-gain for finger 1
float[]	finger2_d_gain	Set the d-gain for finger 2
float[]	finger3_d_gain	Set the d-gain for finger 3
float[]	finger4_d_gain	Set the d-gain for finger 4

float[]	finger5_d_gain	Set the d-gain for finger 5

Note

* In DG-4F, set the base joint using the variable for finger 5

3.3.3 set_i_gain

Data Types	Variables	Description
bool	send_i_gain_flag	Set whether to send data to the gripper
float[]	finger1_i_gain	Set the i-gain for Finger 1
float[]	finger2_i_gain	Set the i-gain for Finger 2
float[]	finger3_i_gain	Set the i-gain for finger 3
float[]	finger4_i_gain	Set the i-gain for finger 4
float[]	finger5_i_gain	Set the i-gain for Finger 5
float[]	finger1_i_limit	Set the i-limit for Finger 1
float[]	finger2_i_limit	Set the i-limit for Finger 2
float[]	finger3_i_limit	Set the i-limit for Finger 3
float[]	finger4_i_limit	Set the i-limit for Finger 4
float[]	finger5_i_limit	Set the i-limit for Finger 5

Note

* In DG-4F, set the base joint using the variable for finger 5

3.3.4 set_load_gain_recipe

You can store up to 20 gain recipes (1-20)

Data Types	Variables	Description
bool	send_load_gain_flag	Set whether to send data to the gripper
int	load_gain_recipe	Set the gain recipe to call

3.3.5 set_save_current_gain

You can store up to 20 gain recipes (1-20)

Data Types	Variables	Description
bool	send_save_gain_flag	Set whether to send data to the gripper
int	save_gain_recipe	Save the current gain recipe with that

		number

3.3.6 set_pid_gain_control

Data Types	Variables	Description
bool	send_gain_control_flag	Set whether to send data to the gripper
int	gain_control	0: PD control 1: PID Control

3.4 POSE

Components that allow you to manipulate the functionality associated with each joint pose.

3.4.1 set_save_current_pose

Up to 100 pose recipes are stored.

Data Types	Variables	Description
bool	send_save_pose_flag	Set whether to send data to the gripper
int	save_pose_recipe	Set up a pose recipe to save

3.4.2 set_save_target_pose

Up to 100 pose recipes are stored.

Data Types	Variables	Description
bool	send_target_pose_flag	Set whether to send data to the gripper
int	save_target_pose_recipe	Set up a pose recipe to save

3.4.3 set_load_pose_recipe

Up to 100 pose recipes are stored.

Data Types	Variables	Description
bool	send_load_pose_flag	Set whether to send data to the gripper

int	save_load_pose_recipe	Set a Pose Recipe to Load

3.4.4 set_target_joint

Data Types	Variables	Description
bool	send_target_joint_flag	Set whether to send data to the gripper
float[]	finger1_joint	Enter a target position value for finger 1
float[]	finger2_joint	Enter a target position value for finger 2
float[]	finger3_joint	Enter a target position value for finger 3
float[]	finger4_joint	Enter a target position value for finger 4
float[]	finger5_joint	Enter a target position value for finger 5

Note

* In DG-4F, set the base joint using the variable for finger 5.

3.4.5 set_motion_time

Set the time for the gripper to reach the target position.

This feature is only applicable in position control mode. However, the specified duration is not always guaranteed, as the actual movement time may vary depending on the gripper's configuration settings.

Data Types	Variables	Description
bool	send_joint_motion_time_flag	Set whether to send data to the gripper
int[]	finger1_motion_time	Set the duration for the finger 1
int[]	finger2_motion_time	Set the duration for the finger 2
int[]	finger3_motion_time	Set the duration for the finger 3
int[]	finger4_motion_time	Set the duration for the finger 4
int[]	finger5_motion_time	Set the duration for the finger 5

Note

* In DG-4F, set the base joint using the variable for finger 5.

3.5 GRASP

This component allows control of the grasp-related functions of the Delto Gripper.

The grab feature is currently available only for the DG-3F and DG-5F models.

3.5.1 set_grasp_on_off

Data Types	Variables	Description
bool	send_grasp_on_off_flag	Set whether to send data to the gripper
int	grasp_on_off	Grasp operation setting

3.5.2 set_save_current_grasp

Store up to 20 grasp recipes (1-20).

Data Types	Variables	Description
bool	send_save_current_grasp_flag	Set whether to send data to the gripper
int	current_grasp_recipe	Set the grasp recipe number to save

3.5.3 set_grasp_force

Sets the force (in N) with which the gripper grasps the object. Values that can be entered range from 0.1 to 20.0.

Data Types	Variables	Description
bool	send_grasp_force_flag	Set whether to send data to the gripper
int	grasp_force	Set grasp force

3.5.4 set _grasp_mode

Data Types	Variables	Description
bool	send_grasp_mode_flag	Set whether to send data to the gripper
int	grasp mode	Set the Grasp Mode (refer to the table below for details).

grasp mode

For detailed instructions and guidance, please refer to the user manual.

No	DG-3F	No	DG-5F
1	3 fingers toward the center	21	5 fingers toward the center
2	Fingers 1 and 2 follow each other	22	Fingers 1, 2, and 3 follow the center point
3	Fingers 1 and 3 follow each other	23	Finger 1 follows the center of fingers

			2 and 3
4	Fingers 2 and 3 follow each other	24	Fingers 1 and 2 follow each other
5	Finger 1 follows the center of fingers 2 and 3	25	Fingers 1 and 3 follow each other
6	The three fingers move as if they are gripping an object	26	Fingers 1 and 4 follow each other
7	-	27	Fingers 1 and 5 follow each other
8	-	28	Finger 1 follows the center of the other fingers
9	-	29	The five fingers move as if they are gripping an object
No	**DG-4F**	**No**	**DG-2F**
31	4 fingers toward the center	51	2 fingers toward the center
32	The center point of fingers 2 and 3 and the center point of fingers 1 and 4 follow each other.	52	The two fingers move as if they are gripping an object
33	The four fingers move as if they are gripping an object		
34	Finger 1 follows the center of the other fingers		
35	Finger 4 follows the center of the other fingers		
36	Fingers 1 and 2 follow each other		
37	Fingers 1 and 4 follow each other		
38	Fingers 3 and 4 follow each other		

3.5.5 set_grasp_option

Data Types	Variables	Description
bool	send_grasp_option_flag	Set whether to send data to the gripper
int	Grasp option	Setting Grasp Options ▢ 0: Disable grasp options ▢ 1: Parallel Grasp Option ▢ 2: Fixed tilt grasp option

3.5.6 set_position_control

Select a motor to hold the gripper in position when grasping.

Data Types	Variables	Description
bool	send_position_control_flag	Set whether to send data to the gripper
float[]	finger1_position_control	Positioning mode for the finger1
float[]	Finger2_position_control	Positioning mode for the finger2
float[]	Finger3_position_control	Positioning mode for the finger3
float[]	Finger4_position_control	Positioning mode for the finger4
float[]	Finger5_position_control	Positioning mode for the finger5

Note

* In DG-4F, set the base joint using the variable for finger 5.

3.6 Blend

Components that combines pose, gain, and time to run a sequential blend.
Up to 10 blend recipes are stored.
Up to 50 recipes are stored in a single blend recipe.

3.6.1 set_start_blend_number

Data Types	Variables	Description
bool	send_start_blend_flag	Set whether to send data to the gripper
int	start_blend_recipe_number	Set the blend recipe number to call

3.6.2 set_load_blend_number

If the size of each recipe is inconsistent or exceeds the recipe range, *input_error* is raised.

Data Types	Variables	Description
bool	send_save_blend_flag	Set whether to send data to the gripper
int	blend_recipe_number	Set the blend recipe number to

		save
int[]	pose_recipe_number_list	Pose recipe List
int[]	gain_recipe_number_list	Gain recipe list
int[]	blend_wait_time	Set the wait time after an action completes

3.7 Current

This component allows you to control the functions related to current control of the Delto Gripper.

3.7.1 set_current_control

Data Types	Variables	Description
bool	send_current_control_flag	Set whether to send data to the gripper.
float[]	current_control	Whether to control the current. ▢ 0: Current control mode OFF ▢ 1: Current control mode ON

3.7.2 set_target_current

Data Types	Variables	Description
bool	send_current_control_flag	Set whether to send data to the gripper.
float[]	finger1_current	Set the target current value for the finger1
float[]	finger2_current	Set the target current value for the finger2
float[]	finger3_current	Set the target current value for the finger3
float[]	finger4_current	Set the target current value for the finger4
float[]	finger5_current	Set the target current value for the finger5

Note

* In DG-4F, set the base joint using the variable for finger 5.

3.8 Setting

Components that control whether data is saved after a power down, and the values of gripper settings such as joint arrival range and joint offset.

3.8.1 set_joint_offset

Data Types	Variables	Description
bool	send_joint_offset_flag	Set whether to send data to the gripper
Float[]	joint_offset	Set Joint Offset

3.8.2 set_joint_inpose

Data Types	Variables	Description
bool	send_joint_inpose_flag	Set whether to send data to the gripper
int	joint_inpose	Setting the angle to check the reach of a gripper joint

3.8.3 set_teach_mode

Data Types	Variables	Description
bool	send_teach_mode_flag	Set whether to send data to the gripper
int	teach_mode	Change Teaching Mode ▢ 0: Teaching Mode OFF ▢ 1: Teaching Mode ON

3.8.4set_eeprom_write

Pose recipes and other Modbus-sent data must be stored in the EEPROM so that the set values are retained when power is cycled.

Data Types	Variables	Description
bool	send_eeprom_write_flag	Set whether to send data to the gripper
int	eeprom_write	Saving the EEPROM 1: Save EEPROM

3.9 Receive

Components that can see the gripper's status data

3.9.1 Receiving Gripper Status Data

This component is used for the purpose of checking the current state of the gripper. The gripper status information is read via Modbus each time the component is executed and stored in a Global Variable. The user can check the value of this Global Variable as often as needed to monitor the status of the gripper in real-time.

Data Types	Variables	Description
int	g_dg_moving	Whether the gripper is moving
int	g_dg_arrived_target_joint	Gripper Target Position Value Reached
int	g_dg_blend_status	Gripper Blend Status
int[]	g_dg_joint_velocity	Velocity data for 20 joints Unit: RPM
int[]	g_dg_joint_current	Current Data for 20 joints Unit: mA
float[]	g_dg_joint_position	Angle data for 20 joints Unit: degree
float[]	g_dg_joint_temperature	Temperature data from 20 joints Unit : °C
int	g_dg_gpio_input1	gripper input1

3.10 Output

This component allows control of the gripper's outputs.

Data Types	Variables	Description
bool	send_gpio_output_flag	Set whether to send data to the gripper
int	gpio_output1	Set gripper output1 0: 0V 1: 24V / 24mA
int	gpio_output2	Set gripper output2 0: 0V 1: 24V / 24mA
int	gpio_output3	Set gripper output3 0: 0V 1: 24V / 24mA


TESOLLO®