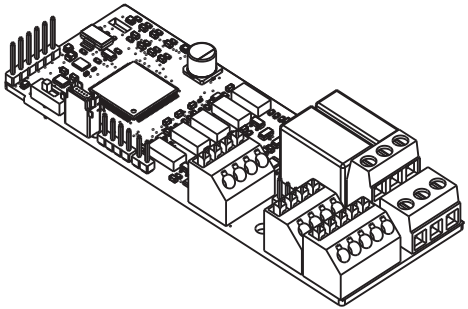
Operation Manual

Programmable Extension Card



SHENZHEN INVT ELECTRIC CO., LTD.

Safety precautions

The extension card can be installed and operated only by people who have taken part in professional training on electrical operation and safety knowledge, obtained the certification, and been familiar with all steps and requirements for installing, performing commissioning on, operating, and maintaining the device, and are capable of preventing all kinds of emergencies.

Before installing, removing, or operating the extension card, read the safety precautions described in this manual and the inverter operation manual carefully to ensure safe operation.

For any physical injuries or damage to the device caused due to your neglect of the safety precautions described in this manual and the inverter operation manual, our company shall not be held liable.

- You need to open the housing of the inverter when installing or removing the extension card. Therefore, you must disconnect all power supplies of the inverter and ensure that the voltage inside the inverter is safe. For details, see the description in the inverter operation manual. Severe physical injuries or even death may be caused if you do not follow the instructions.
- Store the extension card in a place that is dustproof and dampproof without electric shocks or mechanical pressure.
- The extension card is electrostatic sensitive. Take measurements to prevent electrostatic discharge when performing operations involving it.
- Tighten the screws up when installing the extension card. Ensure that it is firmly fixed and properly grounded.

Contents

1 Product confirmation

Perform the following checks after receiving the programmable extension card:

- Check whether the programmable extension card is damaged.
- Check whether the received extension card is the one you purchased based on the bar code and label on the printed circuit board (PCB).
- Check whether the following items are received:

  One programmable extension card, one tie wrap, one tie, one M3 screw, and one manual
- If the extension card is damaged, a wrong one is delivered, or some items are missed, contact the supplier in a timely manner.
- Contact INVT to obtain the device description file, library file, and configuration file of the programmable extension card. The device description file is in the .xml format, the library file is in the .library format, and the configuration file is *Gateway.cfg*.

Environmental requirements

Table 1-1 Environmental requirements

Item	Requirements
Operation temperature	-10–+50°C
Storage temperature	-20–+60°C
Relative humidity	5%–95%
Other climatic conditions	No condensation, freezing, rain, snow, or hail, etc. Solar radiation: less than 700 W/m^2
Air pressure	70–106 kPa
Vibration and impact	5.9 m/s^2 (0.6 g) at the sine vibration of 9 Hz to 200 Hz

2 Product overview

The programmable extension card can be used in combination with an inverter to replace a micro PLC in some applications. It integrates the CODESYS Runtime System developed by 3S-Smart Software Solutions (3S) on the bottom layer. You can perform secondary development through CODESYS on an upper computer. The secondary development supports multiple programming languages, including structured text (ST), adder diagram, function block, and instruction list. After the programming is complete, you can download the program to the programmable extension card for operation.

When the programmable extension card is used in combination with an inverter, all the input/output (I/O) terminals of the inverter for external connections can be invoked by the programmable extension card as peripherals, which increases the points of the extension card.

3 Secondary development platform

3.1 Introduction to the CODESYS development environment on an upper computer

The secondary development platform for the programmable extension card is the CODESYS programming software developed by 3S. You can download the CODESYS software on the official website (https://www.codesys.com/). Currently, INVT uses CODESYS V3.5 SP6 for development, and it is recommended that you use V3.5 SP6 or later version for development.

After installing the CODESYS programming software on an upper computer, open the software, and then install *Devdesc_v100.xml*, the device description file developed for INVT, as described in the following:

1. Open CODESYS V3.5 SP6, and choose **Tools** -> **Device Repository**, as shown in Figure 3-1. If **Device Repository** is not displayed, you can choose **Options** to set it to be displayed.

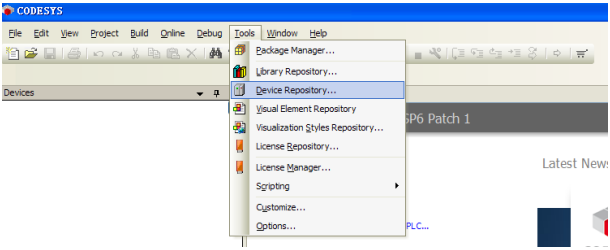


Figure 3-1

2. Click Install to install the device description file *Devdesc_v100.xml*, as shown in Figure 3-2. Select **All supported description files (*.xml;*.eds;*.dcf;*.gs?)** in **File Type** when you choose the file to be installed.

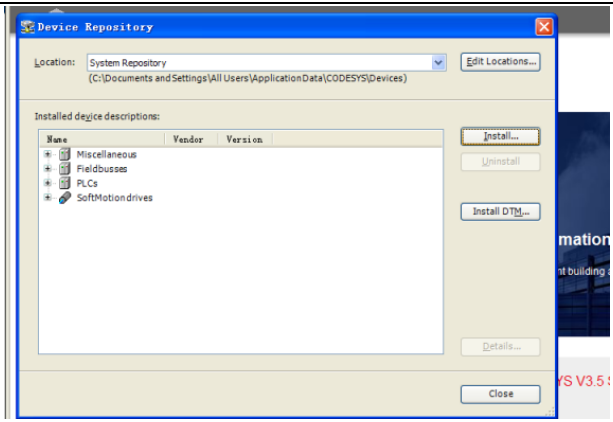


Figure 3-2

3. After the device description file is successfully installed, the programmable extension card for INVT is displayed, as shown in Figure 3-3. The version is V3.5.5.10.

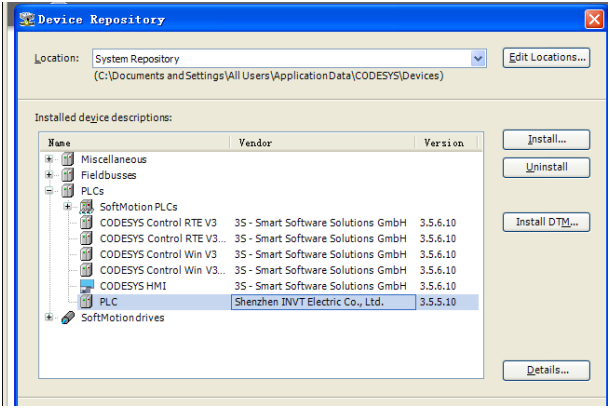


Figure 3-3

3.2 PLC program download interface

Programmable logic controller (PLC) programs are downloaded through the serial port of the programmable extension card. The programmable extension card integrates the USB-serial port conversion circuit (MiniUSB interface). You can use a standard USB cable to connect the extension card to a PC. The USB-serial port conversion driver for the extension card can be downloaded in the following address: http://www.wch.cn/downloads/CH341SER_EXE.html; CH341SER.EXE.

In practice, the CODESYS programming software on the upper computer communicates with the programmable extension card by using Gateway. The serial communication mode is adopted, and the default baud rate is 115,200. Therefore, you need to replace the *Gateway.cfg* file in the **GatewayPLC** folder with the *Gateway.cfg* file provided by INVT. The path of the **GatewayPLC** folder is C:\Program Files\3S CODESYS\GatewayPLC in this case, and it varies according to the installation path of CODESYS. Then, you need to open the *Gateway.cfg* file through Notepad (as shown in Figure 3-4) and modify the information based on the COM port currently used on the PC. If COM1 is used by the PC, set **Com.0.Port=1**. After the port is changed, you need to restart Gateway.

```
[CmpBlkDrvCom]
;Example for a configuration of this block dri
;EnableAutoAddressing=1 should be set to activ
;Mainnet-Mainnet-Communication on serial line.
Com.0.Port=1
Com.0.Name=MyCom
Com.0.Baudrate=115200
Com.0.EnableAutoAddressing=0
Com.0.HalfDuplexAutoNegotiate=1
Com.0.EnableRtsToggleHandshake=1
Com.0.LocalAddress=2
```

Figure 3-4

3.3 Instructions for CODESYS

1. After opening CODESYS, choose **File** -> **New Project** -> **Standard Project** to create a project, click **OK**, and choose the PLC device for INVT in the dialog box that appears, as shown in Figure 3-5.

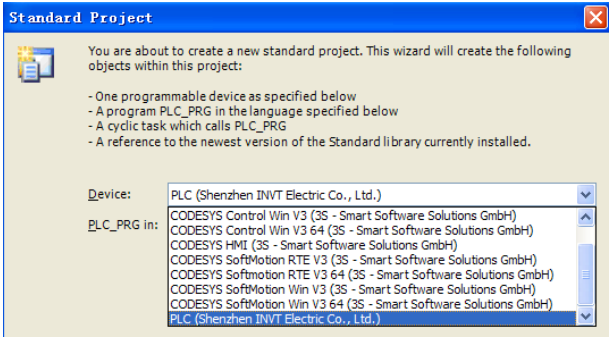


Figure 3-5

CODESYS supports multiple programming languages, including ST, ladder diagram, function block, and instruction list. You can choose a programming language based on your own programming habits, as shown in Figure 3-6.

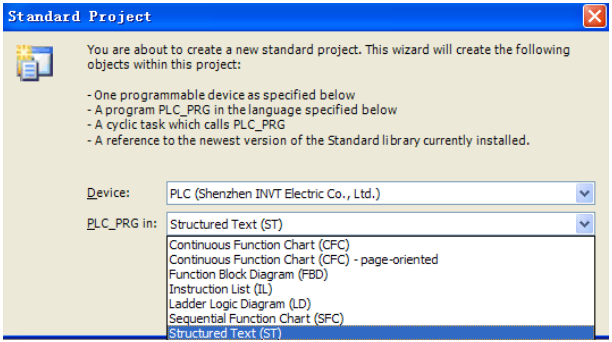


Figure 3-6

2. After compiling and checking the PLC program, click **Scan network** on the **Device** tab, as shown in Figure 3-7.

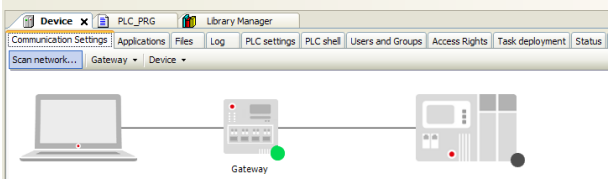


Figure 3-7

If the programmable extension card has been powered on and properly connected, Gateway of CODESYS on the upper computer can find the CODESYS Runtime System on the bottom layer of the programmable extension card. Choose **MyDevice** and click **OK**, as shown in Figure 3-8.

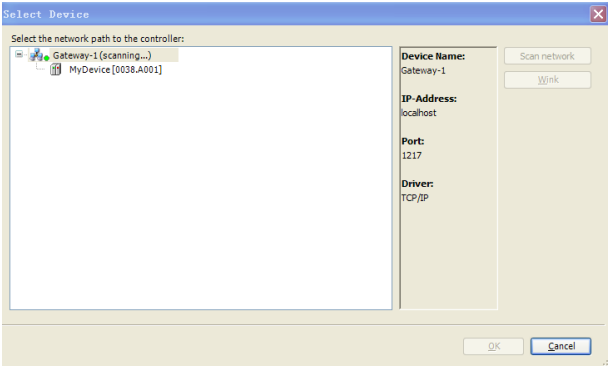


Figure 3-8

After the device is properly connected, information about the PLC device is displayed, as shown in Figure 3-9.

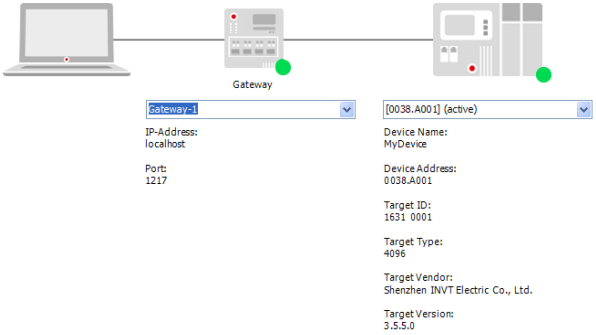

Gateway
Gateway-1
IP-Address:
localhost
Port:
1217
[0038.A001] (active)
Device Name:
MyDevice
Device Address:
0038.A001
Target ID:
1631 0001
Target Type:
4096
Target Vendor:
Shenzhen INVT Electric Co., Ltd.
Target Version:
3.5.5.0

Figure 3-9

3. Click LOGIN/LOGOUT to download the PLC program, as shown in Figure 3-10.

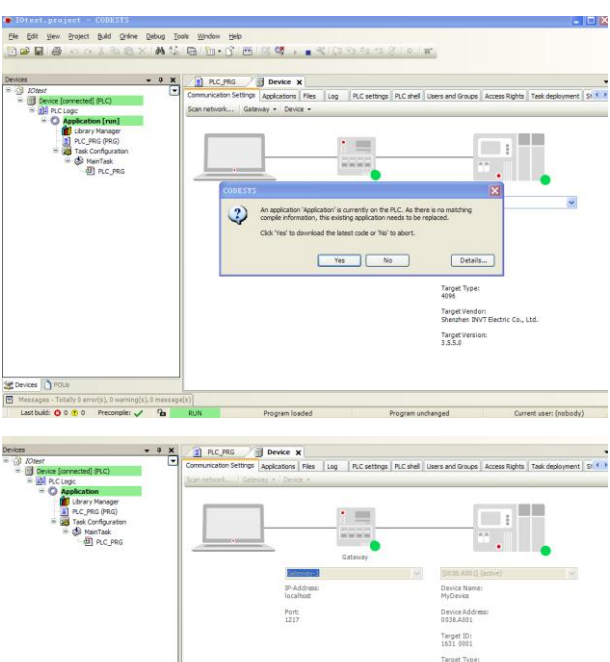


Figure 3-10

4. Download the PLC program to flash memory or random-access memory (RAM).

The underlying driver supports the download of the PLC program to the flash memory or RAM. You can set the download on the **Boot application** tab in the **Properties** dialog box. As shown in Figure 3-11, if the first item Create implicit boot application on download is selected, the PLC program is downloaded to the flash memory, and if it is unselected,

the PLC program is downloaded to RAM.

Note: If the PLC program is downloaded to RAM, the data is not saved at power outage.

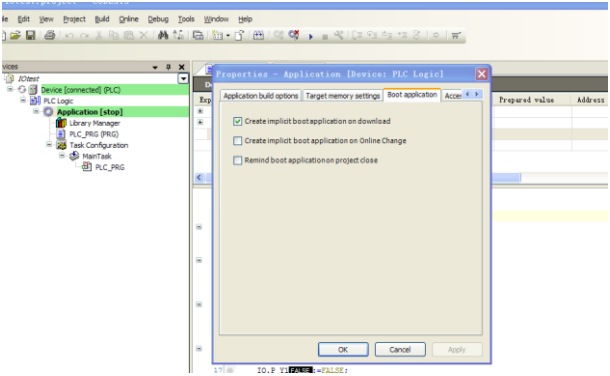


Figure 3-11

After the PLC program is downloaded, you can click **START** (the **RUN/STOP** DIP switch needs to be turned to RUN) to perform online commissioning or data monitoring. Online commissioning supports break-point execution, single-step execution, running to the specified step, etc., as shown in Figure 3-12.

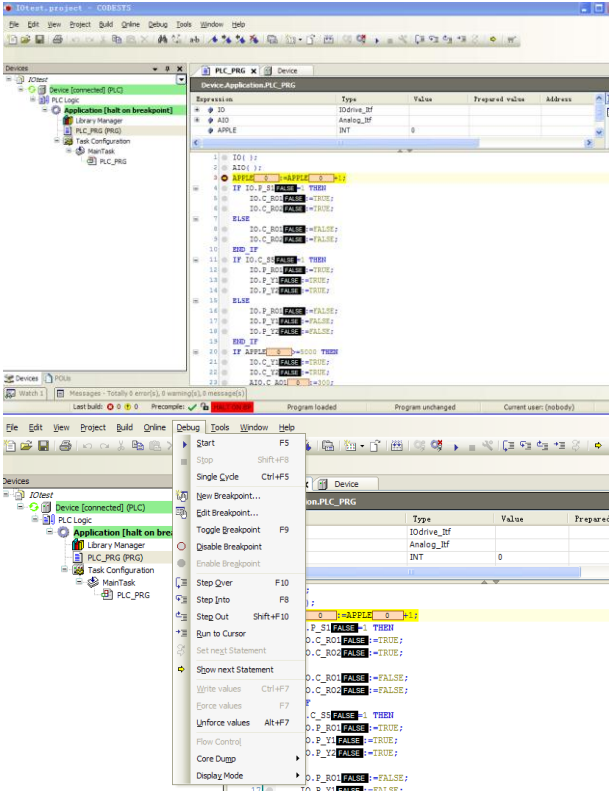


Figure 3-12

During commissioning, you can also use forced values or preset values, as shown in Figure 3-13, which makes the commissioning more flexible.

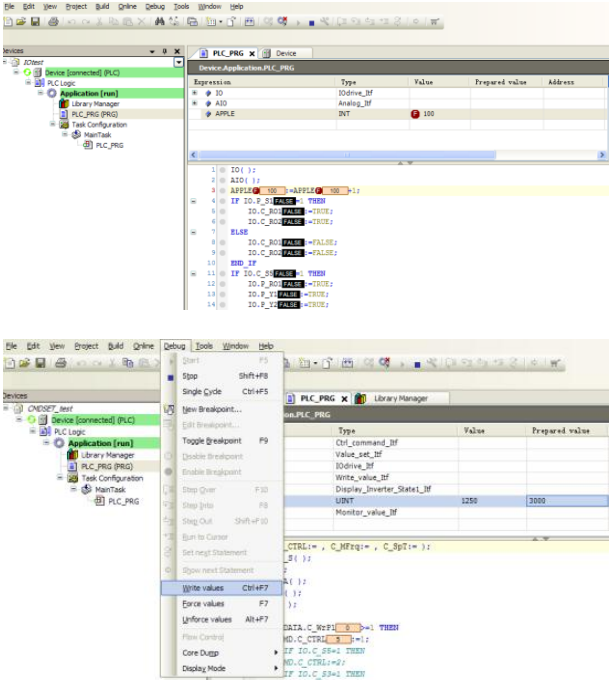


Figure 3-13

If you want to secure part of the PLC program or some algorithms in secret, you can compile and encapsulate library functions as required and then provide the programming interfaces (data I/O interfaces) of the library functions for invocation. The library functions mentioned here refer to the library developed by users and are not related to the underlying programmable extension card or inverter. The library functions described in chapter 4 are in another format, different from those described here, and are involved in the underlying driver. They can be compiled and encapsulated only by developers of the driver.

4 CODESYS programming interfaces and definitions

The programmable extension card integrates the PLC secondary development function of the CODESYS. Besides the built-in PLC function parameters, the input parameters required by CODESYS programming also include hardware parameters of the extension card and hardware and software parameters of the inverter. The output of the CODESYS programming can not only used by the internal PLC, but can also be used to control the hardware output interfaces of the extension card and inverter and to control the start and stop commands for the inverter.

This chapter describes the relationship between the interfaces for the CODESYS secondary development and those (including the hardware and software instruction interfaces) of programmable extension card and inverter. The following describes the CODESYS programming interfaces in details, and the functions involved with an inverter can be used only when the programmable extension card is enabled for the inverter. You can enable or disable the programmable extension card for the inverter through the function code P27.00, as described in the following table.

Function code	Name	Detailed parameter description	Setting range	Set value
P27.00	Programmable extension card enabling	This parameter is used to enable or disable the programmable extension card. 0: Disabled 1: Enabled The programming interfaces and data involved with the inverter are valid only after the programmable extension card is enabled. If it is disabled, the interfaces and data cannot be used.	0–1	0

4.1 CODESYS programming interface function library installation and invocation

All the CODESYS programming interfaces involved with the programmable extension card or inverter are provided in the format of an external encapsulated library. Therefore, before using the interfaces, you need to install the encapsulated library *CmpINVT_V100.library*, which is the function library developed for INVT's programmable extension cards. It includes all programming interfaces involved with the programmable extension card and inverter.

Note: The name of the library varies according to product model. For example, the name of

the library for the GD350 series is *CmpGD350_PLC.library*.

The library functions are installed and invoked as described in the following:

1. Open CODESYS V3.5 SP6, and choose **Tools** -> **Device Repository**, as shown in Figure 4-1. If **Library Repository** is not displayed, you can choose **Options** to set it to be displayed.

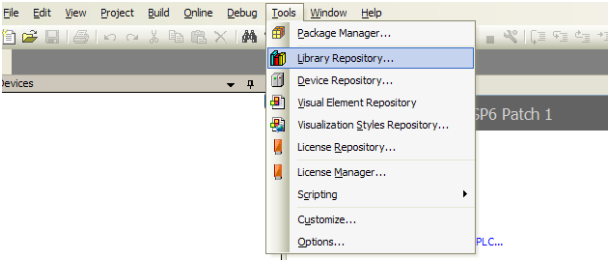


Figure 4-1

2. Click **Install** to install the library file *CmpINVT_V100.library*, as shown in Figure 4-2. Select **Library files** during the installation in the dialog box for choosing the file to be installed.

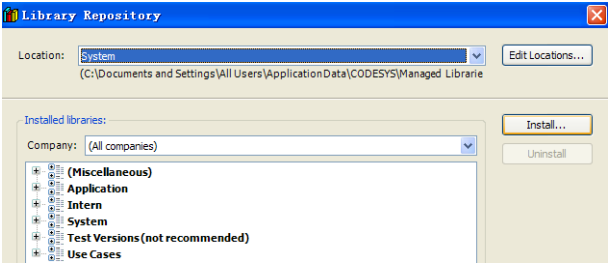


Figure 4-2

Similarly, you can uninstall a function library. Click the function library to display the version information, and then click Uninstall to uninstall it, as shown in Figure 4-3.

After being installed, the function library can be used only after it is invoked (added).

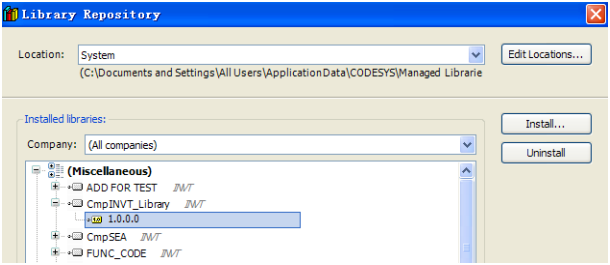


Figure 4-3

3. Double-click **Library Manager**, and then click **Add library**, as shown in Figure 4-4

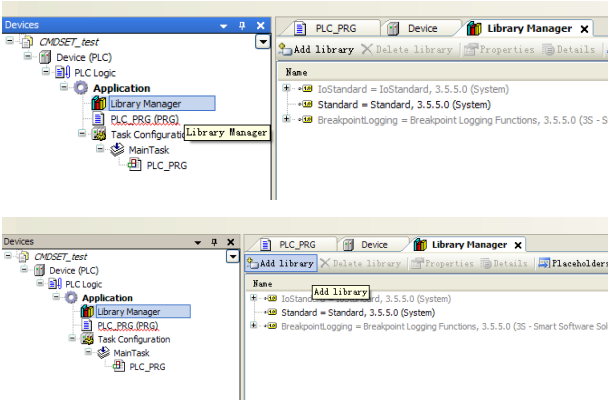


Figure 4-4

4. Choose the function library **CmpINVT_Library** and click **OK**, as shown in Figure 4-5.

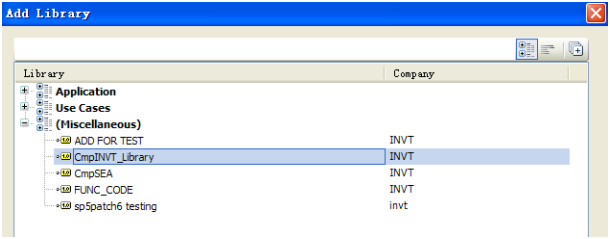


Figure 4-5

After being added, the function library and its version information are displayed on the **Library Manager** tab, as shown in Figure 4-6. Similarly, you can delete the function library by clicking **Delete library** if it is not required.

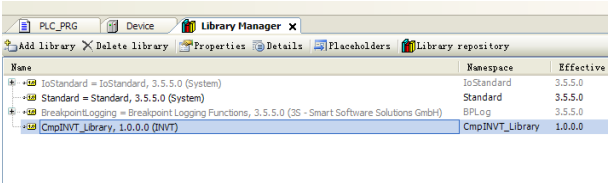


Figure 4-6

5. After a function library is added, you can view information contained in the library, including multiple functions and their interfaces.

Note:

After a project is created, you need to install and invoke the library developed for INVT. Besides, the system may invoke some standard library files. The standard library version varies according to CODESYS software version. If the standard library version of the software is not consistent with that described in the device description file, an error may be reported, indicating that the library fails to be loaded, and errors may occur in program compiling. On the Library Manager tab of the CODESYS software, a download button is provided, which allows you to download the missed standard library. You need only to connect the software to the Internet and download the library, as shown in Figure 4-7.

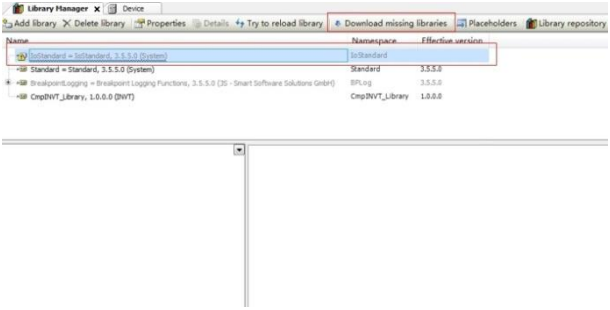


Figure 4-7

4.2 Digital I/O programming interfaces

In the current plan, the programmable extension card supports its built-in I/O points, and it can use the I/O points (the number of I/O points varies according to product) of an inverter when it is inserted on the control board of the inverter. For virtual I/O points or intermediate relays, you can define them on CODESYS and use them as required.

Use the GD350 inverter as an example. The following table describes digital I/O points of the inverter and programmable extension card and their corresponding interfaces. In the table, C indicates the control board or terminal block of the inverter, and P indicates the programmable extension card.

Name on CODESYS (Varying according to development)	Hardware interface (bool type)
C_S1	Corresponding to input point S1 of the inverter control board
C_S2	Corresponding to input point S2 of the inverter control board
C_S3	Corresponding to input point S3 of the inverter control board
C_S4	Corresponding to input point S4 of the inverter control board
C_S5	Corresponding to input point S5 of the I/O extension card for the inverter control board
C_S6	Corresponding to input point S6 of the I/O extension card for the inverter control board

Name on CODESYS (Varying according to development)	Hardware interface (bool type)
C_S7	Corresponding to input point S7 of the I/O extension card for the inverter control board
C_S8	Corresponding to input point S8 of the I/O extension card for the inverter control board
C_S_HDIA	Corresponding to input point HDIA (for digital input) of the inverter control board
C_S_HDIB	Corresponding to input point HDIB (for digital input) of the inverter control board
P_S1	Corresponding to input point PS1 of the programmable extension card
P_S2	Corresponding to input point PS2 of the programmable extension card
P_S3	Corresponding to input point PS3 of the programmable extension card
P_S4	Corresponding to input point PS4 of the programmable extension card
P_S5	Corresponding to input point PS5 of the programmable extension card
P_S6	Corresponding to input point PS6 of the programmable extension card
C_Y1	Corresponding to output point Y1 of the inverter control board
C_Y2	Corresponding to output point Y2 of the I/O extension card for the inverter control board
C_RO1	Corresponding to output point RO1 of the inverter control board
C_RO2	Corresponding to output point RO2 of the inverter control board
C_RO3	Corresponding to output point RO3 of the I/O extension card for the inverter control board
C_RO4	Corresponding to output point RO4 of the I/O extension card for the inverter control board
C_Y_HDO	Corresponding to output point HDO (for digital input) of the inverter control board
P_Y1	Corresponding to output point PY1 of the programmable extension card
P_Y2	Corresponding to output point PY2 of the programmable extension card

Name on CODESYS (Varying according to development)	Hardware interface (bool type)
P_RO1	Corresponding to output point PRO1 of the programmable extension card
P_RO2	Corresponding to output point PRO2 of the programmable extension card

After being compiled, the underlying driver code for I/O programming interfaces is encapsulated into an external library and provided directly for users to invoke in the CODESYS programming environment on an upper computer. You can use the library as required and define the output functions.

4.3 Analog I/O programming interfaces

The control board of an inverter and the programmable extension card provide analog input and output for CODESYS programming. The number of analog I/O programming interfaces varies according to product.

Use the GD350 inverter as an example. The following table describes analog I/O points of the inverter and programmable extension card and their corresponding interfaces. In the table, C indicates the control board or terminal block of the inverter.

Name on CODESYS	Function description	Range	Attribute
C_AI1	Analog input AI1 of the inverter control board	0.00–10.00V/0.00–20.00mA	Read only
C_AI2	Analog input AI2 of the inverter control board	-10.00–10.00V	Read only
C_AI3	Analog input AI3 of the I/O extension card for the inverter control board	0.00–10.00V/0.00–20.00mA	Read only
C_AO1	Analog output AO1 of the inverter	0.00–10.00V/0.00–20.00mA	Writable Range: 0–1000
C_AO2	Analog output AO2 of the I/O extension card for the inverter control board	0.00–10.00V/0.00–20.00mA	Writable Range: 0–1000

Note: The values -1000 to +1000 read or input correspond to the actual values -10.00V to +10.00 V.

4.4 Control command and function instruction programming interfaces

Besides the digital and analog I/O terminals that are provided by CODESYS for users, you can also use communication commands to control the operation of the inverter. Compared to terminal functions, communication commands are not as abundant. Communication commands are mainly used to control the start and stop of the inverter. The following describes the control commands provided by CODESYS for inverters:

1. Basic control commands

CODESYS control commands are communication control commands. Therefore, you need to set P00.01 to 2 (communication) and P00.02 to 4 (programmable extension card).

Name on CODESYS: C_CTRL	
Value	**Output function description**
0	Disabled
1	Forward running (FWD)
2	Reverse running (REV)
3	Forward inching
4	Reverse inching
5	Decelerating to stop
6	Coasting to stop
7	Fault reset
8	Fast emergency stop
9	Reserved
10	Reserved

When using control commands, note that: When being powered on (initialization is not complete) or during restart, the system enters the running protection state if it receives a running command, and the inverter does not run until the running command is cancelled (running protection can be stopped by stopping the system) and another running command is received.

To prevent the system from entering the running protection state at power-on, it is recommended that you wait 500 ms to 1000 ms after the system is powered on and then send an inverter running command.

Restart refers to that the RUN/STOP dual in-line package (DIP) switch on the PLC card is turned to STOP and then to RUN or the START/STOP button on the CODESYS monitoring software on the upper computer is clicked to stop and clicked again to start when the inverter is running. When this happens, the system enters the running protection state if the running command is not cancelled.

Generally, when the system is running properly, we do not turn the RUN/STOP DIP switch on the PLC card to STOP or click the STOP button on the on the CODESYS monitoring software

on the upper computer. When performing commissioning on the code, however, you may stop the system and start it again. When this happens, you need to pay attention to the scenarios where running protection may occur.

2. Multi-step speed running command

Name on CODESYS: C_MFrq	
Value	**Output function description**
0	Disabled
1	Multi-step speed 0
2	Multi-step speed 1
3	Multi-step speed 2
4	Multi-step speed 3
5	Multi-step speed 4
6	Multi-step speed 5
7	Multi-step speed 6
8	Multi-step speed 7
9	Multi-step speed 8
10	Multi-step speed 9
11	Multi-step speed 10
12	Multi-step speed 11
13	Multi-step speed 12
14	Multi-step speed 13
15	Multi-step speed 14
16	Multi-step speed 15

If the value is greater than 16, the command is invalid. When the value is 0, the command is disabled. The default value is 1. Note that only channels are set through multi-step speed running commands. The speed in each step needs to be set through function codes of the P10 group. Additionally, you need to set P00.06/P00.07 (frequency source).

3. ACC/DEC time setting command

Name on CODESYS: C_SpT	
Value	**Output function description**
0	ACC/DEC time 1
1	ACC/DEC time 2
2	ACC/DEC time 3
3	ACC/DEC time 4

If the value is greater than 3, the command is invalid. The default value is 0. Note that only channels are set ACC/DEC time setting commands. The specific ACC/DEC time needs to be set through function codes.

The three commands mentioned in the preceding description share one function block function.

4.5 Function parameter setting programming interfaces

The inverter provides some function parameter setting interfaces for the programmable extension card. You can set these parameters through CODESYS on an upper computer.

Name on CODESYS	Output function description
FRQ_SET	Frequency set through CODESYS (0–F_{max}, unit: 0.01 Hz)
TOR_SET	Torque setting (-3000–+3000, among which 1000 corresponds to 100.0%)
F_UP_FWD	Upper frequency setting limit for forward running (0–F_{max}, unit: 0.01 Hz)
F_UP_REV	Upper frequency setting limit for reverse running (0–F_{max}, unit: 0.01 Hz)
T_UP_ELEC	Upper limit of electromotive torque (0–3000, among which 1000 corresponds to 100.0% of the rated current of the motor)
T_UP_GEN	Upper limit of brake torque (0–3000, among which 1000 corresponds to 100.0% of the rated current of the motor)
ACC1	ACC time 1 set through CODESYS (0–36000, unit: 0.1s)
DEC1	DEC time 1 set through CODESYS (0–36000, unit: 0.1s)
VF_SET	V/F voltage reference (0–1000, corresponding to 0.0%–100.0%)
PID_GV_S	PID reference (-1000–+1000, corresponding to -100.0%–+100.0%)
PID_FB_S	PID feedback (-1000–+1000, corresponding to -100.0%–+100.0%)

Note:

1. When setting parameters, you need to set the corresponding function, except the ACC/DEC time 1, to CODESYS setting first. (Example: When setting the frequency, you need to set the frequency source of P00.06/P00.07 to 13, which indicates setting through CODESYS programming.)

4.6 Function parameter and inverter state viewing interfaces

1. Function parameter viewing (read-only)

The inverter provides function parameters of the P17 and P18 groups for the programmable extension card, and CODESYS can directly invoke these parameters through the external library to view the function parameters of the inverter (function parameter interfaces vary according to product).

Use the GD350 inverter as an example. The following table describes the function parameters of the inverter and their corresponding interfaces. For details about the function parameters, see the GD350 inverter operation manual.

Function code of the inverter	Name on CODESYS	Function code read (read-only)
P17 group		
P17.00	Set_frq	Set frequency
P17.01	Output_frq	Output frequency
P17.02	Lineset_frq	Ramp frequency reference
P17.03	Output_volt	Output voltage
P17.04	Output_current	Output current
P17.05	Motor_rpm	Rotating speed of the motor
P17.06	Tor_current	Torque current
P17.07	Mag_current	Excitation current
P17.08	Motor_power	Power of the motor
P17.09	Output_tor	Output torque of the motor
P17.10	Est_motorfrq	Estimated frequency of the motor
P17.11	DC_volt	DC bus voltage
P17.12	Di_State	Digital input terminal state
P17.13	Do_State	Digital output terminal state
P17.15	Torgue_set	Torque reference
P17.16	Line_speed	Linear speed
P17.20	HDIA_frq	HDIA input frequency
P17.21	HDIB_frq	HDIB input frequency
P17.23	PID_set	PID reference
P17.24	PID_feedback	PID feedback
P17.25	Power_factor	Power factor of the motor
P17.26	Cur_runtime	Period of current operation
P17.27	Cur_stage	Simple PLC and current multi-step speed
P17.29	Syn_mac_angle	Magnetic pole angle of the synchronous machine
P17.33	Mac_c_set	Excitation current reference
P17.34	Tor_c_set	Torque current reference
P17.35	LineAC_c	AC incoming current
P17.41	Tor_up_elec	Upper limit of electromotive torque
P17.42	Tor_up_gen	Upper limit of brake torque
P17.43	Frq_up_fwd	Upper frequency limit for forward running in the torque control mode

Function code of the inverter	Name on CODESYS	Function code read (read-only)
P17.44	Frq_up_rev	Upper frequency limit for reverse running in the torque control mode
P18 group		
P18.00	Encoder_frq	Detected encoder frequency

2. Temperature and fault display viewing (read-only)

The inverter provides temperature, power level, and fault information for the programmable extension card. You use CODESYS to invoke the external library to view the information (temperature and fault viewing interfaces vary according to product).

Use the GD350 inverter as an example. The following table describes the inverter temperature and fault viewing interfaces. For details about the function parameters, see the GD350 inverter operation manual.

Function code of the inverter	Name on CODESYS	Function code read (read-only)
P07.11	T_sink	Bridge rectifier module temperature
P07.12	T_invert	Inverter module temperature
P07.14	Invert_runtime	Running time
P07.19	Electric_high	MSB of power consumption of the inverter
P07.20	Electric_low	LSB of power consumption of the inverter
P07.27	Fault_type	Type of current fault
P07.33	Fault_runfrq	Running frequency at current fault
P07.34	Fault_linefrq	Ramp frequency reference at current fault
P07.35	Fault_outvolt	Output voltage at current fault
P07.36	Fault_outcurrent	Output current at current fault
P07.37	Fault_dc_volt	Bus voltage at current fault
P07.38	Fault_temper	Highest temperature at current fault
P07.39	Fault_di	Input terminal state at current fault
P07.40	Fault_do	Output terminal state at current fault

3. Current inverter state and parameter viewing (read-only)

The inverter provides the current states and some parameters for CODESYS to invoke. Some of the parameters are the same as those in the P17 group.

Name on CODESYS	Parameter read (read-only)

Name on CODESYS	Parameter read (read-only)
C_state1	Inverter state 1 1: In forward running 2: In reverse running 3: Stopped 4: Faulty 5: PoFF
C_state2	Inverter state 2 Bit0: =0: Not ready to run, =1: Ready to run Bit1–Bit2: =00: Motor 1, =01: Motor 2, =10: Motor 3, =11: Motor 4 Bit3: =0: Asynchronous machine, =1: Synchronous machine Bit4: =0: No overload pre-alarm generated, =1: Overload pre-alarm generated Bit5–Bit6: =00: Keypad-based control, =01: Terminal-based control, =10: Communication-based control Bit7: Reserved Bit8: =0: Speed control mode, =1: Torque control mode Bit9: =0: Non-position control mode, =1: Position control mode Bit10–Bit11: =00: Vector control without PG 0, =01: Vector control without PG 1, =10: Closed-loop vector control mode, =11: V/F control Bit12–Bit15: Reserved
C_err_code	Fault code of the inverter

4.7 User-defined PLC faults (10 faults in the current design)

The programmable extension card provides 10 user-defined faults for users. You can use them during secondary development through CODESYS. The user-defined faults are also transmitted back to the inverter and displayed on the panel of the inverter as external faults. The symbol for the user-defined external faults is P_Err (the display of DSP fault codes is updated accordingly). The following table describes the corresponding fault codes on the inverter.

Fault name on CODESYS: P_Err		
Fault code displayed on the inverter	**P_Err fault number**	**Description**
/	0	No fault (deleting faults)
P-E1	1	User-defined PLC fault 1
P-E2	2	User-defined PLC fault 2
P-E3	3	User-defined PLC fault 3

Fault name on CODESYS: P_Err		
Fault code displayed on the inverter	**P_Err fault number**	**Description**
P-E4	4	User-defined PLC fault 4
P-E5	5	User-defined PLC fault 5
P-E6	6	User-defined PLC fault 6
P-E7	7	User-defined PLC fault 7
P-E8	8	User-defined PLC fault 8
P-E9	9	User-defined PLC fault 9
P-E10	10	User-defined PLC fault 10

The setting range of the P_Err fault number is 1 to 10. A fault number greater than 10 is invalid.

After a fault occurs, the inverter stops running. To run the inverter, you need to perform fault reset first (the same principle as for inverter faults). During programming, note that: If the conditions for fault reset are not met, you need to delete the fault in a timely manner. Otherwise, the fault persists. In addition, the inverter can display only one fault. If a fault has occurred, other faults are not reported. You can set a restriction through PLC programming (after a fault is reported, the inverter does not respond to other detected faults until the current fault is deleted and reset).

4.8 Programming interfaces for the data writing parameter group

The inverter provides a group of function parameters for users to write data to the programmable extension card through the inverter.

You can write data to the parameters on the programmable extension card by setting corresponding parameters in the P27 group on the inverter (that is, you can invoke the values of the library functions of the P27 group through CODESYS to assign values to the corresponding variables). You can define 10 parameters.

Function code of the inverter	Name on CODESYS	Parameter description
P27.00	/	Programmable extension card enabling 0: Disabled; 1: Enabled
P27.01	C_WrP1	Parameter value written to WrP1 of the PLC
P27.02	C_WrP2	Parameter value written to WrP2 of the PLC
P27.03	C_WrP3	Parameter value written to WrP3 of the PLC
P27.04	C_WrP4	Parameter value written to WrP4 of the PLC
P27.05	C_WrP5	Parameter value written to WrP5 of the PLC
P27.06	C_WrP6	Parameter value written to WrP6 of the PLC
P27.07	C_WrP7	Parameter value written to WrP7 of the PLC

Function code of the inverter	Name on CODESYS	Parameter description
P27.08	C_WrP8	Parameter value written to WrP8 of the PLC
P27.09	C_WrP9	Parameter value written to WrP9 of the PLC
P27.10	C_WrP10	Parameter value written to WrP10 of the PLC

4.9 Programming interfaces for the monitoring parameter group

The inverter provides a group of parameters for the programmable extension card to monitor PLC data.

You can include the to-be-monitored PLC parameters in the function library of the P27 group during the secondary development through CODESYS on the upper computer. That is, you can monitor the variables of the PLC by using the P27 parameter group of the inverter (reading data of the PLC and displaying it in function codes). You can define 10 variables.

Function code of the inverter	Name on CODESYS	Parameter description
P27.11	/	Running state of CODESYS 0: Stopped; 1: Running
P27.12	C_MoP1	Monitoring (viewing) the value of MoP1 of the PLC
P27.13	C_MoP2	Monitoring (viewing) the value of MoP2 of the PLC
P27.14	C_MoP3	Monitoring (viewing) the value of MoP3 of the PLC
P27.15	C_MoP4	Monitoring (viewing) the value of MoP4 of the PLC
P27.16	C_MoP5	Monitoring (viewing) the value of MoP5 of the PLC
P27.17	C_MoP6	Monitoring (viewing) the value of MoP6 of the PLC
P27.18	C_MoP7	Monitoring (viewing) the value of MoP7 of the PLC
P27.19	C_MoP8	Monitoring (viewing) the value of MoP8 of the PLC
P27.20	C_MoP9	Monitoring (viewing) the value of MoP9 of the PLC
P27.21	C_MoP10	Monitoring (viewing) the value of MoP10 of the PLC

The RUN/STOP DIP switch on the programmable extension card is mainly used to start and stop a PLC program. If the switch is turned to STOP, the running of the PLC program is

stopped; and if the switch is turned to RUN, the PLC program is run. The running state can be viewed through P27.11. If the state of the PLC program is switched from RUN to STOP, the control command sent to the inverter is deleted (that is, if the inverter is in the running state, it is stopped), and hardware output is also deleted (including the AO1, AO2, RO1, RO2, Y1, and Y2 output of the inverter and the PRO1, PY1, PY2, and external fault output of the programmable extension card). If the state of the PLC program is switched from STOP to RUN, PLC output takes effect in 0.5s. For details, see the following description.

4.10 RUN/STOP DIP switch description

The RUN/STOP DIP switch on the programmable extension card is mainly used to start and stop a PLC program, and is not involved with the download of the PLC program.

SW7 switch on the programmable extension card	Function	Detailed description
RUN	Running CODESYS	It is used to run the PLC program. If the state of the PLC program is switched from STOP to RUN, PLC output takes effect in 0.5s.
STOP	Stopping CODESYS	It is used to stop the PLC program and delete PLC output.

PLC output refers to the control command the PLC program sends to the inverter.

Hardware output includes the AO1, AO2, RO1, RO2, Y1, and Y2 output of the inverter and the PRO1, PY1, PY2, and external fault output of the programmable extension card.

If you turn the RUN/STOP DIP switch on the programmable extension card to STOP (when this happens, if CODESYS is used on an upper computer for monitoring, the START/STOP button on the upper computer is automatically turned to STOP) or click STOP of the START/STOP button when you are performing online monitoring or commissioning through CODESYS on an upper computer, the PLC program is stopped (is suspended), PLC output is deleted, and the inverter coasts to stop.

If you turn the RUN/STOP DIP switch on the programmable extension card to RUN (when this happens, if CODESYS is used on an upper computer for monitoring, the START/STOP button on the upper computer is automatically turned to START), the PLC program is started (continues to run), but the hardware output of the PLC program takes effect in 0.5s. In this case, if another running command is sent to the inverter and it does not enters the running protection state, the running command takes effect in 0.5s.

Note that: If the RUN/STOP DIP switch on the programmable extension card is always in the STOP state and you only click START of the START/STOP button on the CODESYS on an upper computer to run the PLC program, the PLC program is running properly, but PLC output is invalid. The PLC output takes effect only after you turn the RUN/STOP DIP switch to RUN.

In conclusion, the proper operation steps are described as follows:

When downloading a PLC program through CODESYS on an upper computer, you need to turn the RUN/STOP DIP switch to STOP first. After the PLC program is downloaded, you need to turn the RUN/STOP DIP switch to RUN, then you can perform online code commissioning and monitoring or enable/disable PLC output by clicking the START/STOP button on CODESYS on the upper computer.

5 Programming examples

5.1 Example 1

A customer requires the following:

When there are input signals on terminal PS1 on the programmable extension card, RO1 of the inverter outputs signals; and when there are input signals on terminals S2 and S3 of the inverter, RO2 of the inverter outputs signals.

Implementation method: The ladder diagram language is used for programming, and the library CmpINVT_Library needs to be added in the new project.

Figure 5-1 shows the code.

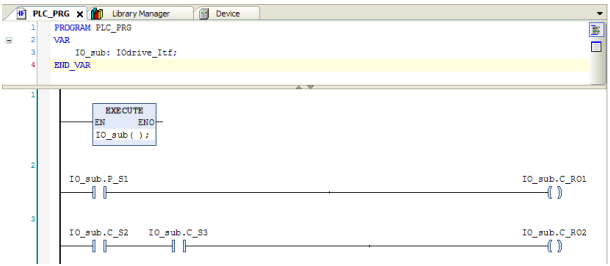


Figure 5-1

Note:

Library functions need to be declared and defined before they are used. Therefore, an EXECUTE module is inserted in the beginning of the code to define and declare the function IO_sub (). The type of this function is IOdrive_Itf (including all I/O programming interfaces for the programmable extension card and inverter). In this case, you can directly use the prefix IO_sub. to invoke the I/O terminals, such as IO_sub.P_S1 and IO_sub.C_S2 in Figure 5-1. The same principle applies for functions in other examples. You can press F2 to quickly add the definition and declaration of a function.

Figure 5-2 shows the online commissioning and monitoring results.

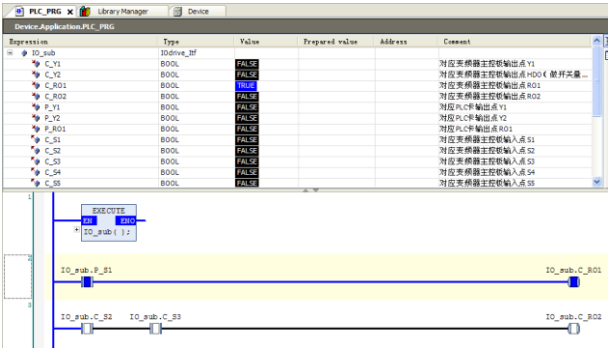


Figure 5-2

5.2 Example 2

A customer requires the following:

When there are input signals on terminal S5 of the inverter, the inverter runs forward; when there are input signals on terminal S6, the inverter runs reversely; when a fault occurs, fault reset can be performed through S7 input; and when there are input signals on terminal S8, PLC external fault 5 is reported.

Implementation method: The ST language is used for programming, and the library CmpINVT_Library needs to be added in the new project. In this example, multiple functions of the CMD_control.C_CTRL command are used. Therefore, you need to pay attention to the priorities to prevent an assigned value from being replaced by a subsequently assigned one. It is recommended that you use the statements of "IF" and "ELSIF".

Figure 5-3 shows the online commissioning and monitoring results.

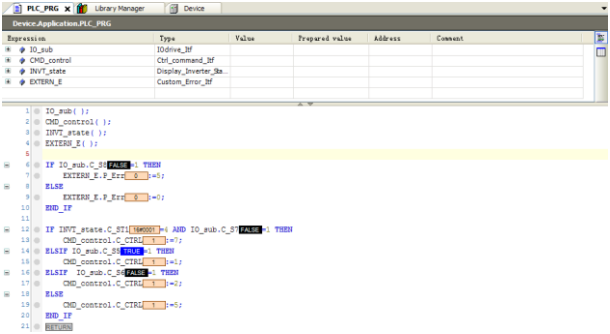


Figure 5-3

5.3 Example 3

Define a 16-bit unsigned variable APPLE, and add 10 to it repeatedly (ranging from 0 to 65535). The following table describes the requirements.

Value of APPLE	Multi-step speed used
< 10,000	Multi-step speed 0
≥ 10,000 & < 20,000	Multi-step speed 1
≥ 20,000 & < 30,000	Multi-step speed 2
≥ 30,000 & < 40,000	Multi-step speed 3
≥ 40,000 & < 50,000	Multi-step speed 4
≥ 50,000	Multi-step speed 5

In addition, the variable APPLE is to be displayed in P28.01. You can observe the value change on the inverter. Besides, when there are input signals on terminal S1 of the inverter, the inverter runs; and when there are input signals on terminal PS1 of the programmable extension card, terminals PY1 and PY2 output signals.

Implementation method: The library CmpINVT_Library needs to be added in the new project. Two programming languages are used in this example: ST for setting the multi-step speed and the control over the running of the inverter through terminal S1, that is, a file of the function type with the name of PLC_PRG is created; and ladder diagram for the control over the PY1 and PY2 output through terminal PS1 of the programmable extension card, that is, a POU of the function block type with the name of POU_FB is created. In addition, both the ST and

ladder diagram programming modes use the IOdrive_Itf programming interfaces related to I/O terminals. Therefore, you can define them directly in GVL global variables. For details, see Figure 5-4.

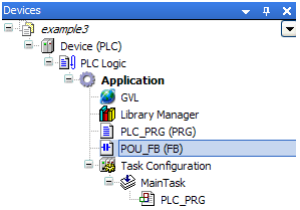


Figure 5-4

Note:

When the multi-step speed running function is used, you need to set P00.06 of the inverter to 6, indicating that the frequency source is multi-step speed. Similar to the multi-step speed setting for the inverter, you need to set the value of the speed in each step in the P10 parameter group of the inverter.

Figure 5-5 shows the online commissioning and monitoring results.

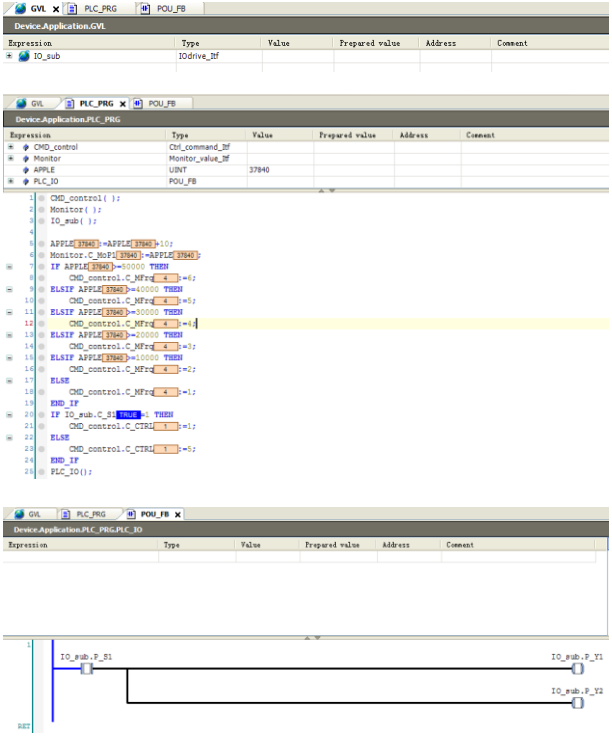

GVL
PLC_PRG
POU_FB
Device.Application.GVL
Expression
Type
Value
Prepared value
Address
Comment
IO_sub
IOdrive_Itf
Device.Application.PLC_PRG
CMD_control
Ctrl_command_Itf
Monitor
Monitor_value_Itf
APPLE
UINT
37840
PLC_IO
POU_FB
CMD_control( );
Monitor( );
IO_sub( );
APPLE[37840]:=APPLE[37840]+10;
Monitor.C_MoPS[37840]:=APPLE[37840];
IF APPLE[37840]>=50000 THEN
CMD_control.C_MFrq[4]:=6;
ELSIF APPLE[37840]>=40000 THEN
CMD_control.C_MFrq[4]:=5;
ELSIF APPLE[37840]>=30000 THEN
CMD_control.C_MFrq[4]:=4;
ELSIF APPLE[37840]>=20000 THEN
CMD_control.C_MFrq[4]:=3;
ELSIF APPLE[37840]>=10000 THEN
CMD_control.C_MFrq[4]:=2;
ELSE
CMD_control.C_MFrq[4]:=1;
END_IF
IF IO_sub.C_SI[TRUE]=1 THEN
CMD_control.C_CTRL[1]:=1;
ELSE
CMD_control.C_CTRL[1]:=5;
END_IF
PLC_IO();
Device.Application.PLC_PRG.PLC_IO
IO_sub.P_SI
IO_sub.P_Y1
IO_sub.P_Y2
RET

Figure 5-5

Appendix A Related functions codes of the inverter

Function code	Name	Detailed parameter description	Setting range	Default value
P00.02	Communication channel of running commands	0: Modbus 1: PROFIBUS/CANopen/DeviceNet 2: Ethernet 3: EtherCat/PROFINET 4: Programmable extension card 5: Bluetooth **Note:** The values 1, 2, 3, 4 and 5 indicate extension functions that can be used only after the corresponding cards are inserted.	0–5	0
P00.06	Setting mode of frequency A	0: Keypad 1–13: Reserved 14: Programmable extension card 15: Reserved	0–15	0
P00.07	Setting mode of frequency B	0: Keypad 1–13: Reserved 14: Programmable extension card 15: Reserved	0–15	2
P03.11	Torque setting mode	0: Torque control disabled 1–11: Reserved 12: Programmable extension card (100% corresponding to the rated current of the motor)	0–12	0
P03.14	Setting mode of upper forward running frequency limit in torque control	0: Keypad (P03.16) 1–10: Reserved 11: Programmable extension card (100% corresponding to the maximum frequency) 12: Reserved	0–12	0
P03.15	Setting mode of upper reverse running frequency limit in torque control	0: Keypad (P03.17) 1–10: Reserved 11: Programmable extension card (100% corresponding to the maximum frequency) 12: Reserved	0–12	0

Function code	Name	Detailed parameter description	Setting range	Default value
P03.18	Setting mode of upper electromotive torque limit	0: Keypad (P03.20) 1–9: Reserved 10: Programmable extension card (100% corresponding to three time the rated current of the motor) 11: Reserved	0–11	0
P03.19	Setting mode of upper brake torque limit	0: Keypad (P03.21) 1–9: Reserved 10: Programmable extension card (100% corresponding to three time the rated current of the motor) 11: Reserved	0–11	0
P04.27	Voltage setting mode	0: Keypad (P04.28) 1–11: Reserved 12: Programmable extension card 13: Reserved	0–13	0
P06.01	Y1 output	0: Invalid 1–40: Reserved 41: Corresponding to C_Y1 of CODESYS (P27.00 needs to be set	0–63	0
P06.02	HDO output		0–63	0
P06.03	Relay output RO1		0–63	1

Function code	Name	Detailed parameter description	Setting range	Default value
P06.04	Relay output RO2	to 1.) 42: Corresponding to C_Y2 of CODESYS (P27.00 needs to be set to 1.) 43: Corresponding to C_HDO of CODESYS (P27.00 needs to be set to 1.) 44: Corresponding to C_RO1 of CODESYS (P27.00 needs to be set to 1.) 45: Corresponding to C_RO2 of CODESYS (P27.00 needs to be set to 1.) 46: Corresponding to C_RO3 of CODESYS (P27.00 needs to be set to 1.) 47: Corresponding to C_RO4 of CODESYS (P27.00 needs to be set to 1.) 48–63: Reserved	0–63	5
P06.14	AO1 output	0: Running frequency	0–47	0
P06.15	Reserved	1–27: Reserved	0–47	0
P06.16	HDO high-speed pulse output	28: Corresponding to C_AO1 of CODESYS (P27.00 needs to be set to 1.) 29: Corresponding to C_AO2 of CODESYS (P27.00 needs to be set to 1.) 30–47: Reserved	0–47	0
P07.27	Type of current fault	0: No fault 1–44: Reserved 45: User-defined PLC card fault 1 (P-E1) 46: User-defined PLC card fault 2 (P-E2) 47: User-defined PLC card fault 3 (P-E3) 48: User-defined PLC card fault 4		
P07.28	Type of last fault			
P07.29	Type of last but one fault			
P07.30	Type of last but two fault			
P07.31	Type of last but three fault			

Function code	Name	Detailed parameter description	Setting range	Default value
P07.32	Type of last but four fault	(P-E4) 49: User-defined PLC card fault 5 (P-E5) 50: User-defined PLC card fault 6 (P-E6) 51: User-defined PLC card fault 7 (P-E7) 52: User-defined PLC card fault 8 (P-E8) 53: User-defined PLC card fault 9 (P-E9) 54: User-defined PLC card fault 10 (P-E10)		
P09.00	PID reference setting mode	0–10: Reserved 11: Programmable extension card 12: Reserved	0–12	0
P09.02	PID feedback setting mode	0–8: Reserved 9: Programmable extension card 10: Reserved	0–10	0
P19.00	State of slot 1	0–65535 0: No card 1: Programmable extension card 2–10: Reserved	0–65535	0
P19.01	State of slot 2	0–65535 0: No card 1: Programmable extension card 2–7: Reserved	0–65535	0
P19.02	State of slot 3	0–65535 0: No card 1: Programmable extension card 2–7: Reserved	0–65535	0

Function code	Name	Detailed parameter description	Setting range	Default value
P27.00	Programmable extension card enabling	This parameter is used to enable or disable the programmable extension card. 0: Disabled 1: Enabled The related programming interfaces and data of the inverter are valid only after the programmable extension card is enabled.	0–1	0
P27.01	C_WrP1	0–65535 Parameter value written to WrP1 of the PLC	0–65535	0
P27.02	C_WrP2	0–65535 Parameter value written to WrP2 of the PLC	0–65535	0
P27.03	C_WrP3	0–65535 Parameter value written to WrP3 of the PLC	0–65535	0
P27.04	C_WrP4	0–65535 Parameter value written to WrP4 of the PLC	0–65535	0
P27.05	C_WrP5	0–65535 Parameter value written to WrP5 of the PLC	0–65535	0
P27.06	C_WrP6	0–65535 Parameter value written to WrP6 of the PLC	0–65535	0
P27.07	C_WrP7	0–65535 Parameter value written to WrP7 of the PLC	0–65535	0
P27.08	C_WrP8	0–65535 Parameter value written to WrP8 of the PLC	0–65535	0
P27.09	C_WrP9	-32768–32767 Parameter value written to WrP9 of the PLC	-32768–32767	0
P27.10	C_WrP10	-32768–32767 Parameter value written to WrP10 of the PLC	-32768–32767	0

Function code	Name	Detailed parameter description	Setting range	Default value
P27.11	Running state of CODESYS	0–1 0: Stopped (no output) 1: Running	0–1	0
P27.12	C_MoP1	0–65535 Monitoring (viewing) the value of MoP1 of the PLC	0–65535	0
P27.13	C_MoP2	0–65535 Monitoring (viewing) the value of MoP2 of the PLC	0–65535	0
P27.14	C_MoP3	0–65535 Monitoring (viewing) the value of MoP3 of the PLC	0–65535	0
P27.15	C_MoP4	0–65535 Monitoring (viewing) the value of MoP4 of the PLC	0–65535	0
P27.16	C_MoP5	0–65535 Monitoring (viewing) the value of MoP5 of the PLC	0–65535	0
P27.17	C_MoP6	0–65535 Monitoring (viewing) the value of MoP6 of the PLC	0–65535	0
P27.18	C_MoP7	0–65535 Monitoring (viewing) the value of MoP7 of the PLC	0–65535	0
P27.19	C_MoP8	0–65535 Monitoring (viewing) the value of MoP8 of the PLC	0–65535	0
P27.20	C_MoP9	-32768–32767 Monitoring (viewing) the value of MoP9 of the PLC	-32768–32767	0
P27.21	C_MoP10	-32768–32767 Monitoring (viewing) the value of MoP10 of the PLC	-32768–32767	0

Service line:86-755-86312859 Website:www.invt.com

The products are owned by **Shenzhen INVT Electric Co.,Ltd.**
Two companies are commissioned to manufacture: (For product code, refer to the 2nd/3rd place of S/N on the name plate.)

Shenzhen INVT Electric Co., Ltd. (origin code: 01)
Address: INVT Guangming Technology Building, Songbai Road, Matian, Guangming District,Shenzhen, China

INVT Power Electronics (Suzhou) Co., Ltd. (origin code: 06)
Address: 1# Kunlun Mountain Road, Science&Technology Town, Gaoxin District, Suzhou, Jiangsu, China

Industrial Automation: ■Frequency Inverter ■Servo & Motion Control ■Motor & Electric Spindle ■PLC
■HMI ■Intelligent Elevator Control System ■Traction Drive

Electric Power: ■SVG ■Solar Inverter ■UPS ■Online Energy Management System

66001-00599

201902 (V1.0)