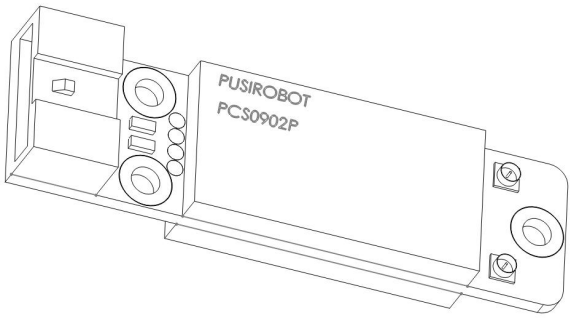
PUSIROBOT

CQPUSI ROBOT CONTROL SYSTEM

User Manual

PCS09xx Series

Programmable capacitance sensor controller



1. Version Control

1） Update Records

Date	Author	Version	Remark
2015-6-4	lijiwen	V0.1.0	Initial
2015-6-22	Liur	V0.1.1	Add operation description
2015-8-26	Jacky	V0.1.2	Add command description
2016-05-13	Huangchen	V0.1.3	Update software UI
2018-06-23	Liur	V0.1.4	Migrate to PCS0902
2018-08-25	ljw	V0.1.5	Update outline dimension
2018-11-1	Huangcheng	V0.1.6	Add sensor control word description
2021-05-11	Liur	V0.1.7	Add additional instruction of mode1and2
2023-10-8	wangwang	V0.1.8	Modify connector and dimension

Catalog

1 Introduction

1.1 Statement of intellectual property right

PCS09xx series controller has been applied for the following national patent:

• Controller scheme and method have been applied for the protection of the invention patent.

• Controller circuit has been applied for the protection of utility model patent.

• Controller appearance has been applied for the protection of appearance patent protection.

PCS09xx series controller has embedded firmware code, it would be considered as a violation of intellectual property protection act and regulations that any behavior of trying to destroy the function of firmware code protection. If this behavior acquires the software or other achievements of intellectual property protection without authorization of CQPUSI, CQPUSI has the right to stop such behavior by filing a lawsuit according to the act.

1.2 Disclaimer

The using method of the device and other content in the description of this manual is only used to provide convenience for you, and may be update in future version. To ensure the application conforms to the technical specifications is the responsibility of your own. CQPUSI does not make any form of statement or guarantee to the information, which include but not limited to usage, quality, performance, merchantability or applicability of specific purpose. CQPUSI is not responsible for these information and the consequences result caused by such information. If the CQPUSI device is used for life support and/or life safety applications, all risks are borne by the buyer. The buyer agrees to protect the CQPUSI from legal liability and compensation for any injury, claim, lawsuit or loss caused by the application.

2 Overview

2.1 General Description

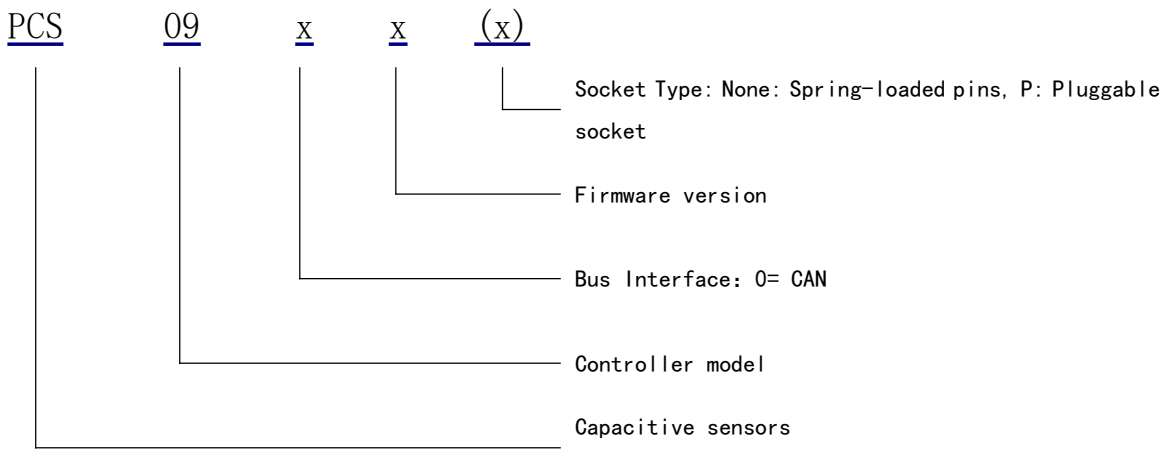
PCS09xx is a programmable capacitance sensor controller with high cost performance and simple and flexible interface. The series controllers can detect capacitance which changes below 1pf, and can convert the test results into IO level output and intuitive LED indication. PCS09xx controller provides a simple and convenient set of control commands, which can be connected to the control host through CAN bus (or RS485 or RS232) and directly read the measured values. It is suitable for various medical, environmental protection, reagent level detection for laboratory instruments, container level detection and other application scenarios.

2.2 Feature

- ✓ Two capacitive sensor detection channels
- ✓ The edge of digital level output can be configured
- ✓ LED result indication
- ✓ Capacitance detection range: 0 ~ 2.5nf
- ✓ Accuracy of the capacitive detection: 10fF
- ✓ Microsecond ultrafast response speed
- ✓ Using active shielding technology to improve detection reliability
- ✓ Adaptive low frequency excitation circuit for ultra-low EMI radiation
- ✓ Gold Plated shielded connectors for excellent anti-interference capability
- ✓ 16mm x 56mm x7.6mm. Small volume, easy to install
- ✓ User defined programming, and can be configured offline automatic operation
- ✓ Control routines and underlying drivers based on VC++

2.3 Production & Ordering Information

In order to serve you quicker and better, please provide the product number in following format when ordering PCS09xx:



3 Connector Description

3.1 Terminal port location

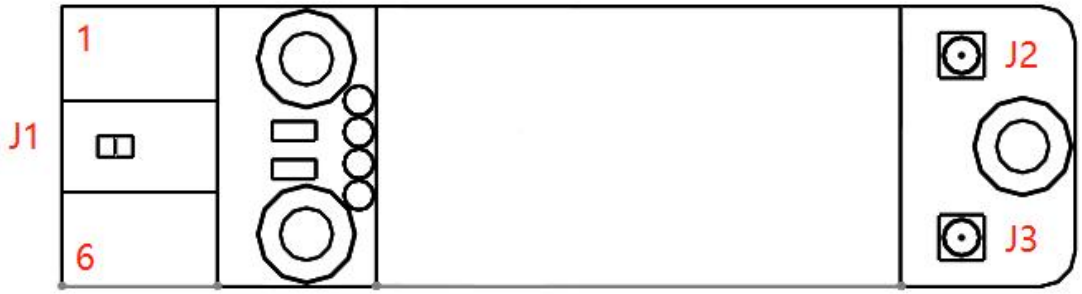


Figure 3-1

3.2 Power connection J1

Pin	1	2	3	4	5	6
Designator	NC	TR	CANH	CANL	GND	VCC

Description:

VCC: Supply voltage, 9~24V, 12V nominal;

GND: Supply voltage ground.

CANH/CANL: CAN communication interface;

TR: Trigger signal output pin (5V), configurable to be high or low after triggering;

3.3 Measurement interface J2

Capacitance measurement channel 2 interface, using HRS U.FL-R-SMT-1 coaxial connector. The shield of the connector interface is suspended and placed as close as possible to the electrode or reagent pin, and the core of the connector interface is connected to the positive pole of the measuring electrode.

3.4 Measurement interface J3

Capacitance measurement channel 1 interface, using HRS U.FL-R-SMT-1 coaxial connector. The shield of the connector interface is suspended and placed as close as possible to the electrode or reagent pin, and the core of the connector interface is connected to the positive pole of the measuring electrode.

4 Typical application

PCS09xx controller uses coaxial cable to connect with the capacitor plate to be measured. There are several typical connection modes of double-layer metal probe level detection, which can be selected and set by PUSICAN tool software, as shown below.

4.1 Single-ended pattern 1

In this pattern, first detection channel is selected. The shielding layer of J4 connects the outer metal of the probe, and the core wire of the J4 connects the inner layer of the probe. Because of the active shielding signal on the shielding layer of J4, the outer metal of the probe must be insulated from the frame reference ground and

power ground.

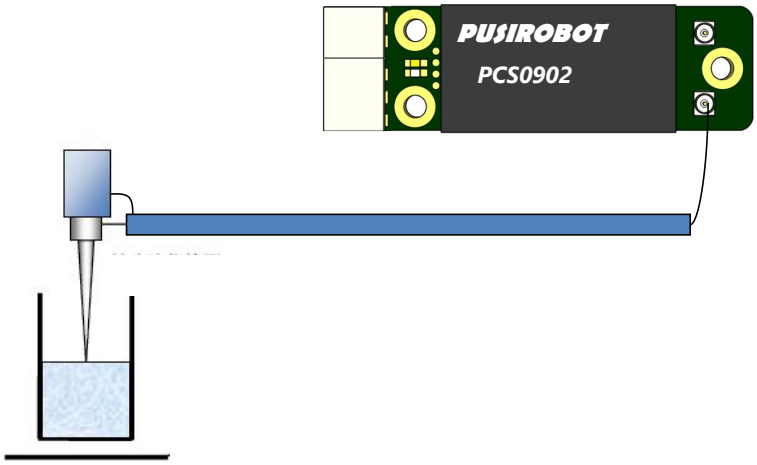


Figure 4-1

4.2 Single-ended pattern 2

In this pattern, second detection channel is selected. It is similar to the connection mode of single-ended pattern 1.

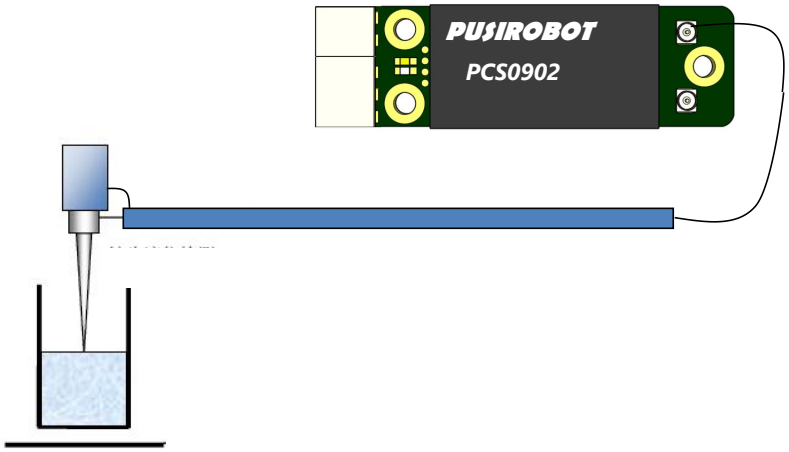


Figure 4-2

4.3 Single-ended pattern 3

When the volume of the reagent bottle or the liquid to be tested is very small, in order to improve the detection accuracy, the connection method of Figure 4-3 can be used. The core wire of the J4 cable is connected to the probe and the shield layer is suspended. The core wire of the J3 cable is connected to the metal plate or metal container. This pattern can also be used to measure the capacitance value of electronic components at both ends.

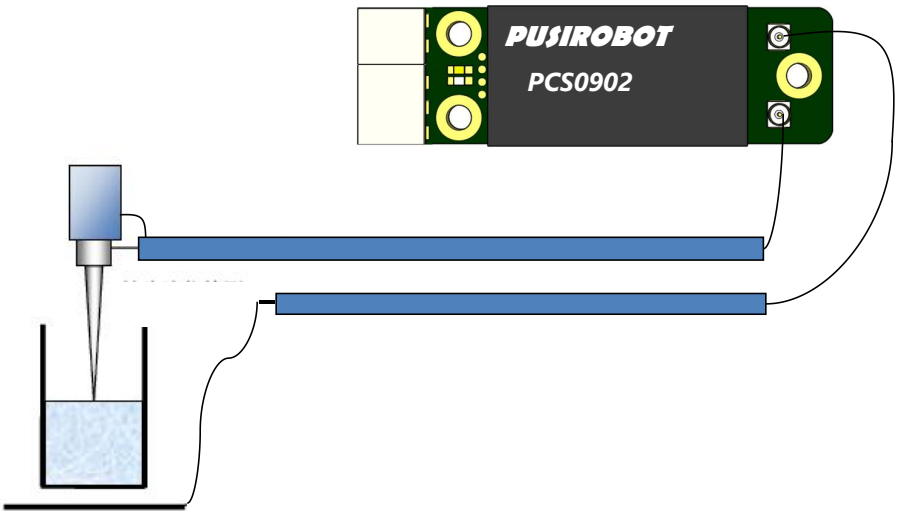


Figure 4-3

4.4 Differential pattern

If the coupling capacitance of the equipment rack is very large, or the liquid level in the container needs to be measured non-contact, the differential pattern can be used. The core wire of the J3 cable is connected to one side electrode, and the core wire of the J4 cable is connected to the electrode on the other side. If the electrodes on both sides are made of two layers of metal, and the outer layers are connected to the shielding layer of the coaxial cable respectively, the detection accuracy will be significantly improved.

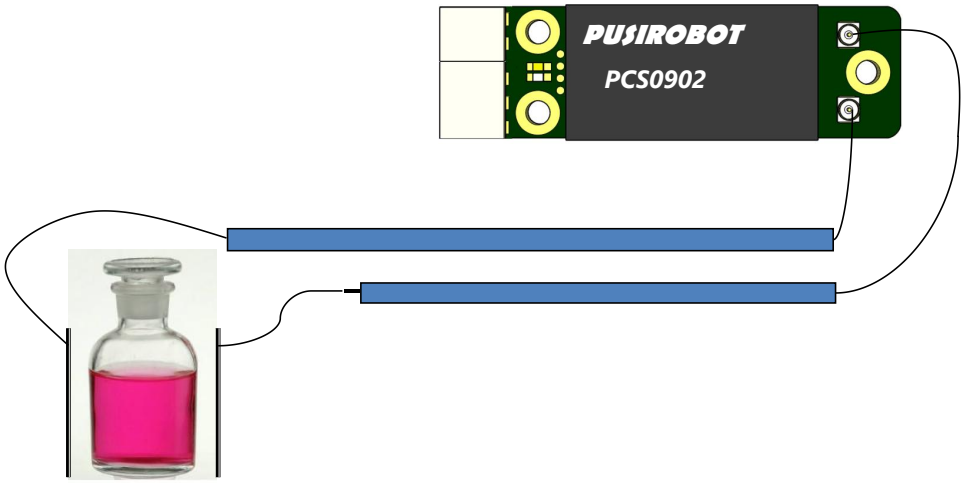
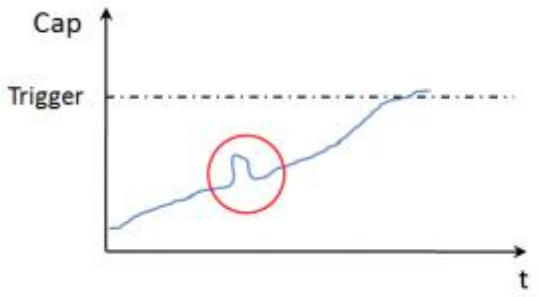


Figure 4-4

4.5 Sliding window mode

The sliding window mode is suitable for cases where the capacitance value drifts slowly or the capacitance to be measured is very small, as shown in the figure below, in the sliding window mode, it can achieve stable detection of capacitance jumps down

to 500fF, it is superimposed on the above four modes, please refer to the 6003 register description for the specific configuration.



4.6 Trigger configuration

PCS09xx controller supports capacitance trigger level output, and the trigger threshold can be configured by debugging tool software PUSICAN. When the measured capacitance reaches the threshold value, the corresponding LED indicator lights up, and the TR output port will output the corresponding level.

4.7 Capacitance calibration

In order to obtain better stability and detection accuracy, the PCS09xx controller uses digital signal processing technology to control the measurement loop, as shown in Figure 4-5 below, the external environment (temperature, humidity) and connection mode of the system have a significant impact on the system reference capacitance value.

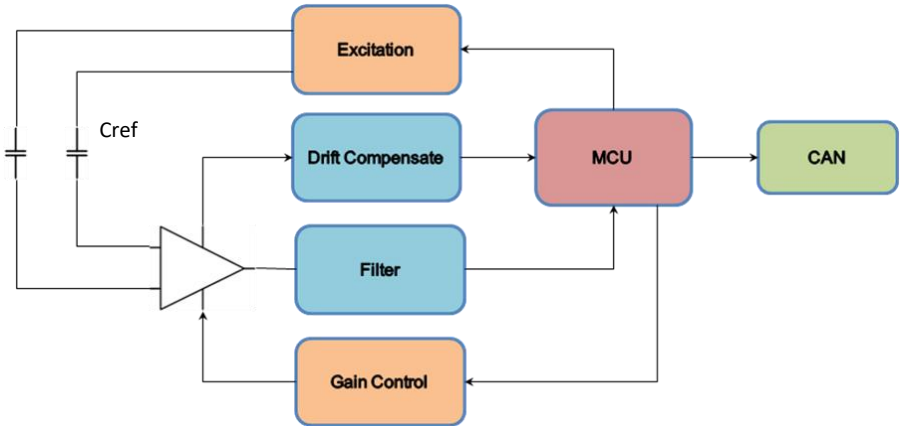


Figure 4-5

PCS09xx built-in control algorithm ensures that measurement linearity, as shown in Figure 4-6, is maximally protected from these environmental factors. However, users may need to compensate for the basic capacitance value depending on circumstances.

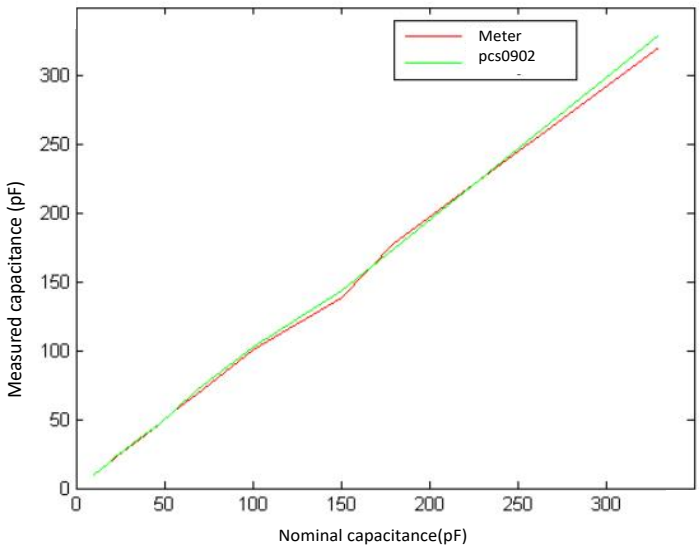


Figure 4-6

4.8 Empty needle calibration

When the PCS09xx controller and level probe assembly are installed, and the basic capacitance formed by external mechanisms such as the rack has been stabilized, the empty pin calibration can be performed through the command (or PUSICAN software tool), and the PCS09xx will obtain the current capacitance value and automatically subtract the capacitance value in the subsequent measurement process. When the set trigger value is less than the capacitance value after the null pin calibration, the TIRG pin will output the trigger level.

4.9 Automatic reporting mode

After the PCS09xx controller is powered on, it enters the continuous measurement mode automatically, and the Active indicator on the board flickers. The measurement results will be automatically reported through the PDO of the CANOPEN protocol.

Note: RS485/RS232 version does not have automatic reporting function due to protocol restrictions.

4.10 Response delay

For high-precision laboratory testing equipment, the accuracy of liquid level detection, false alarm rate, response speed are three important indicators. The PCS09xx controller uses a high-speed processor to execute control routines and filter algorithms, and reduces the response time to microseconds, while ensuring detection accuracy and false alarm rate. As shown in the following figure. At the same time, in order to reduce the antenna effect caused by the reagent needle, one of the main defects of inductance resonance measurement, and reduce electromagnetic radiation, the PCS09xx controller limits the excitation frequency to less than 10 KHz.

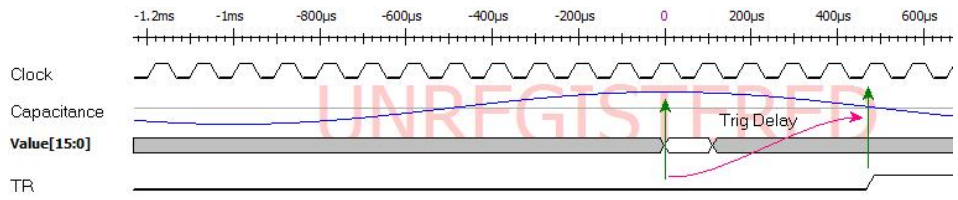


Figure 4-8

5 Installation method

The PCS09xx has three mounting holes, and it is recommended to use three M3 copper pillars for auxiliary installation, as shown in Figure 4-9 below. The height of the copper pillar should be greater than 3mm to avoid interference between the module and the base. Once the controller is securely locked, the cable connection is ready to go.

PCS09xx uses wiring plugs to plug in, and the applicable wire specification is #26~#22AWG.

Figure 4-9

6 CANopen communication

6.1 CANopen introduction

CAL provides all network management services and message protocols, but it does not define the contents of objects or the kind of objects being communicated (it defines how, not what). This is where CANopen enters the picture. CANopen is built on top of CAL, using a subset of CAL services and communication protocols, and providing an implementation of a distributed control system. It does this in such a way that nodes can range from simple to complex in their functionality without compromising the interoperability between the nodes in the network.

Central concept in CANopen is the Device Object Dictionary(OD), a concept used in other fieldbus systems as well(Profibus, Interbus-S). CANopen communication through the object dictionary (OD) can access all the parameters of the driver. Note that the Object Dictionary of CAL, it is an implementation aspect of CANopen. The Object Dictionary which PCS0902 is supported is shown in Appendix 1.

CANopen communication model defines the following messages (communication objects);

Abbreviation	Full name	Description
SDO	Service Data Object	Used for non-time critical data, such as

		parameters.
PDO	Process Data Object	Used to transfer time critical data(Setting values, Control word, status information, etc.)
SYNC	Synchronization Message	Used to synchronize CAN nodes.
EMCY	Emergency Message	Used to transport alarm event of a driver.
NMT	Network Management	Used for CANopen network management.
Heartbeat	Error Control Protocol	Used for monitoring the life status of all nodes.

6.2 CAN frame structure

CAN Data is transmitted between the host (controller) and the bus node through the data frame. The following table is the structure of the data frame.

Header	Arbitration domain		Control domain	Data domain	Check field	Response domain	Tail frame
	COB-ID(communication object identifier)	RTR(remote request)					
1bit	11 or 29 bits	1bit	6bits	0~8byte	16bits	2bits	7bits

When the transceiver is realized through the software, except to the COB-ID and data domain, the other bits in the frame structure are completed by the hardware of CAN transceiver controller. So Users only need to limit the COB-ID and data domain.

Note: This drive uses standard frame format, COB-ID is 11 bits. Remote frame is not supported temporarily.

The distribution of COB-ID is as follows:

Function Code				NODE ID (Node address)						
10	9	8	7	6	5	4	3	2	1	0

The controller parameters are accessed by the SDO read and write objects. For the driver, whose status information is needed to report to the main station in real time can be achieved by configuring the PDO.

6.3 CAN communication configuration

PCS0902 factory default settings: node ID is 5, the baud rate is 125Kbit/s. The user can modify the settings by supporting the CANOPEN master debugging tool.

6.3.1 Node ID

Object name	节点 ID
SDO ID	0x2002
Object type	U8, rw

Range	1-127
Storage type	ROM
Default value	5

6.3.2 Baud rate

Object name	Baud rate
SDO ID	0x2003
Object type	U8,rw
Range	0,1,2,3,4,5,6,7,8
Storage type	ROM
Default value	5

Relationship between each index and the baud rate is as follows:

0: 20Kbit/s

1: 25Kbit/s

2: 50Kbit/s

3: 100Kbit/s

4: 125Kbit/s

5: 250Kbit/s

6: 500Kbit/s

7: 800Kbit/s

8: 1000Kbit/s

6.4 System information acquisition

6.4.1 Device node name

Object name	Device node name
SDO ID	0x1008
Object type	string,ro
Range	-
Storage type	ROM
Default value	-

6.4.2 Hardware version

Object name	Hardware version
SDO ID	0x1009
Object type	string,ro
Range	-
Storage type	ROM
Default value	-

6.4.3 Software version

Object name	Software version
SDO ID	0x100A
Object type	string,ro
Range	-
Storage type	ROM
Default value	-

6.4.4 System control

Object name	System control
SDO ID	0x2007
Object type	U8,ro
Range	1, 2, 3
Storage type	RAM
Default value	-

System control values are defined as follows:

1: Jump to bootloader

2: Save Object Dictionary parameters

3: Reset factory settings

Note: The Storage type in the Object Dictionary which is ROM parameter is temporarily stored in memory after written by SDO. If you need to keep it permanently, you need to perform power down save operation for the Object Dictionary parameter.

6.5 Sensor parameters

6.5.1 Sensor status word

Object name	Sensor status
SDO ID	0x6000
Object type	U16,rw
Range	bit
Storage type	RAM
Default value	0-3

Sensor status is defined as follows:

Bit0: Triger

Bit1: Overflow

When the overflow state is set to one, the trigger state bit is also set to one, so there are only three values: 0/1/3.

6.5.2 Sensor control word

Object name	Sensor control word
SDO ID	0x6001
Object type	U16,rw
Range	bit
Storage type	RAM
Default value	0-1

The sensor status word is defined as follows:

Bit0: empty needle calibration

In addition, the empty pin calibration is to display the current capacitance value as 0, which cannot eliminate the basic capacitance value, which can be simply understood as a tare function.

6.5.3 Capacitance value

Object name	Capacitance value (Unit: fF)
SDO ID	0x6002
Object type	S32,rw
Range	-2^31~2^31-1
Storage type	RAM
Default value	0

6.5.4 Work pattern

Object name	Work pattern
SDO ID	0x6003
Object type	U8, rw
Range	0~3
Storage type	RAM
Default value	0

The value of the working mode is defined at bit[2:0] as follows:

0: Single-ended pattern 1.

1: Single-ended pattern 2.

3: Single-ended pattern 3.

4: Differential pattern.

Bit[4] defines the sliding window mode to be turned on or off; When bit[4]=1 enables the sliding window mode, bit[15:8] defines the detection length parameter of the sliding window. bit[31:16] defines the trigger threshold, 16 bits with a signed number. The sliding window mode can be turned on at the same time as the mode defined by bit[2:0], which is equivalent to a detection method superimposed on bit[2:0].

Example of command settings:

Send: 605 23 03 60 00 11 0A 64 00

Return: 585 60 03 60 00 00 00 00 00

Note: Set the detection mode to sliding window mode, the detection mode is single-ended 2, the signal input port is, the detection length is 10, and the trigger value is 100ff.

It can also be directly set on the pusican debugging software above v60, the data input is the input trigger number and threshold, for example, 10 times, 100fF is 00 64 0A, converted to decimal input is 25610, and then check the sliding window mode. The original mode selection to single-ended 1 is to use the interface of single-ended 1 to apply the sliding window mode.

Note: (1) The sliding window will slide forward with the test waveform, and the fixed width is 51.2ms; (2) When the trigger is successful, the rising edge of the TR port will remain high for 90ms;(3) Due to the increase of functions, the object type of the 6003 register above v17 firmware is S32, and the object type of the old version below v17 is U8

6.5.5 Trigger settings

The trigger output of PCS09xx can be set. When the actual capacitance value is greater than the set capacitance value, the trigger signal output is effective.

Object name	Trigger setting
SDO ID	0x6005
Object type	Record
Storage type	ROM
Number of parameters	2

Subindex 0x01: Trigger polarity setting

Object type	U8,rw
Range	0~1
Default value	0

When set to 0, trigger signal output effective level is high level, and when set to 1, trigger signal output effective level is low level.

Subindex 0x02: Trigger capacitance value

Object type	S32,rw
Range	-2^31~2^31-1
Default value	0

7 Introduction of the Debug Tool

The PCS09xx can be used for command debugging, IO port setting detection, stepper motor parameter debugging and custom programming through the CQPUSI tool software PUSICAN.

7.1 Installation preparation

Tool software PUSICAN requires CAN adapter (USB2CAN or PCI2CAN) support. Currently, the tool software has supported a wide variety of common USB2CAN adapter on the market. If you need to support other types of adapter, please contact with sales staffs.

7.2 Software installation

7.2.1 Driver installation

Install the driver of adapter, please follow the instructions on the user manual of adapter.

7.2.2 Tool software installation

Debug tool PUSICAN is green free installation software. After download, extract it into a special folder. Double-click the pusican.exe, then the software is running.

7.3 Software instructions

7.3.1 Preparation for using

Connect the PCS09xx and CAN adapter to the computer. Then power PMC007CxSxP up. After power up normally, the LED lights will be flashing at 2.5Hz frequency.

7.3.2 Main interface

Double click on the desktop PUSICAN shortcut icon to enter the main interface. As shown in :

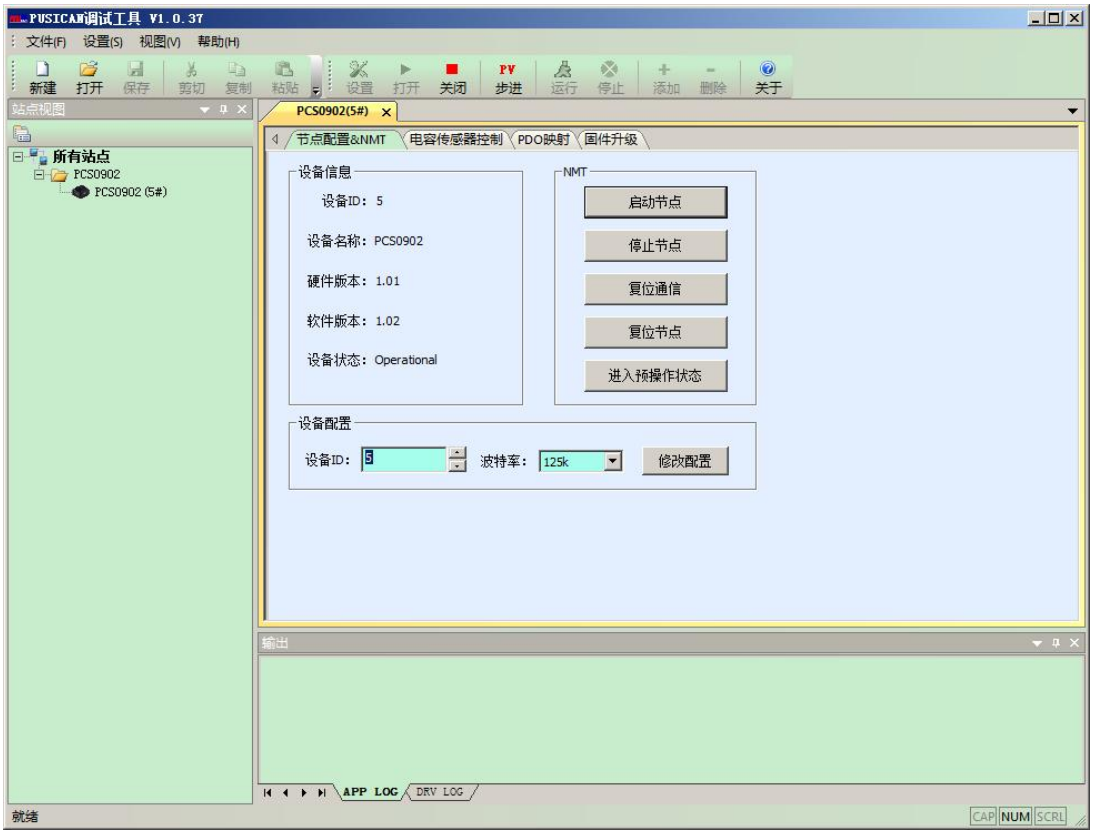


Figure 7-1

In the main interface, click "Settings" icon to select the adapter and baud rate. PCS09xx controller factory default baud rate is 125kHz. Click "open" icon, debugging tools will boot the adapter scanning site, and all online activity equipment are listed in the left side tree list. Double-click on operation site and the right workspace will display the control interface of the device.

Start node: Make PCS09xx into the operational state.

Stop node: Make PCS09xx into stopped state. The node will not respond to any SDO command.

Reset communication: The communication parameters take effect immediately by this operation, after SDO modifies the communication parameters.

Reset node: Notify the node reenter power on reset process.

Pre-operation state: In this state, the node waits for the network command of the main station, receives the configuration request of the main station, so it can receive and send all messages except PDO.

7.3.3 Capacitive sensor control

At the main interface, click "capacitance sensor control" to enter the interface. After entering the interface, the software will automatically acquire the capacitance value and display it on the interface. At the same time, the working pattern, trigger mode and other parameters can be set. As shown in Figure 7-2.

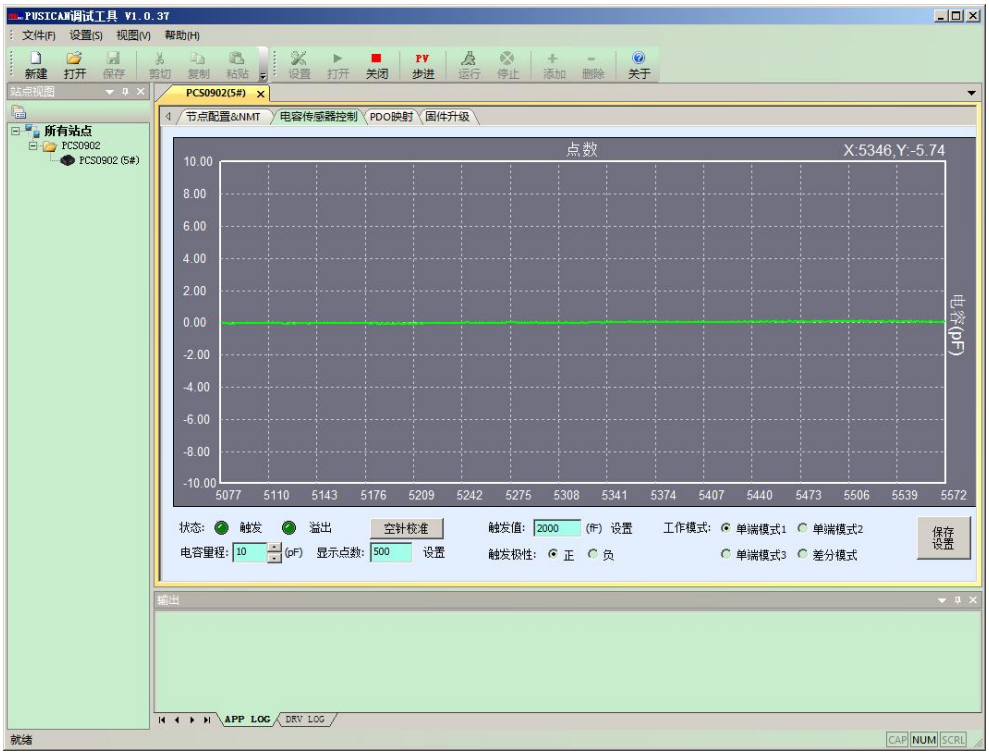


Figure 7-2

7.3.4 PDO mapping

Click "PDO Mapping" on the main interface to enter the interface, after entering the interface, the software will automatically read the current mapping object from the device and display it on the interface, as shown in 7-3.

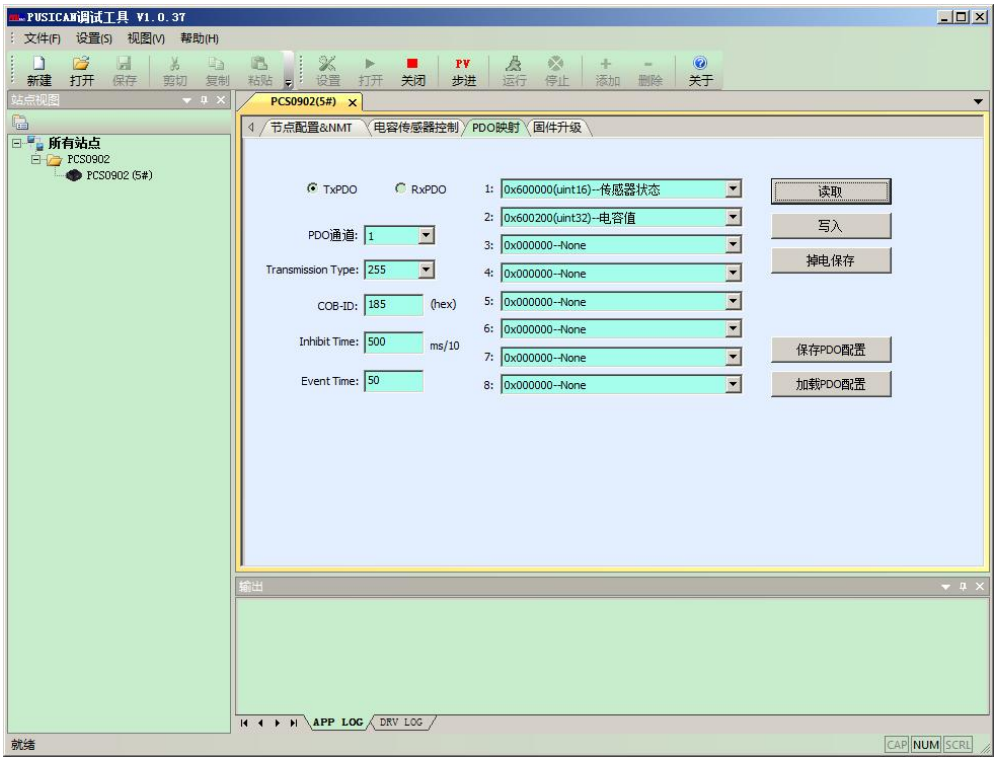


Figure 7-3

The PCS09xx supports 4 channels of transmit PDO and 4 channels of receive PDO,

with up to 8 objects per lane mapped. For details about the parameters, see Section 2.1 in Appendix 2.

7.3.5 Firmware upgrade

Firmware upgrade of PCS09xx can be carried out in bootloader mode through the CAN bus. In "Firmware upgrade" interface, click "into the bootloader/ application", PCS09xx will enter bootloader mode. The node ID and baud rate are set in the application model.

After entering the bootloader mode, LED lights will be double flash. In the "application path" column select the upgrade file, click the "upgrade" button to start upgrading, as shown in Figure 7-4. After firmware upgrade is completed, tool will prompt a dialog box, then you can click on "Enter bootloader/ application" (or repower on the controller), and then the controller will switch to the normal application mode.

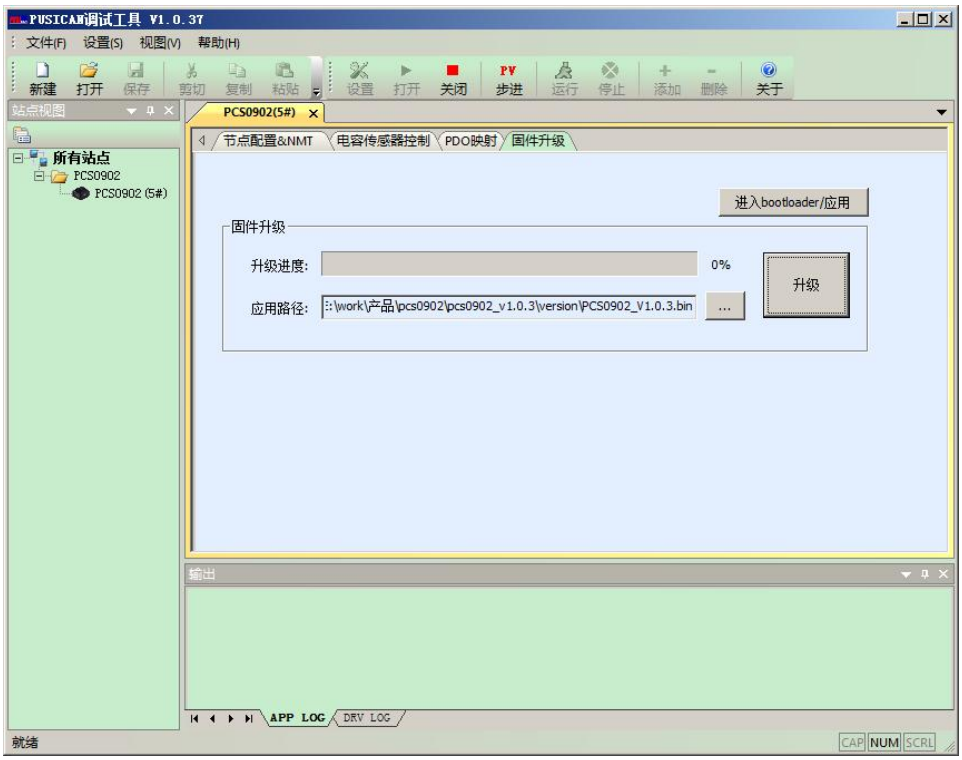
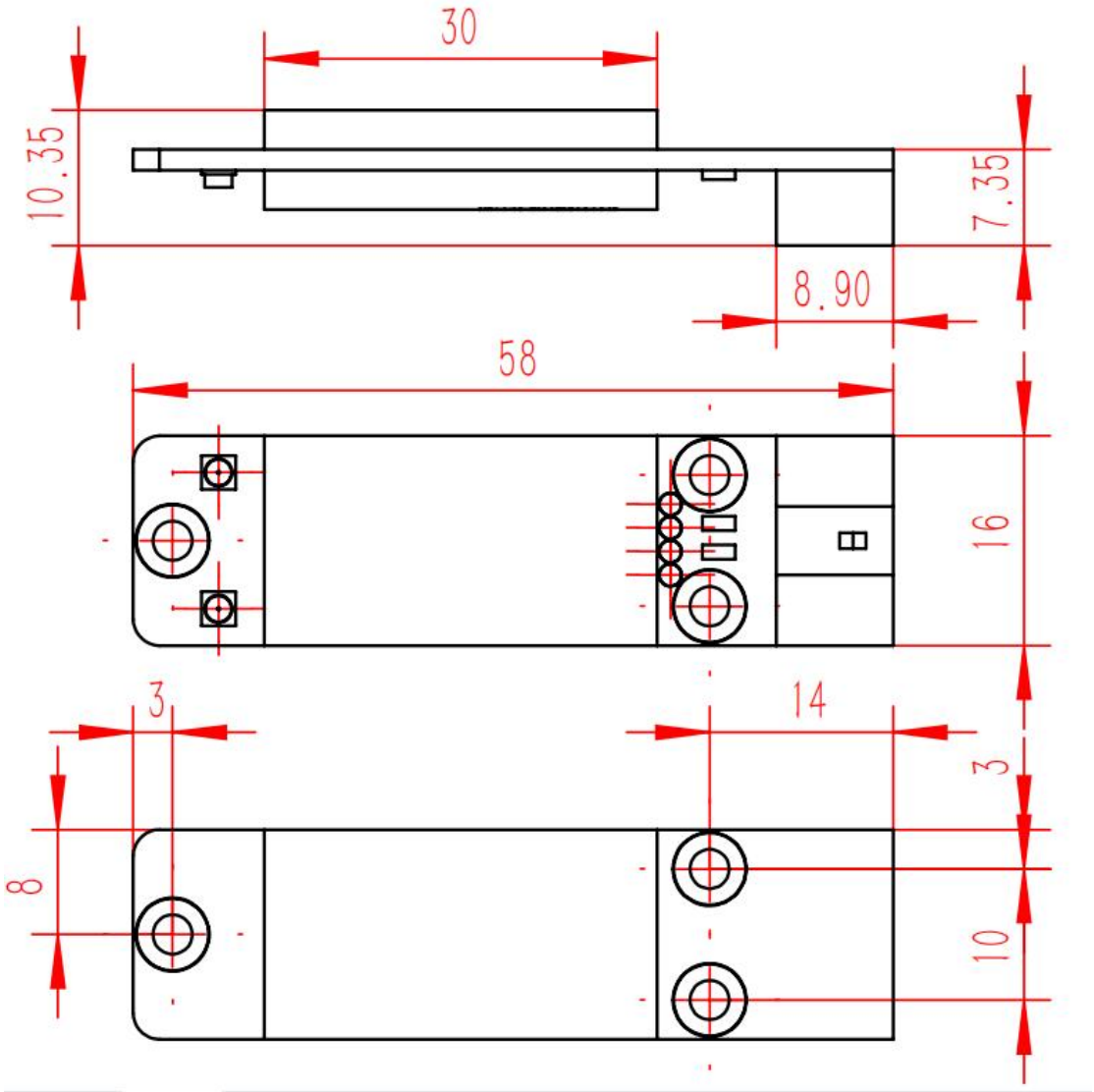


Figure 7-4

8 Electrical Characteristics

Parameter	Condition	Min	Typical	Max	Unit
Supply Power Voltage	Normal 25℃	4.6	5	12	V
Operation Temperature	12V DC	-20		55	℃
Maximum operating current	No trigger case	30	40	50	mA
Measurement interface high level	Normal 25℃	2.8	3.0	3.3	V
TR interface low level	12V Power Voltage	-0.5	0	1.0	V
TR interface high level	12V Power Voltage	2.8	3.0	3.3	V

9 Dimensions



10 Appendix 1 PCS09xx Object dictionary table

Index	Sub index	Object	Name	Type	Attr.	PDO	Storage type
1000h	--	VAR	Device type	UINT32	RO	NO	ROM
1001h	--	VAR	Error registor	UINT8	RO	Optional	RAM
1002h	--	VAR	manufacturer status register	UINT32	RO	Optional	RAM
1003h	--	ARRAY	pre-defined error field	--	--	--	RAM
	0h		number of errors	UINT8		NO	
	1h-7h		standard error field			Optional	
1005h	--	VAR	COB-ID SYNC	UINT32	RW	NO	ROM
1006h	--	VAR	communication cycle period	UINT32	RW	NO	ROM
1007h	--	VAR	synchronous window length	UINT32	RW	Optional	ROM
1008h	--	VAR	manufacturer device name	Visible String	const	NO	ROM
1009h	--	VAR	manufacturer hardware version	Visible String	const	NO	ROM
100ah	--	VAR	manufacturer software version	Visible String	const	NO	ROM
1014h	--	VAR	COB-ID Emergency message	UINT32	RO	NO	ROM
1015h	--	VAR	Inhibit Time EMCY	UINT16	RW	NO	ROM
1016h	--	ARRAY	Consumer Heartbeat Time	--	--	--	ROM
	0h		number entries	UINT8	RO	NO	
	1h-3h		Consumer Heartbeat Time	UINT32	RW	NO	
1017h	--	VAR	Producer Heartbeat Time	UINT16	RW	NO	ROM
1018h	--	RECORD	Identity Object	--	--	--	ROM
	0h		number of entries	UINT8	RO	NO	
	1h		Vendor ID	UINT32	RO	NO	
	2h		Product code	UINT32	RO	NO	
	3h		Revision number	UINT32	RO	NO	
	4h		Serial number	UINT32	RO	NO	
1200h	--	RECORD	Server SDO parameter	--	--	--	ROM
	0h		number of entries	UINT8	RO	NO	
	1h		COB-ID Client->Server (rx)	UINT32	RO	NO	
	2h		COB-ID Server -> Client (tx)	UINT32	RO	NO	

	3h		Node-ID of the SDO client	UINT32	RW	NO	
1280h	--	RECORD	Client SDO parameter	--	--	--	RAM
	0h		number of entries	UINT8	RO	NO	
	1h		COB-ID Client->Server (tx)	UINT32	RW	NO	
	2h		COB-ID Server -> Client (rx)	UINT32	RW	NO	
	3h		Node-ID of the SDO server	UINT32	RW	NO	
1400h	--	RECORD	receive PDO parameter	--	--	--	ROM
	0h		largest sub-index supported	UINT8	RO	NO	
	1h		COB-ID used by PDO	UINT32	RW	NO	
	2h		transmission type	UINT8	RW	NO	
	3h		inhibit time	UINT16	RW	NO	
	4h		compatibility entry	UINT8	RW	NO	
	5h		event timer	UINT16	RW	NO	
1401h	--	RECORD	receive PDO parameter	--	--	--	ROM
	0h		largest sub-index supported	UINT8	RO	NO	
	1h		COB-ID used by PDO	UINT32	RW	NO	
	2h		transmission type	UINT8	RW	NO	
	3h		inhibit time	UINT16	RW	NO	
	4h		compatibility entry	UINT8	RW	NO	
	5h		event timer	UINT16	RW	NO	
1402h	--	RECORD	receive PDO parameter	--	--	--	ROM
	0h		largest sub-index supported	UINT8	RO	NO	
	1h		COB-ID used by PDO	UINT32	RW	NO	
	2h		transmission type	UINT8	RW	NO	
	3h		inhibit time	UINT16	RW	NO	
	4h		compatibility entry	UINT8	RW	NO	
	5h		event timer	UINT16	RW	NO	
1403h	--	RECORD	receive PDO parameter	--	--	--	ROM
	0h		largest sub-index supported	UINT8	RO	NO	
	1h		COB-ID used by PDO	UINT32	RW	NO	
	2h		transmission type	UINT8	RW	NO	
	3h		inhibit time	UINT16	RW	NO	
	4h		compatibility entry	UINT8	RW	NO	
	5h		event timer	UINT16	RW	NO	

1600h	--	RECORD	receive PDO mapping	--	--	--	ROM
	0h		number of mapped application objects in PDO	UINT8	RO	NO	
	1h-8h		PDO mapping for the nth application object to be mapped	UINT32	RW	NO	
1601h	--	RECORD	receive PDO mapping	--	--	--	ROM
	0h		number of mapped application objects in PDO	UINT8	RO	NO	
	1h-8h		PDO mapping for the nth application object to be mapped	UINT32	RW	NO	
1602h	--	RECORD	receive PDO mapping	--	--	--	ROM
	0h		number of mapped application objects in PDO	UINT8	RO	NO	
	1h-8h		PDO mapping for the nth application object to be mapped	UINT32	RW	NO	
1603h	--	RECORD	receive PDO mapping	--	--	--	ROM
	0h		number of mapped application objects in PDO	UINT8	RO	NO	
	1h-8h		PDO mapping for the nth application object to be mapped	UINT32	RW	NO	
1800h	--	RECORD	transmit PDO parameter	--	--	--	ROM
	0h		largest sub-index supported	UINT8	RO	NO	
	1h		COB-ID used by PDO	UINT32	RW	NO	
	2h		transmission type	UINT8	RW	NO	
	3h		inhibit time	UINT16	RW	NO	
	4h		reserved	UINT8	RW	NO	
	5h		event timer	UINT16	RW	NO	
1801h	--	RECORD	transmit PDO parameter	--	--	--	ROM
	0h		largest sub-index supported	UINT8	RO	NO	
	1h		COB-ID used by PDO	UINT32	RW	NO	
	2h		transmission type	UINT8	RW	NO	
	3h		inhibit time	UINT16	RW	NO	
	4h		reserved	UINT8	RW	NO	

	5h		event timer	UINT16	RW	NO	
1802h	--	RECORD	transmit PDO parameter	--	--	--	ROM
	0h		largest sub-index supported	UINT8	RO	NO	
	1h		COB-ID used by PDO	UINT32	RW	NO	
	2h		transmission type	UINT8	RW	NO	
	3h		inhibit time	UINT16	RW	NO	
	4h		reserved	UINT8	RW	NO	
	5h		event timer	UINT16	RW	NO	
1803h	--	RECORD	transmit PDO parameter	--	--	--	ROM
	0h		largest sub-index supported	UINT8	RO	NO	
	1h		COB-ID used by PDO	UINT32	RW	NO	
	2h		transmission type	UINT8	RW	NO	
	3h		inhibit time	UINT16	RW	NO	
	4h		reserved	UINT8	RW	NO	
	5h		event timer	UINT16	RW	NO	
1a00h	--	RECORD	transmit PDO mapping	--	--	--	ROM
	0h		number of mapped application objects in PDO	UINT8	RO	NO	
	1h-8h		PDO mapping for the n-th application object to be mapped	UINT32	RW	NO	
1a01h	--	RECORD	transmit PDO mapping	--	--	--	ROM
	0h		number of mapped application objects in PDO	UINT8	RO	NO	
	1h-8h		PDO mapping for the n-th application object to be mapped	UINT32	RW	NO	
1a02h	--	RECORD	transmit PDO mapping	--	--	--	ROM
	0h		number of mapped application objects in PDO	UINT8	RO	NO	
	1h-8h		PDO mapping for the n-th application object to be mapped	UINT32	RW	NO	
1a03h	--	RECORD	transmit PDO mapping	--	--	--	ROM
	0h		number of mapped	UINT8	RO	NO	

			application objects in PDO				
	1h-8h		PDO mapping for the n-th application object to be mapped	UINT32	RW	NO	
2002h	--	VAR	Node ID	UINT8	RW	NO	ROM
2003h	--	VAR	Baud rate	UINT8	RW	NO	ROM
6000h	--	VAR	Sensor status word	UINT16	RW	Optional	RAM
6001h	--	VAR	Controller control word	UINT16	RW	Optional	RAM
6002h	--	VAR	Capacitance value	INT32	RW	NO	RAM
6003h	--	VAR	Working pattern	UINT8	RW	NO	RAM
6005h	--	RECORD	Trigger setting	UINT8	RW	NO	ROM
	0h		Number of parameters	UINT8	RO	NO	ROM
	1h		Trigger output polarity	UINT8	RW	NO	RAM
	2h		Trigger value	INT32	RW	NO	RAM

11 Appendix 2 CANOPEN Communication example

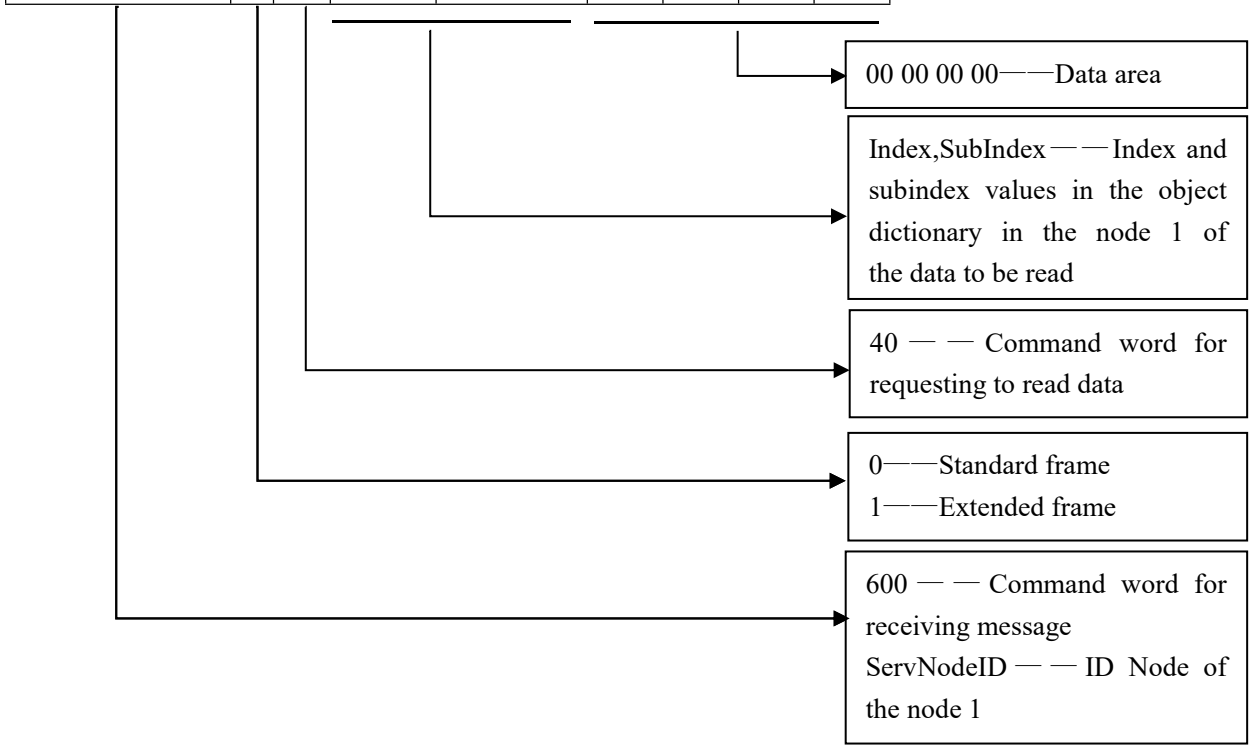
11.1 SDO Reading and writing example

11.1.1 SDO Read

11.1.1.1 Data frame forma

Master send:

600+ServNodeID	0	40	Index	Subindex	00	00	00	00



Slave response:

When the data length is 1 byte								
580+ServNodeID	0	4F	Index	Subindex	d0	0	0	0
When the data length is 2 bytes								
580+ServNodeID	0	4B	Index	Subindex	d0	d1	0	0
When the data length is 3 bytes								
580+ServNodeID	0	47	Index	Subindex	d0	d1	d2	0
When the data length is 4 bytes								
580+ServNodeID	0	43	Index	Subindex	d0	d1	d2	d3

11.1.1.2 SDO Read example

Master send: 605 40 02 60 00 00 00 00 00

Slave response:585 43 02 60 00 FC 5E 00 00

The master initiated a read request to the device whose node ID is 5. The index and subindex of the request are 0x6002 and 0x00 respectively, which corresponds to capacitance value in the PCS09xx Object Dictionary. The slave response 43 indicates that the parameter length is four byte, the data is 0x5EFC, which indicates that the capacitance is 24316fF.

11.1.2 SDO Write in

11.1.2.1 Data frame format

Master send:

When the data length is 1 byte								
600+ServNodeID	0	2F	Index	Subindex	d0	0	0	0
When the data length is 2 byte								
600+ServNodeID	0	2B	Index	Subindex	d0	d1	0	0
When the data length is 3 byte								
600+ServNodeID	0	27	Index	Subindex	d0	d1	d2	0
When the data length is 4 byte								
600+ServNodeID	0	23	Index	Subindex	d0	d1	d2	d3

Correct response from the slave station:

580+ServNodeID	0	60	Index	Subindex	00	00	00	00

00 00 00 00——Data area

Index, Subindex——Index and subindex values in the object dictionary in the node 3 of the data to be read

60 — Command word for response to write data successfully

0——Standard frame
1——Extended frame

580——Send SDO
ServNodeID — — ID Node of the node 3

Error response from the slave station:

580+ServNodeID	0	80	Index	Subindex	SDO abort code error

SDO abort code error — — According to the error, the code is different. For details see Appendix 4

Index, Subindex——Index and subindex values in the object dictionary in the node 3 of the data to be read

80 — Command word for response

0——Standard frame
1——Extended frame

580 — — Command word for sending message
ServNodeID — — ID Node of the node 3

Note: Abort code error SDO returns the corresponding parameters according to the specific error. The specific parameters are shown in Appendix 4.

11.1.2.2 SDO Write example

Master send: 605 2F 03 20 00 07 00 00 00

Slave response:585 60 03 20 00 00 00 00 00

The master initiated a write request to the device whose node ID is 5. The index and subindex of the request are 0x2003 and 0x00 respectively, which corresponds to the baud rate setting parameter in the PCS09xx Object Dictionary, and the write data is 7, which indicates the baud rate is set to 800Kbit/s. The slave response 60 indicates the data is written successfully.

12 Appendix 3 PDO configuration example

12.1 PDO overview

PDO communication is based on the Producer/Consumer model, which is mainly used to transfer real-time data. The node which generated data puts the data with its own node ID on the bus, and nodes which need the data can be configured to receive the data sent by the node. The transmission of PDO is triggered by the event, which can represent a change in a PDO variable and can also be a time of expiration or a specific message to be received. Process data is transmitted directly in an CAN message without a protocol header file. The length of a PDO is between 0 and 8 bytes.

PDOs is included in the mapping parameter and communication parameter. PMC007xx supports 4 PDOs.

12.1.1 The structure PDO——Mapping parameter

A PDO in the Object Dictionary consists of adjacent items. The mapping parameters define the connection of these items. A mapping parameter defines a data source through an index, a subindex, and a number of bits.

For example:

Index	*Sub-index*	*Object Data*	*Description*
0x1A00	0	4	Number of mapped entries
	1	0x20000310	The entry at index 0x2000, sub-index 3, with a length of 16 bit, is mapped to bytes 0 and 1 within the CAN message.
	2	0x20000108	The entry at index 0x2000, sub-index 1, with a length of 8 bit, is mapped to byte 2 within the CAN message.
	...	...	

Table 1: Example for mapping parameters for the first TPDO

A CAN message has not more than 8 bytes. This means that there can send 8 object items at most when there is only one PDO used.

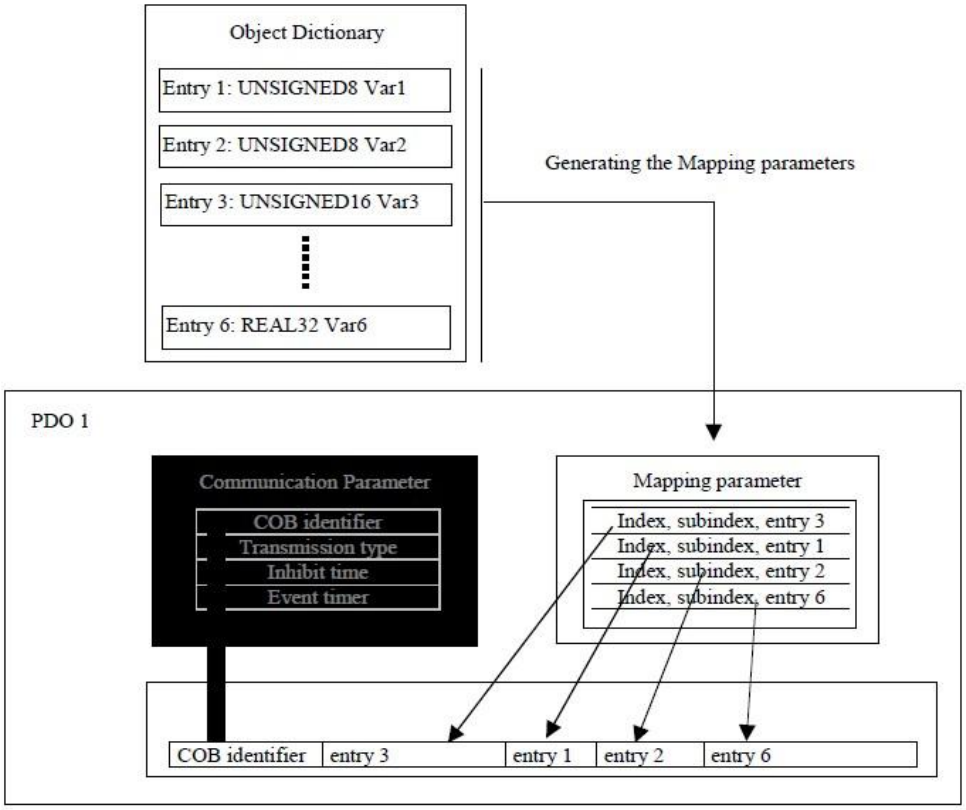


Figure 3: Mapping of Object Dictionary entries into a PDO

12.1.2 The structure PDO——Communication parameter

In order to transmit a PDO, the communication parameter defines the nature of the transport and the CAN identifier.

Index	Sub-index	Object Data	Description
1800h	0	Number on entries	
	1	COB-ID	CAN identifier for the PDO
	2	Transmission Type	transmission type of the PDO
	3	Inhibit Time	minimum inhibit time for a TPDO
	4	reserved	reserved
	5	Event Time	maximum time between two TPDOs

Table 4: Communication parameter for the first TPDO

PDO communication parameter is an item in the Object Dictionary.

(RPDOs: index 0x1400–0x15FF, TPDOs: 0x1800-0x19FF)

If allowed, the communication parameter can be modified by the CAN with the help of the data service object.

12.1.2.1 COB-ID(CAN identifier. Subindex 1)

COB-ID as proof of identity, the priority of PDO is before the bus access. For

every CAN message, there is only one sender (producer). However, it allows multiple recipients (consumers) for the existing message.

Bit	*31*	*30*	*29*	*28 – 11*	*10 - 0*
11-bit-ID	0/1	0/1	0	000000000000000000	11-bit identifier
29-bit-ID	0/1	0/1	1	29-bit identifier	

Table 5: Structure of a COB-ID for PDOs

The thirtieth bit is 0, which indicates that a remote transmission request (RTR) is allowed for this PDO.

PDO COB-ID distribution:

PDO1(send)	181H-1FFH
PDO1(receive)	201H-27FH
PDO2(send)	281H-2FFH
PDO2(receive)	301H-37FH
PDO3(send)	381H-3FFH
PDO3(receive)	401H-47FH
PDO4(send)	481H-4FFH
PDO4(receive)	501H-57FH

12.1.3 PDO Trigger mode

Sending of PDO can be triggered by the following methods:

1) Event trigger.
2) Time trigger.
3) Single query.
4) Synchronization.

When only use event to trigger the sending of PDO, once the event process is changed, the PDO is sent. It may bring very serious consequences, that is, when the frequency of a process data change is very high, the PDO is sent uninterruptedly, that will cause the message of other nodes is not sent out, which seriously affect the efficiency of the bus.

CANopen uses the " Inhibit time " mechanism to solve this problem. Inhibit time is a configurable time period in units of 100 us. The same PDO sends at least this time interval, so it can determine the maximum transmission frequency of an event triggered PDO.

Generally, the sending of PDO can be triggered by a combination of any of the trigger mode. But the most common way is to combine the Event trigger and Time trigger. In the case of single event trigger, since process data did not change for a long time (such

as temperature variables), the PDO have not been triggered for a long time. It will affect the nodes just joined the network. So if plus time triggered mode, PDO is forced to send again within the stipulated time. For example, for a PDO, the inhibition time is configured as 5, event timer is configured as 250, so the PDO can be sent when process data changes. The minimal interval of sending is 5ms, on the other hand, no matter whether there is no change in the data, the PDO will be sent every 250ms.

Configuration of PDO trigger mode is realized through setting subindex 2 in the Object Dictionary of PDO communication parameter. The range of the subindex is 0-255 The following lists the different values for different trigger modes.

0: PDO is sent after SYNC is received, but not cycle.

1-240: PDO is sent periodically after SYNC is received . The value is the number of SYNC between two send of PDO .

255：Event trigger.

12.2 PDO Configuration example

PCS09xxx supports PDO mapping by SDO configuration. To configure the GPIO value to be TPDO1 as an example, the SDO is sent as:

Set the communication COB-ID to be 187, that is, the device whose node ID is 7 receives the PDO

Master send: 605 23 00 18 01 87 01 00 00

Event trigger is set

Master send: 605 2F 00 18 02 FF 00 00 00

Set Inhibit time as 5ms

Master send: 605 2B 00 18 03 32 00 00 00

Set Event time as 1000ms

Master send:605 2B 00 18 05 E8 03 00 00

Set the number of map entries to be 1

Master send:605 2F 00 1A 00 01 00 00 00

Set the mapping parameters to map 0x6002 to TPDO1

Master send:605 23 00 1A 01 10 00 12 60

After the configuration is completed, PCS09xx will send PDO message every 1s. The message contains capacitance value information.

13 Appendix 4 SDO abort code error

Abort code	Code function description
05030000	There is no alternation of trigger bits
05040000	SDO protocol timeout
05040001	Illegal or unknown Client/Server command word
05040002	Invalid block size (only Block Transfer mode)
05040003	Invalid serial number (only Transfer Block mode)
05030004	CRC error (only Transfer Block mode)
05030005	Out of memory
06010000	Access is not supported for the Object.
06010001	Try to read write-only objects
06010002	Try to write read-only objects
06020000	Object does not exist in the Object Dictionary
06040041	Object cannot be mapped to PDO
06040042	The number and length of the mapped object exceeds the PDO length
06040043	General parameters are not compatible
06040047	General equipment is not compatible
06060000	Hardware error causes the object access failure
06060010	Data type does not match, and service parameter length does not match
06060012	Data type does not match, the service parameter is too large
06060013	Data type does not match, the service parameter is too small
06090011	The subindex does not exist
06090030	Beyond the range of the parameter values (when write access)
06090031	Parameter value is written too large
06090032	Parameter value is written too small
06090036	The maximum value is less than the minimum value
08000000	General error
08000020	Data cannot be transferred or saved to applications
08000021	Due to local control, data cannot be transferred or saved to applications
08000022	Due to the current device status, data cannot be transferred or saved to applications
08000023	The dynamic condition of Object dictionary generates error or Object Dictionary does not exist