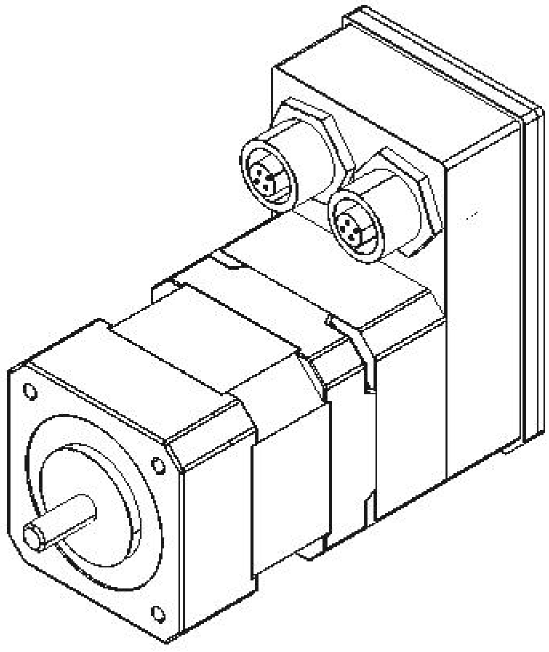
PUSIROBOT

CQPUSI ROBOT CONTROL SYSTEM

User Manual

PMC007BXS Series

Miniature Integrated Stepper Motor Controller



1. Version Control

1） Update Records

Date	Author	Version	Remark
2014-10-19	huangcheng	V0.1.0	Initial
2014-11-25	Liur	V0.1.1	Fix typo
2014-12-08	huangcheng	V0.1.2	Add system control objects; Add BOOT and RESET signal descriptions to signal interface J1, merge external emergency stop enable and external emergency stop trigger mode objects into one object, modify tool screenshots and add related function descriptions.
2015-5-6	huangcheng	V0.1.3	PDO function supplement
2016-6-15	wentao	V0.1.4	Add senseless stall detection
2016-9-22	Liur	V0.1.5	Add function description for close loop
2016-11-7	huangcheng	V0.1.6	Add external stop3
2016-11-25	wentao	V0.1.7	Close loop function supplement
2017-10-8	liur	V0.1.8	Add pvtb mode, update feature & function
2018-6-15	jiawei	V0.1.9	Migrated from pmc007cxsx
2018-7-23	hc	V0.2.0	Modify instruction
2018-09-27	hc	V0.2.1	1、 Add analog input function 2、 Add step notify related objects
2018-11-30	Tanlu	V0.2.2	Supplement description for Ext port
2018-12-3	huangcheng	V0.2.3	Add sensor type parameter description
2019-02-25	huangcheng	V0.2.4	1、Functional description of adding PV/PP mode 2、Adding analog location function description
2019-05-12	liur	V0.2.5	1. Add absolute encoder support; 2. Add sensorness stall detection; 3. Add smooth mode; 4. Extend to 48V supply range.
2019-8-9	huangcheng	V0.2.6	1. Add driver mode settings 2. Add power down behavior settings
2020-3-10	chenya	V0.1.2	Migrate from PMC007Cx
2021-10-9	yj	V0.1.4	Update some details about commands
2022-3-3	Wangw	V0.1.5	Modify register description
2023-5-5	Tony	V0.1.6	Remove step notify register

Catalog

1 Introduction

1.1 Statement of intellectual property right

PMC007CxSxP 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.

PMC007BXS 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

PMC007BXS is a kind of miniature integrated stepper motor microstepping controller, which can be directly installed in the rear of 42/57/86 series stepper motor. The series controller provides a variety of models which can be chosen based on bus control of CAN and different current value. It is easy to achieve industrial control network of as many as 32 nodes, which can achieve closed-loop control based on encoder according to the requirements of user. PMC007BXS adopts industrial standard CANOPEN DS301 control protocol, which not only greatly simplifies the complexity of the upper layer control system, but also maximally reserves flexibility of control, and is suitable for all kinds of high precision, wide range of industry using.

2.2 Features

- ✓ Wide range of 12-48V single voltage supply
- ✓ Output current 0.3A ~ 6A. Adjustable phase current by commands
- ✓ Automatic control of S curve acceleration and deceleration
- ✓ Support Position/Velocity/PP/PV/PVT/SP/Analog Position/Analog velocity etc. motion mode
- ✓ Support 0/2/4/8/16/32/64/128/256 microstepping resolution
- ✓ Suitable for 4/6/8 lines of 2 phase stepper motor
- ✓ Solenoid brake control function
- ✓ Support sensorless stall detection
- ✓ Support 200-4000CPR incremental encoder;
- ✓ Support SSI/BISS multi-turn absolute encoder;
- ✓ Miniature size 42mmx42mmx18mm
- ✓ Precision aluminum shell, conducive to the protection and heat dissipation
- ✓ Automatic over-temperature, over-current, under-voltage and overvoltage protection

2.3 Production & Ordering Information

In order to serve you quicker and better, please provide the product number in following format when ordering PMC007BXS:

PMC 007 B 3 S(E)P 2 I

I: I:Incremental; M:Multi-turn; S:Single-turn

2: 1:External encoder support; 2:Brake control support; 3-6: Ain option

3: Peak Current: 3=3A; 6=6A

B: Bus: B=MODBUS RTU

007: Model

PMC: Pulse Motor Controller

Remark:

E: Closed-loop type.

P: Enhancement type.

Please be sure to contact the sales staff to confirm whether the required model is in a normal state of supply before placing an order.

3 Connector Description

3.1 Terminal port location

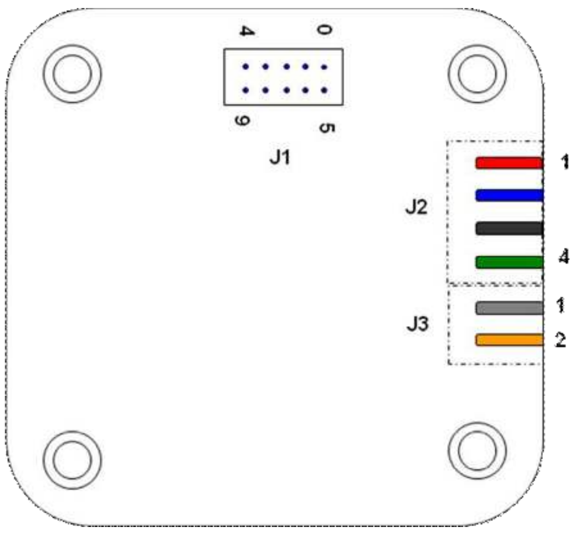


Figure 3-1

3.2 Motor connection J2

Pin no:	1	2	3	4
Designator:	M10	M11	M20	M21

Description:

M10, M11: the stepper motor phase A

M20, M21: the stepper motor phase B

WARNING: Incorrect connection of power or phase will permanently damage the controller! (Corresponding to red, blue and black, green line order in closed-loop type) .

3.3 Power connection J3

Pin no:	1	2
Designator:	GND	VCC

Description:

VCC: Supply voltage, 12~48VDC.

GND: Supply voltage ground.

Remark:

1. When the current exceeds 3A, It is recommended to connect an electrolytic capacitor at least 1000uF near the J3 interface.

2. Hot plug is prohibited, which may damage the controller permanently!

3.4 Signal connection J1

Pin no:	0	1	2	3	4
Designator:	GND	Coil+	Coil-	RXD	TXD
Pin no:	5	6	7	8	9
Designator:	DVDD	AIN/EXT2	EXT1	GPIO8	FSET

Description:

DVDD: Controller voltage output(+5V). Maximum current is 100mA.

GND: Controller digital ground.

EXT1: External limit switch signal input 1, 0~24V.

AIN: Analog input for adjusting speed, 0~3.3V, (optional4-20mA).

GPIO8: Digital input and output,0~3.3V.

FSET: Factory reset input,0~3.3V. Low level effectiveness.

TXD: RS232/RS485 bus transmitting signals.

RXD: RS232/RS485 bus transmitting signals.

Coil+: Solenoid valve/brake positive control terminal, its voltage is equal to the voltage of power supply VCC; or encoder interface;

Coil-: Solenoid valve/brake negative control terminal; or encoder interface;

WARNING: The voltage of all signal ports must be between -0.3V~+5.3V, otherwise, it may cause permanent damage to the controller.

3.5 MODBUS Network Operation

Modbus/RTU have a common physical layer with a standard RS-232 or RS-485 that can be configured with 1~32 slave addresses; a topologically constructed RS-422/485 network, typically with a terminal resistance of 120 ohms paralleled at the last slave device. Modbus/RTU support full-duplex and half-duplex connection mode, usually we recommend to build RTU communication network in full-duplex connection mode.

Note: it is recommended to use the MODBUS RTU bus specified 120 ohm shielded twisted pair, and the ends of the twisted pair are required to connect a 120 ohm termination resistor. In addition, The PTA2C in the diagram is a USB-RS485 conversion module provided by the third party.

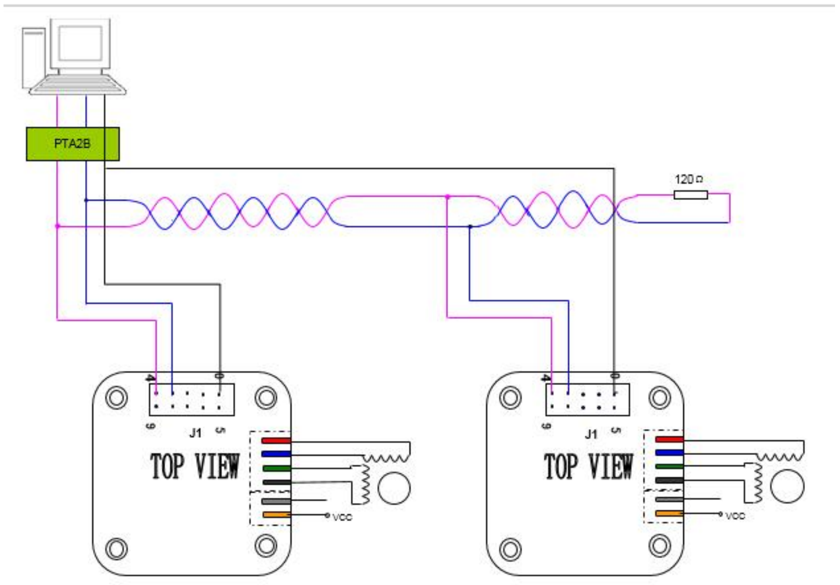


Figure 3-2

3.6 Limit switch connection

It is recommended to test the level changes and triggers by means of a mechanical switch before connecting the optocoupler.

There is a dedicated pin Ext1 in PMC007BXS controller, which are used to connect the external limit(zero point, home position) switch. The trigger mode of each pin can be selected by the instruction. Default trigger mode is falling edge trigger. At this time, the level of the EXT1 is from low level to high level because the Ext1 pin integrates a fixed drop-down resistor and a reverse buffer as shown below Figure 3-3.

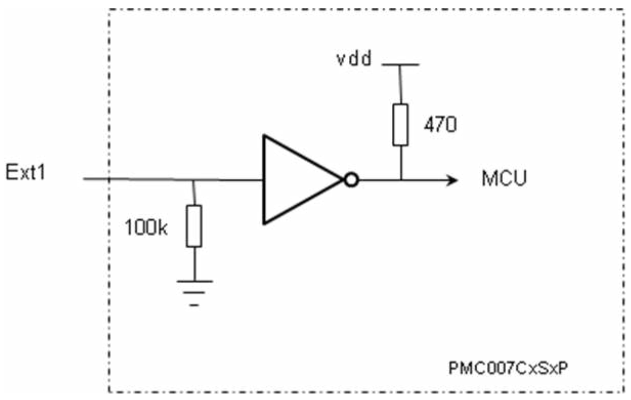


Figure 3-3

The input level range of EXT1 is 0~24V. When input voltage exceeds 3V, it is considered as a high level. The input port can be directly connected to the 5~24V PNP output type sensor, as shown in figure 3-4 left. For NPN or sensors which is open-collector output type, A 1Kohm pull-up resistor is required between the power supply and the Ext1 pin, as shown in figure 3-4 right.

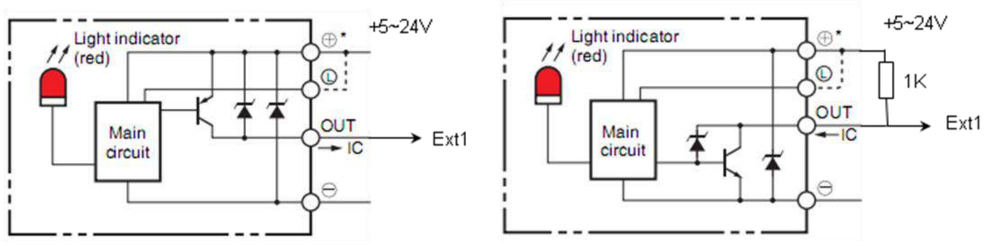


Figure 3-4

When using U-slot optocoupler, the transmitter terminal can be directly connected to GPIO8 and GND, and the collector of the receiver is connected to DVDD, Emitter is connected to EXT1, as shown in Figure 3-5 below.

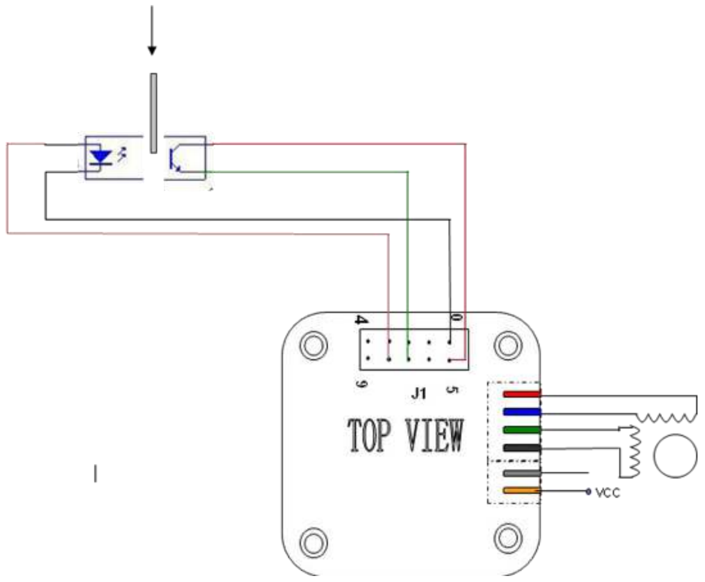


Figure 3-5

3.7 The second limit switch is connected

The AIN and GPIO8 pin of the PMC007BXS controller can also be multiplexed into two other limit switch inputs, but its acceptance voltage should not exceed 5V. When using 24V DC three wire NPN open-collector output type proximity switch, such as OMRON E2EC/ X□C□or E2E-X□D1S series. The connection is shown in the following figure and AIN and GPIO8 pin needs to be configured to pull up enable. Since the input port of the controller can only accept the 5V voltage range, Therefore, the 24V DC three line NPN open-collector proximity switch or 24V PNP type proximity switch cannot be connected to the controller.

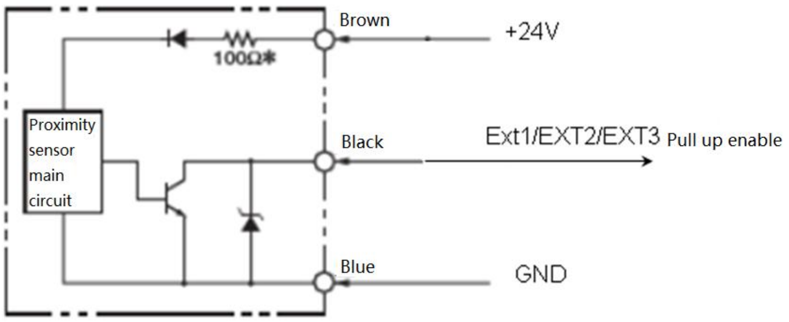


Figure 3-6

If the normal U-slot optocoupler is connected, the GPIO8 port can be used to drive the emitter LED. Set Ext2 port to pulldown enable if falling edge trigger mode is chose as shown Figure 3-7 left, set Ext2 port to pullup enable if rising edge trigger mode is chose as shown Figure 3-7 right.

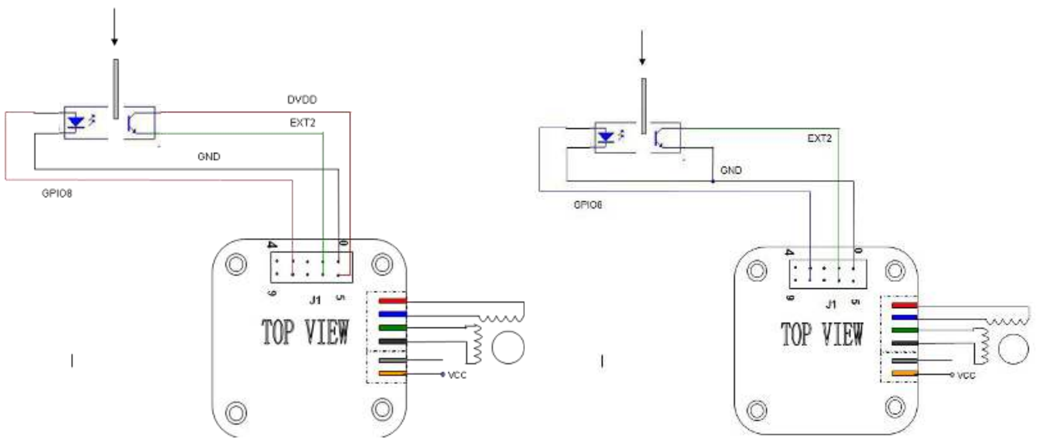


Figure 3-7

If the PNP or NPN proximity sensors with internal circuits are used, the care should be taken for choosing trigger mode. For figure 3-8 left PNP sensor, set Ext2 port to falling edge trigger mode and pulldown enable. For giure 3-8 right NPN sensor, set Ext2 port to rising edge trigger mode and pullup enable.

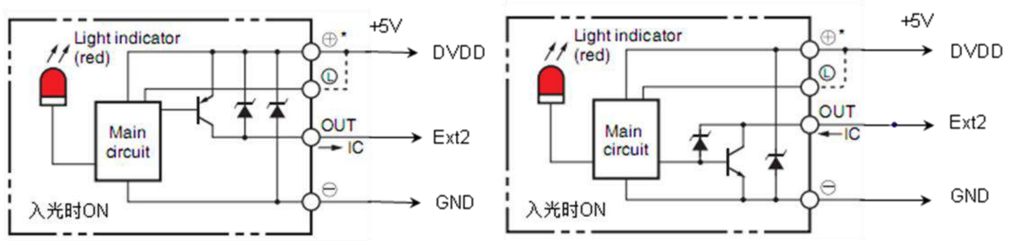


Figure 3-8

3.8 Mechanical switch connection

When using mechanical button switch or relay contact as the limit, for EXT1, the connection mode is as following figure left, using the falling edge trigger mode. For EXT2, the connection mode is as following figure right, enabling internal pull-up resistance, using falling edge trigger mode, as shown in figure 3-9 below.

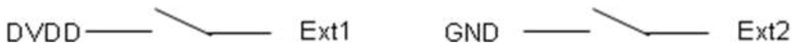


Figure 3-9

3.9 Analog Adjusting Speed

PMC007BXS controller can use analog adjusting speed control function in offline mode. In this application, the AIN pin is used as the analog input port, as shown in Figure 3-6 below. It can also be directly connected to the external input voltage, whose range is 0~3.3V. When using PLC or other industrial control equipment to output 4 ~ 20mA analog control, special comments are needed to distinguish versions.

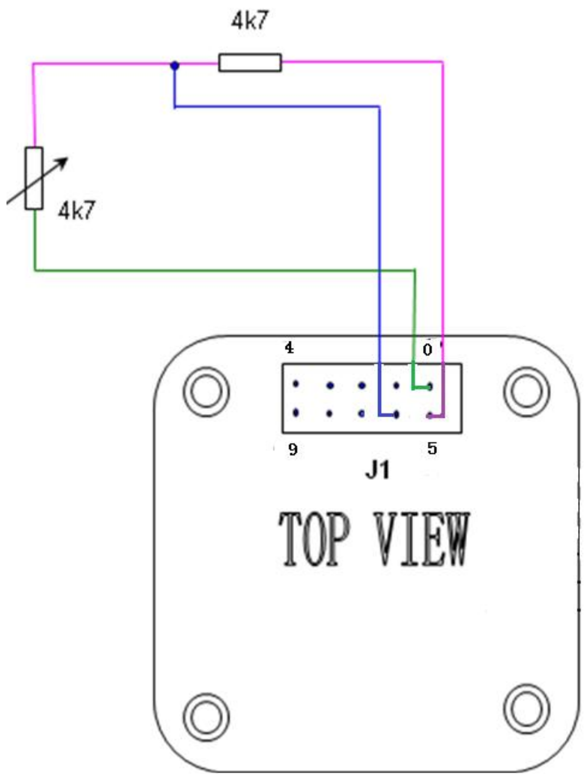


Figure 3-6

3.10 Solenoid valve / brake connection

PMC007BXS controller supports direct control of inductive loads such as solenoid valves, electromagnets, electromagnetic brakes, and DC motors. As shown in following figure, connect the load to the Coil+ and Coil- pins on the controller J1. The output voltage is the same as the input voltage of the controller, and the maximum output current is 800mA. In order to reduce the working temperature of the load coil, the controller supports the PWM dynamic voltage adjustment function, and the user can modify the output voltage in real time by command

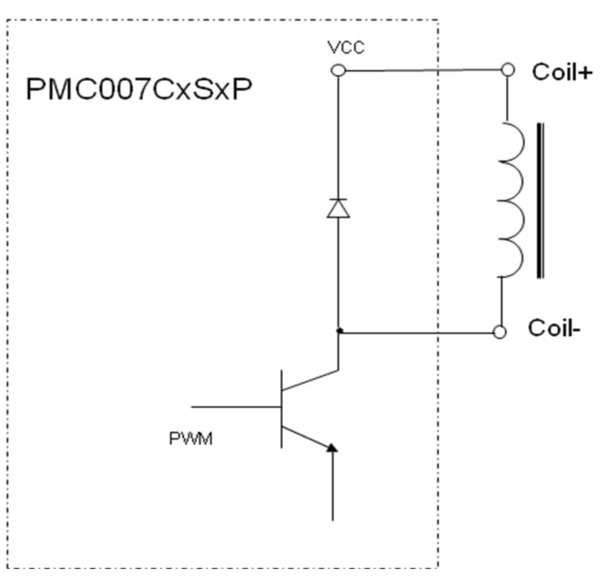


Figure 3-7

3.11 Factory Reset

When the PMC007BXS controller performs a problematic user defined program, or when users accidentally overwrite the controller baud rate, the communication interface may lose response. In this case, if it is still unable to response after repower up, you can use this function to restore factory configuration. Connect the FSET of J1 to GND at least 5ms, and then repower up. The controller is automatically restored to the factory configuration, including the motor parameters, but the user's custom program will be reserved for debugging analysis.

4 Specific application description

4.1 Multi-axis Interpolation

The PMC007 controller can be configured as PVT motion mode. In this mode, the controller uses PUSIROBOT unique cubic spline interpolation optimization algorithm. In the same time coordinate, control the position and speed of the multi axis accurately, so that the end mechanism can realize the trajectory of the line , arc and complex curve, this is an important feature in robot arm or multi-axis application as shown in Figure 4-2 below.

The operation method of PVT mode is described in detail in 5.11.

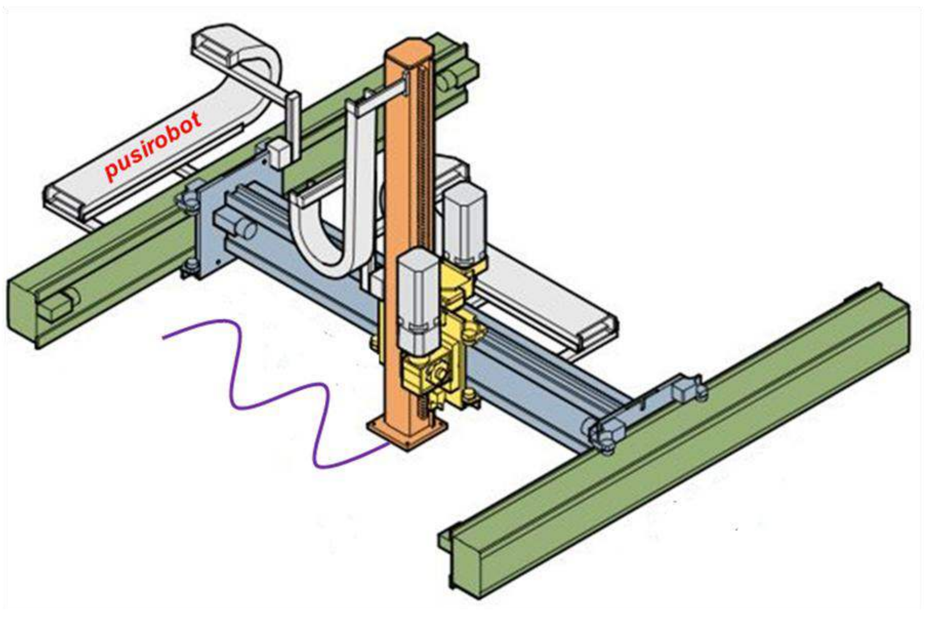


Figure 4-2

In the application of literary creation, such as floating ball matrix, clock matrix, umbrella matrix, etc. In order to present a holistic view, it is necessary to synchronize hundreds of axes. The PMC007 controller uses a special optimization algorithm, which can greatly reduce the bus load and improve the real-time response.

4.2 Drive mode

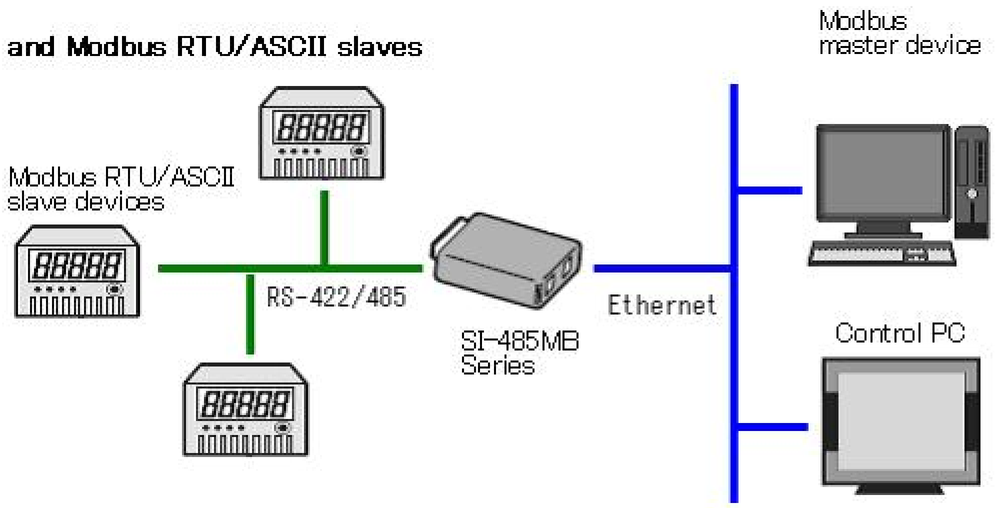
PMC007BXS supports both high-speed mode and low-speed mute mode. The controller is in high-speed mode by default.When the user is sensitive to noise and works at low speed, low speed mute mode can be selected. In high speed mode, the power saving switch can be turned on to reduce the heat generation of the motor. The parameters of driver mode are set by debugging tool for the first time, and the setting method is described in 7.3.3.2.

5 MODBUS communication

5.1 MODBUS introduction

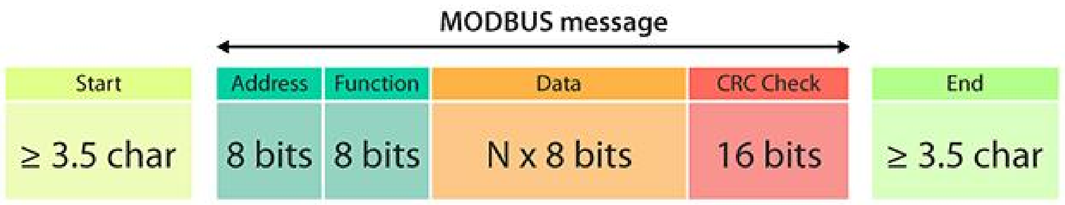
Modbus protocol, which is a bus protocol that allows the master station and one or more slave stations to share data, consists of 16-bit registers. The master can read and write individual or multiple registers.

the standard Modbus port on the controller is using a RS-232 compatible serial interface that defines connectors, wiring cables, signal levels, transmission baud rates, and parity. Controller communication uses master-slave technology, that is, the host can start data transmission, called query. Other devices (slave) return the response to the query or process the actions required by the query. Host equipment shall include master processors, programmers and PLC. The slave includes a programmable controller, a servo drive and a step drive. MODBUS RTU network structure is shown below:



5.2 MODBUS frame structure

Modbus/RTU is a master-slave technique, and the CRC verification range is from the device address bit to the data bit; the detailed message format of each function code, please see appendix. Modbus/RTU message frames are as follows:



as a master-slave communication technology, data can be transmitted between a master station (e.g.: PC) and 32 substations (e.g.: PMC007Bx). The following protocols must be observed between master and slave stations:

1) All information transmitted on the RS485 communication loop will be initialized and controlled by the master station;

2) No communication can begin at a sub-station;

3) All communications in the RS485 loop are transmitted in frame format;

4) If the master or sub-station receives a frame format containing unknown commands, it does not respond;

5.3 MODBUS communication configuration

PMC007 is set to 1 node ID and baud rate of 9600 by default, and users can modify the settings through the supporting MODBUS master debugging tool.

5.3.1 Node ID

Object name	Node ID
Instruct address	0x2028

Object type	U8, rw
Range	1-127
Storage type	ROM
Default value	5

5.3.2 Baud rate

Object name	Baud rate
Instruct address	0x2002a/0x2002b
Object type	U8, rw
Range	4800/9600/19200/38400/51200/115200
Storage type	ROM
Default value	9600

5.3.3 Group ID

Object name	Group ID
Instruct address	0x200e
Object type	U8, rw
Range	1-127
Storage type	ROM
Default value	0

In a MODBUS RTU network, the function remains.

5.4 System information acquisition

5.4.1 Device node name

Object name	Device node name
Instruct address	0x101a..21
Object type	string, ro
Range	-
Storage type	ROM
Default value	-

5.4.2 Hardware version

Object name	Hardware version
Instruct address	0x1022..23
Object type	string, ro
Range	-
Storage type	ROM
Default value	-

5.4.3 Software version

Object name	Software version
Instruct address	0x1024..25
Object type	string, ro
Range	-
Storage type	ROM
Default value	-

5.4.4 System control

Object name	System control
Instruct address	0x200f
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.

5.5 Motor control parameters

5.5.1 Error status

Object name	Driving state
Instruct address	0x6000
Object type	U8, rw
Range	bit
Storage type	RAM
Default value	0

Driver state are defined as follows:

Bit0: TSD, over temperature shutdown

Bit1: AERR, coil A error

Bit2: BERR, coil B error

Bit3: AOC, A over current

Bit4: BOC, B over current

Bit5: UVLO, low voltage fault

Write 1 to the corresponding bit, the corresponding error state will be cleared.

5.5.2 Controller status

Object name	Controller status
Instruct address	0x6001
Object type	U8, rw
Range	bit
Storage type	RAM
Default value	0

Controller status is defined as follows:

Bit0: External stop 1
Bit1: External stop 2
Bit2: Stall state
Bit3: busy state
Bit4: External stop 3
Bit5: The FIFO of PVT Mode 3 is empty
Bit6: FIFO Lower bound of PVT Mode 3
Bit7: FIFO upper limit of PVT mode 3

Each bit except for busy state can be written 1 to clear the response state.

5.5.3 Rotation direction

Object name	Rotation direction
Instruct address	0x6002
Object type	U8, rw
Range	0, 1
Storage type	RAM
Default value	1

The value of the rotation direction is defined as follows:

0: forward
1: backward

5.5.4 Maximum speed

Object name	Target speed (pps)
Instruct address	0x6003..4
Object type	S32, rw
Range	-200000 ~ +200000
Storage type	RAM
Default value	0

Note: the speed is a signed variable. Positive represents that the direction is 1, and negative represents that the direction is 0. So in the displacement mode it is recommended to set the speed firstly , and then set the direction.

5.5.5 Relative displacement command

Object name	Relative displacement command
Instruct address	0x6005..6
Object type	U32, rw
Range	0x0-0xFFFFFFFF
Storage type	RAM
Default value	0

Write step number, then the controller will control the motor rotate a given number of steps which is calculated based on the current microstepping settings at the setting direction, speed and acceleration.

When the controller is in the busy state, the step command will be ignored. When the error state and the other bits in the controller status are valid, you need to clear up before you start the step command.

In closed-loop mode, the input unit is 1/4 of the encoder resolution. For example, If CPR=500, the motor would rotate a circle when input 2000.

In the closed loop mode of absolute encoder, the input unit is the same counting unit of encoder, for example, the accuracy is 12 bits, so the motor rotates one turn when the input is 4096.

5.5.6 Absolute displacement command

Object name	Absolute displacement command
Instruct address	0x6044..45
Object name	S32, rw
Range	Incremental or single cycle absolute value encoder:-2^31 ~ (2^31-1) Multi-circle absolute value encoder:2^(r+11)~(2^(r+11)1), r is the encoder precision, for example, r=12 represents 12-bit encoder.
Storage type	RAM
Default value	0

Absolute displacement command gives the target position, then the controller will automatically calculate the direction and the required step number, and control the motor rotate to the target position at the setting speed and acceleration.

In the open loop mode, the number of steps is calculated in the current subdivision setting.

In closed-loop mode, the input unit is 1/4 of the encoder resolution. For example, If CPR=500, the motor would rotate a circle when input 2000.

In the closed loop mode of absolute encoder, the input unit is the same counting unit of encoder, for example, the accuracy is 12 bits, so the motor rotates one turn when the input is 4096.

5.5.7 Stop stepping command

Object name	Stop stepping command
Instruct address	0x6053
Object type	U8, rw
Range	0
Storage type	RAM
Default value	0

The command immediately terminates the motor running, regardless of the current mode is location mode or speed mode.

5.5.8 Operation mode

Object name	Operation mode
Instruct address	0x6007
Object type	U8, rw
Range	0, 1, 2
Storage type	RAM
Default value	0

The value of the motor operation mode is defined as follows:

0: Position mode

1: Speed mode

2: PVT mode

3: Encoder following mode (special version of firmware)

4: PP (Profile Position) mode (including analog positioning)

5: PV(Profile Velocity) mode

When the operation mode is switched from the speed mode to the position mode, the motor will stopping at the setting deceleration.

5.5.9 Start speed

Object name	Start speed(Unit: pps)
Instruct address	0x6008
Object type	U16, rw

Range	0-0xFFFF
Storage type	ROM
Default value	600

5.5.10 Stop speed

Object name	Stop speed(Unit: pps)
Instruct address	0x6009
Object type	U16, rw
Range	0-0xFFFF
Storage type	ROM
Default value	600

If the start speed and stop speed jump directly, for example, jump from 0 to the maximum speed of starting speed and then accelerate or decelerate from maximum speed to stop speed and then jump to 0, so the start and stop speed cannot be set to 0.

5.5.11 Acceleration coefficient

Object name	Acceleration coefficient
Instruct address	0x600a
Object type	U8, rw
Range	0-8
Storage type	ROM
Default value	8

5.5.12 Deceleration coefficient

Object name	Deceleration coefficient
Instruct address	0x600b
Object type	U8, rw
Range	0-8
Storage type	ROM
Default value	8

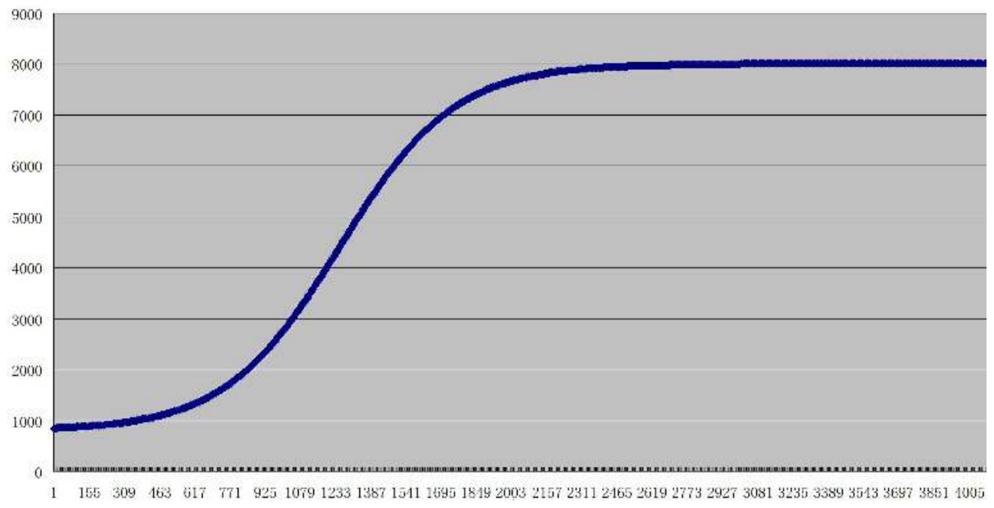


Figure 5-1

The PMC007BXS controller uses S curve acceleration and deceleration. As shown in Figure 5-1, Start speed, stop speed, acceleration and deceleration can be configured separately. There are a total of 8 gears for acceleration and deceleration. The relationship between each gear and the corresponding acceleration value is shown in the following table.

Gear	Acceleration and deceleration value (PPS^2)
0	Acceleration and deceleration cannot be enable
1	77440
2	48410
3	27170
4	21510
5	14080
6	10460
7	6915
8	5210

5.5.13 Microstepping

Object name	Microstepping
Instruct address	0x600c
Object type	U16, rw
Range	0, 2, 4, 8, 16, 32, 64, 128, 256
Storage type	ROM
Default value	0

5.5.14 Maximum phase current

Object name	Maximum phase current
Instruct address	0x600d
Object type	U16, rw
Range	0-6000
Storage type	ROM
Default value	658

The maximum phase current is the supply current to the motor when it is working normally, and is generally set to the rated current of the motor. In some cases, it can be adjusted appropriately, generally with a range of +-20% and not more than 50%. Excessive current will cause the motor to heat up seriously, and long-term operation of the motor may have the risk of demagnetization, affecting the service life of the motor.

The B3 model ranges from 0-3000 in mA and is suitable for motors up to 42 and some 57 motors, and the B6 model is in the range of 0-6000 and is suitable for 57 and 86 motors.

5.5.15 Motor position

Object name	Motor position
Instruct address	0x600e..f
Object type	S32, rw
Range	Incremental or Singleturn Absolute Encoder: -2^31 ~ (2^31-1) Multiturn absolute encoder: -2^(r+11) ~ (2^(r+11))-1), r is the encoder accuracy, such as r=12 means 12-bit encoder
Storage type	RAM
Default value	0

When a stepping order is issued, the controller automatically records the current position which is represented by an signed integer according to the given number of steps. A positive value indicates clockwise rotation, and a negative value indicates a counterclockwise rotation.

In open-loop mode, current position value is calculated by the number of steps, so when users need to change the microstepping, shall read the position information firstly and then change the microstepping, in order to avoid the position conversion error. In closed-loop mode, the 1/4 of the encoder resolution is the unit.

In open-loop mode, when the controller power down, the position information is automatically cleared.

In incremental closed-loop mode, when the controller is powered off, the position where the power is saved can be selected, and the position value of the last power loss will be loaded to the object the next time the power is turned on again.

In the single cycle absolute value closed loop mode, the position information is automatically cleared when the controller is powered off.

In a multi-turn absolute closed-loop mode, the controller reads the encoder position to this object in real time after the controller is powered on.

5.5.16 Calibration zero (absolute value encoder closed loop)

Object Name	Calibration zero
Instruct address	0x6087..88
Instruct address	S32, rw
Object type	Single cycle absolute value encoder: -2^31 ~ (2^31-1) Multi-loop absolute value encoder: -2^(r+11) ~ (2^(r+11))-1), r is encoder accuracy, such as r ≥ 12 for 12-bit encoder
Range	RAM
Storage type	0

The closed-loop controller of absolute encoder supports this object. When the user writes a value of motor position (0x600C object), the value of calibration zero will be calculated automatically, and the motor position read by the user = encoder position will be calibrated zero.

5.5.17 Encoder position (absolute value encoder closed loop)

Object Name	Calibration zero
Instruct address	0x6089..8a
Instruct address	S32, rw
Object type	Single cycle absolute value encoder: -2^31 ~ (2^31-1) Multi-loop absolute value encoder: -2^(r+11) ~ (2^(r+11))-1), r is encoder accuracy, such as r ≥ 12 for 12-bit encoder
Range	RAM
Storage type	0

The controller reads the actual position value of the encoder.

5.5.18 Current reduction

Object name	Current reduction coefficient
Instruct address	0x6010
Object type	U8, rw

Range	0-3
Storage type	ROM
Default value	2

The value is defined as follows:

0: Decay 0%;

1: Decay 50%

2: Decay 75%

3: Decay 87.5%

0-3 corresponds to 4 gears of 100%, 50%, 25% and 12.5% idle current supply, the ratio is based on the percentage of the maximum phase current of 600Bh. The default is 50% for 2nd gear, and when the motor is idle, it will switch to idle current power supply.

5.5.19 Motor enable

Object name	Motor enable
Instruct address	0x6011
Object type	U8, rw
Range	0, 1
Storage type	RAM
Default value	1

The value of the motor is defined as follows:

0: Offline

1: Motor enable

After setting the offline, controller immediately release the control of motor and the current step command is terminated, phase current is reduced to 0. All subsequent step command issued by host computer cannot be processed, until the user reset motor enable.

5.5.20 Stall set（Open-loop）

Object name	After set stall, whether motor stop or not.
Instruct address	0x6043
Object type	Record
Range	0-1
Storage type	ROM
Default value	0

When set to 1, the power opportunity stops after blocking turn, and the motor does not stop when it is 0.

5.5.21 Stall parameters（Open-loop）

Object name	Set stall detection parameters

Instruct address	0x6039
Object type	U16, rw
Range	bit
Storage type	ROM
Default value	0

The values of detection parameters are defined as follows:

Bit0~6: Blocking threshold, with symbol number;

Bit8~15 : Reservation;

PMC007BXS controller uses the reverse EMF of two-phase winding to realize sensorless blocking detection. Its accuracy is affected by many factors, such as current, subdivision, voltage, motor parameters and so on, especially the motor speed and phase inductance. The range of blocking threshold is usually set between-10 and 10.

5.5.22 Real time Speed（close loop only）

Object name	Read real time motor speed
Instruct address	0x607f..80
Object type	S32, ro
Range	-300000 ~ +300000
Storage type	ROM
Default value	0

Real-time speed is a signed variable, positive time represents direction 1, negative time represents direction 0.

5.6 External emergency stop

The PMC007BXS controller provides a special limit switch input port EXT1, which can be used for emergency stop or home position search function.

When the emergency stop is enabled, if the corresponding input pin detects effective trigger edge, controller immediately lock the motor and stop responding to any step command. User can read the status of controller, and check which one input pin trigger the emergency stop. The controller will continue to respond to the new step command, only after the user clears the corresponding status bit.

5.6.1 External emergency stop enable

Object name	External emergency stop
Instruct address	0x6013
Object type	U8, rw
Range	Bit
Storage type	ROM
Default value	0

this represented by 1bit that each external emergency stop enable. 0 indicates the prohibition, and 1 indicates enabling. Its definition is as follows:

bit0: External emergency stop 1 enable settings

bit1: External emergency stop 2 enable settings

bit4: External emergency stop 3 enable settings

5.6.2 The trigger mode of external emergency stop

Object name	The trigger mode of external emergency stop
Instruct address	0x6014
Object type	U8, rw
Range	Bit
Storage type	ROM
Default value	0

trigger mode of each external emergency stop is represented by 1bit. 0 indicates falling edge trigger, and 1 indicates rising edge trigger. Its definition is as follows:

bit0: The trigger mode of external emergency stop 1

bit1: The trigger mode of external emergency stop 2

bit4: The trigger mode of external emergency stop 3

5.6.3 Sensor type

Object name	Sensor type
Instruct address	0x6015
Object type	U8, rw
Range	0-1
Storage type	ROM
Default value	0

Its definition is as follows:

0: when the trigger mode is configured as the rising edge, the controller is configured as the internal pull-down resistance, When configured as a falling edge, the controller is configured with an internal pull-up resistor; typically used for an NPN type of sensor;

1: When the trigger mode is configured as the rising edge, the controller is configured as the internal pull-up resistance, when the trigger mode is configured as the falling edge, the controller is configured with the internal pull-down resistance, typically used for an PNP type of sensor;

The trigger delay of external emergency stop can be modified by 0x601A object. The controller delays the setting time after the detected edge signal, and then detects whether its level state is correct or not, then triggers the motor to stop urgently, otherwise the motor will continue to rotate.

Object name	EXT1/EXT2/EXT3 stabilize delay (ms)

Instruct address	0x6042
Object type	U8, rw
Range	0~200
Storage type	ROM
Default value	100

5.7 General IO port

The PMC007BXS controller provides 7 general purpose IO (GPIO) ports, 2 external emergency stop input (EXT) ports and 2 encoder input (ENC) ports.

5.7.1 General IO port set

1、The direction of IO port

Object name	the direction of IO port
Instruct address	0x602e
Object type	U16, rw
Range	bit
Storage type	ROM
Default value	0

The direction of each IO port is represented by 1bit. 0 represents input, and 1 represents output. The meaning of each bit is as follow:

Bit0: GPIO1

Bit1: GPIO2

Bit2: GPIO3

Bit3: GPIO4

Bit4: GPIO5

Bit5: GPIO6

Bit6: GPIO7

Bit7: EXT1

Bit8: EXT2

Bit9: EXT3/ENC1

Bit10: ENC2

Bit11: GPIO8

Among them, the direction of emergency stop input port and encoder input port is fixed as input port, which cannot be configured.

Note: GPIO0~GPIO7 does not lead to the controller interface. It is only used for off-line programming.

3、IO port configuration

Object name	IO port configuration
Instruct address	0x602f..30
Object type	U32, rw
Range	0-0x3fffff

Storage type	ROM
Default value	0

Each port is configured by 2 bits. If the IO port is configured as a input port, the meaning of the value is as follows:

0: FLOATING

1: IPU

2: IPD

3: AIN

If the IO port is configured as a output port, the meaning of the value is as follows:

0: OD

1: PP

The definition of the IO port configuration is defined as follows:

Bit1-0: GPIO1

Bit3-2: GPIO2

Bit5-4: GPIO3

Bit7-6: GPIO4

Bit9-8: GPIO5

Bit11-10: GPIO6

Bit13-12: GPIO7

Bit15-14: EXT1

Bit17-16: EXT2

Bit19-18: EXT3/ENC1

Bit21-20: ENC2

Bit23-22: GPIO8

5.7.2 General IO port value

Object name	General IO port value
Instruct address	0x6031
Object type	U16, rw
Range	bit
Storage type	RAM
Default value	0

The value of each IO port is represented by 1bit, 0 indicates a high level, 1 indicates a low level, and writing value to the port is not valid for the input port. The meaning of each bit is as follows:

Bit0: GPIO1 value

Bit1: GPIO2 value

Bit2: GPIO3 value

Bit3: GPIO4 value

Bit4: GPIO5 value

Bit5: GPIO6 value

Bit6: GPIO7 value
Bit7: EXT1 value
Bit8: EXT2 value
Bit9: EXT3/ENC1 value
Bit10: ENC2 value
Bit11: GPIO8 value

5.8 Closed-loop control

PMC007BXS supports 200-2000CPR incremental photoelectric encoder and uses PID to realize closed loop control. The following is a detailed description of the closed-loop parameters.

5.8.1 Encoder resolution

Object name	Encoder resolution
Instruct address	0x6054
Object type	U16, rw
Range	Incremental encoder closed loop:200, 400, 500, 600, 800, 1000, 1200, 1600, 2000, 4000 Absolute encoder closed loop:12
Storage type	ROM
Default value	1000, 12

Note: After changing the encoder resolution, the power of controller must be re-energize.

5.8.2 KP parameter

Object name	KP parameter
Instruct address	0x6057
Object type	U8, rw
Range	1-255
Storage type	ROM
Default value	8

This parameter affects the transient response characteristic of the system.

5.8.3 KI parameter

Object name	KI parameter
Instruct address	0x6058
Object type	U8, rw
Range	1-255
Storage type	ROM

Default value	4

This parameter affects the cumulative error characteristics of the system.

5.8.4 KD parameter

Object name	KD parameter
Instruct address	0x6059
Object type	U8, rw
Range	1-255
Storage type	ROM
Default value	8

This parameter affects the transient response characteristic of the system.

5.8.5 Pre-filtering parameter

Object name	Pre-filtering parameter
Instruct address	0x605a
Object type	U8, rw
Range	1-128
Storage type	ROM
Default value	32

This parameter affects the speed characteristics of the system. When speed or microstepping is high, it is recommended to use larger parameter values.

5.8.6 Post-filtering parameter

Object name	Post-filtering parameter
Instruct address	0x605b
Object type	U16, rw
Range	1-255
Storage type	ROM
Default value	16

This parameter is reserved for the time being.

5.8.7 Stall length parameter

Object name	Stall length parameter
Instruct address	0x605c
Object type	U16, rw
Range	1-255
Storage type	ROM

Default value	7

The larger the threshold value for judging the stalled rotor, the less sensitive it is.

5.8.8 Torque ring enable

Object name	Torque ring enable
Instruct address	0x605d
Object type	U8, rw
Range	0-1
Storage type	ROM
Default value	0

When the power torque ring is not used, the PID parameter does not take effect, the controller works in the position loop mode, and the motor will be stopped directly if there is a stalled rotor. After making the torque ring, the closed-loop motor will put the stalled rotor flag (0x6001h[bit2]) in one when it is stalled, and let the motor continue to try to resume operation, that is, after the torque ring is enabled, there will still be a stalled rotor in the motion task or will continue to complete the running task, and this function is only effective with incremental closed-loop control.

5.8.9 Autosave when power is off enable

Object name	Autosave when power is off enable
Instruct address	0x605e
Object type	U8, rw
Range	0-1
Storage type	ROM
Default value	0

The closed-loop takes effect, and the controller automatically detects the power-off of the system after it is enabled, and writes the current motor position into the EEPROM. At the same time, for singleturn and incremental encoder motors, it is necessary to ensure that the motor does not rotate after the power failure, which can be used with the solenoid valve.

5.9 Synchronous position Motion mode

The synchronous positioning motion mode can first set the absolute position and speed of the specified node to run, and then make multiple axes move at the same time through the synchronous start command.

5.9.1 SP speed

Object name	SP speed

Instruct address	0x6047..48
Object type	S32, rw
Range	-2147483648-2147483647
Storage type	RAM
Default value	2

5.9.2 SP position

Object name	SP position
Instruct address	0x6049..4a
Object type	S32, rw
Range	-2147483648-2147483647
Storage type	RAM
Default value	0

5.9.3 SP commands

Object name	SP commands
Instruct address	0000
Object type	U16, rw
Range	0, a, b, c
Storage type	RAM
Default value	0

a: Synchronous operation command, absolute displacement

b: pvt_start PVT boot command

c: pvt_stop PVT stop command

For controllers that need to be synchronized, set to different station numbers, the same group (i.e., the same group ID), and set the motion mode to position mode. The synchronous operation is operated in the absolute positioning mode of position mode, and the speed and position are set to 6047, 6048/6049, 604a respectively. After setting the values for the controllers at different stations, the instructions are sent to the groups to perform the movement by broadcasting.

The broadcast instruction is fixed at 06h to write a single function code, the address is 0, the register number is 0, the fourth byte is the group ID value, the fifth byte is the motion mode, and the synchronous execution value is A

06H instruction(Write a single hold register)

485 address	Register address	Register value
0	0000	010a

00 06 00 00 01 0A 09 8C

09 8C is the crc check code, and there is no response packet after the command is sent.

If you want to stop the synchronization during the synchronization run, you can send a command to stop the task in turn, or you need to set the microstepping again after using the value c.

06H instruction(Write a single hold register)

485 address	Register address	Register value
0	0000	010c

00 06 00 00 01 0C 89 8E

5.10 PVT motion mode

PMC007 supports three PVT control modes, each of which is suitable for different application scenarios.

Mode 1 is a single motion mode. When the controller executes the PVT sequence data written by the host computer, the PVT motion is finished.

Mode 2 is a circular motion mode. PVT motion will be end after dedicated cycle times which is assigned by host.

Mode 3 is a FIFO control mode. The host computer writes the PVT sequence to the controller continuously, the controller takes out PVT data to perform PVT motion.

In addition, the PMC007 support group ID setting, which is used to synchronize two or more nodes start and stop PVT running in the same network. For details about the use process of PVT motion pattern, please refer to the script example of the PUSICAN tool.

5.10.1 PVT Control

Object name	PVT control operation
Instruct address	0x6017
Object type	U8, rw
Range	0-3
Storage type	RAM
Default value	0

0: Stop PVT motion.

1: Start PVT motion.

2: Write the PVT position, speed, and time object data into the queue;

3: Clear all PVT data in the queue.

5.10.2 PVT operation mode

Object name	PVT operation mode
Instruct address	0x6018
Object type	U8, rw
Range	0-2
Storage type	RAM
Default value	0

0: PVT mode 1;

1: PVT mode 2;

2: PVT mode 3;

5.10.3 Max PVT points

Object name	Max PVT points
Instruct address	0x6019
Object type	U16, rw
Range	0-1000
Storage type	RAM
Default value	0

5.10.4 PVT pointer

Object name	PVT pointer
Instruct address	0x601a
Object type	U16, rw
Range	RAM
Storage type	0-1000
Default value	0

5.10.5 PVT mode 1 parameter

1. PVT mode 1 start index

Object name	PVT mode 1 start index
Instruct address	0x601b
Object type	U16, rw
Range	RAM
Storage type	0-1000

Default value	0

2. PVT mode 1 end index

Object name	PVT mode 1 end index
Instruct address	0x601c
Object type	U16, rw
Range	RAM
Storage type	0-1000
Default value	0

5.10.6 PVT mode 2 parameter

1. PVT mode 2 start index at the acceleration stage

Object name	PVT mode 2 start index at the acceleration stage
Instruct address	0x601d
Object type	U16, rw
Range	RAM
Storage type	0-1000
Default value	0

2. PVT mode 2 end index at the acceleration stage

Object name	PVT mode 2 end index at the acceleration stage
Instruct address	0x601e
Object type	U16, rw
Range	RAM
Storage type	0-1000
Default value	0

3. PVT mode 2 start index at the cycle stage

Object name	PVT mode 2 start index at the cycle stage
Instruct address	0x601f
Object type	U16, rw
Range	RAM
Storage type	0-1000

Default value	0

4. PVT mode 2 end index at the cycle stage

Object name	PVT mode 2 end index at the cycle stage
Instruct address	0x0x6020
Object type	U16, rw
Range	RAM
Storage type	0-1000
Default value	0

5. PVT mode 2 cycle times at the cycle stage

Object name	PVT mode 2 cycle times at the cycle stage
Instruct address	0x0x6021
Object type	U16, rw
Range	RAM
Storage type	0-65535
Default value	0

6. PVT mode 2 start index at the deceleration stage

Object name	PVT mode 2 start index at the deceleration stage
Instruct address	0x6022
Object type	U16, rw
Range	RAM
Storage type	0-1000
Default value	0

7. PVT mode 2 end index at the deceleration stage

Object name	PVT mode 2 end index at the deceleration stage
Instruct address	0x6023
Object type	U16, rw
Range	RAM
Storage type	0-1000
Default value	0

5.10.7 PVT mode 3 parameter

1. PVT mode3 FIFO depth

Object name	PVT mode3 FIFO depth
Instruct address	0x6024
Object type	U16, rw
Range	RAM
Storage type	0-1000
Default value	0

2. PVT mode3 FIFO lower limit

Object name	PVT mode3 lower limit
Instruct address	0x6025
Object type	U16, rw
Range	RAM
Storage type	0-1000
Default value	0

In PVT mode 3, once the FIFO depth is less than the set value of this object, and the FIFO lower limit of the controller state object would be set.

3. PVT mode 3 FIFO upper limit

Object name	PVT mode3 upper limit
Instruct address	0x6026
Object type	U16, rw
Range	RAM
Storage type	0-1000
Default value	0

In PVT mode 3, once the FIFO depth is more than the set value of this object, and the FIFO upper limit of the controller state object would be set.

5.10.8 PVT position

Object name	PVT position
Instruct address	0x6027..28

Object type	S32, rw
Range	RAM
Storage type	-2147483648-2147483647
Default value	0

An absolute position which the current PVT point is expected to move.

5.10.9 PVT speed

Object name	PVT speed
Instruct address	0x6029..2a
Object type	S32, rw
Range	RAM
Storage type	2147483648-2147483647
Default value	0

The current PVT point expected speed of motion, unit PPS.

5.10.10 PVT time

Object name	PVT time
Instruct address	0x602b..2c
Object type	S32, rw
Range	RAM
Storage type	0-2147483647
Default value	0

The time is from the last PVT point to the current PVT point, unit ms.

5.11 PP motion mode

The Profile Position Mode will be entered if motion mode 4 is chose, In this mode the trapezoid acceleration is adopted, a set of new parameters can be issued from host, after the control bit is set, the driver will smoothly switch from last set of parameters to the new set of parameters. The description of control bit is refer to section 5.13.3, and the description of control object is refer to section 5.13.1 and 5.13.2.

5.11.1 PP mode parameter 1

5.11.1.1 Acceleration

Object name	acceleration, pps/s

Instruct address	0x6067..68
Object type	U32,rw
Range	ROM
Storage type	>150
Default value	32000

5.11.1.2 Deceleration

Object name	deceleration, pps/s
Instruct address	0x6069..6a
Object type	U32,rw
Range	ROM
Storage type	>150
Default value	32000

5.11.1.3 Start speed

Sub-index 0x03: start speed

Object name	start speed, pps/s
Instruct address	0x606b..6c
Object type	U32,rw
Range	ROM
Storage type	>150
Default value	600

5.11.1.4 Stop speed

Sub-index 0x04: stop speed

Object name	stop speed, pps/s
Instruct address	0x606d..6e
Object type	U32,rw
Range	ROM
Storage type	>150
Default value	600

5.11.2 PP mode parameter 2

PP mode parameter 2 is stored in RAM and cannot be power down preserved.

5.11.2.1 Control word

Sbu-index 0x01: Control word

Object name	Control word
Instruct address	0x6070
Object type	U16, rw
Range	ROM
Storage type	0-0xFFFF
Default value	0

The function description for Control word object (602e,1):

• Bit 4: Start task. Start the task when bit value switch from 0 to 1.

• Bit 5: The task trigged by Bit4 will be immediately executed if this bit is set 1.The task will be executed after last task completed if this bit is set 0.

• Bit 6: The target position (602e,4) is relative position when this bit is set 0, the target position is absolute position if this bit is set 1.

• Bit 8 (Halt): This bit is for PV motion mode, motor will be accelerated to target speed in preferred slope if this bit change from 1 to 0. Motor will be decelerated to zero if this bit change from 0 to 1.

• Bit 9: Motor will change speed after the first target point arrived if this bit is set 1.

5.11.2.2 Pp Status word

Sub-index 0x02: status word

Object name	Pp status word
Instruct address	0x6071
Object type	U16, rw
Range	ROM
Storage type	0-0xFFFF
Default value	0

The function of status word is following:

• Bit 10 : This bit will be set 1 after the final target is arrived.

• Bit 12 : This bit is the ACK bit that will be set after receive new target potion.

Exception: Start a new run task when a run task has not been completed and the next run task should be executed only after that task is completed. In this case, the bit is reset only if the command is accepted and the controller is ready to perform a new running task.

When one running task is enabled and another running task is set, all other running tasks are ignored; to show this, the bit is set.

5.11.2.3 Pp model Running Speed

Subindex 0x03: running speed, the symbol represents the direction of rotation, the positive sign rotates positively, and the negative sign reverses

Object name	Pp running speed
Instruct address	0x6072..73
Object type	S32, rw
Range	ROM
Storage type	-300000- -150, 150-300000
Default value	32000

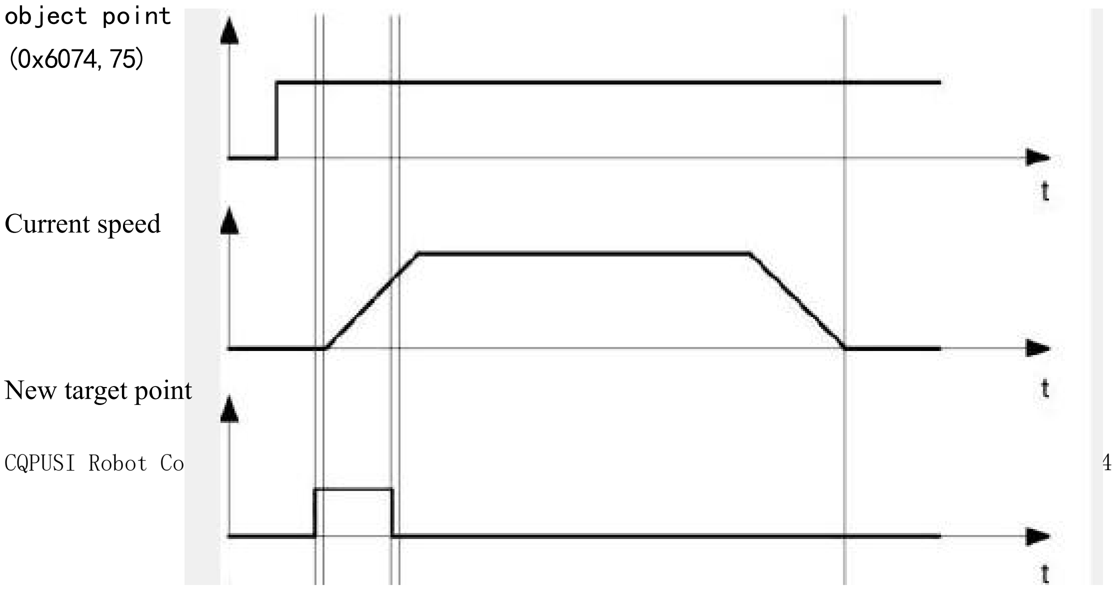
5.11.2.4 Pp model Target Location

Subindex 0x04: PP Model Target Location

Object name	Pp Target location
Instruct address	0x6074..75
Object type	S32, rw
Range	ROM
Storage type	-2^31~2^31
Default value	0

5.11.3 PP model work timing

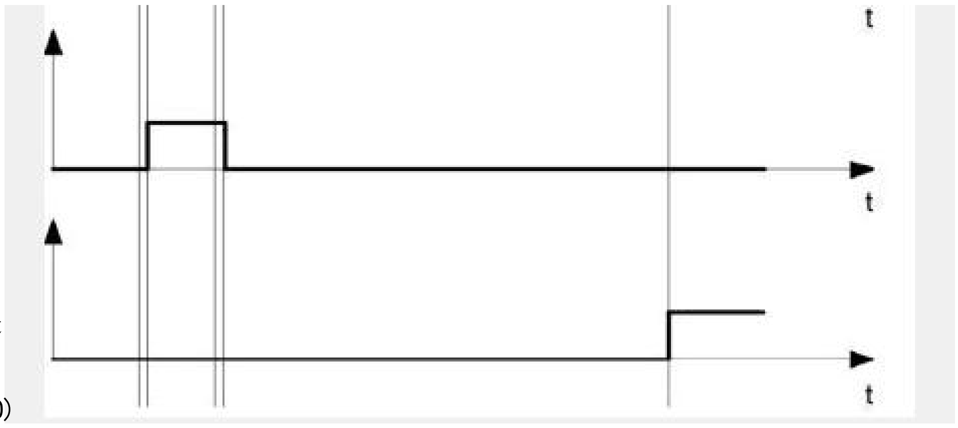
A new target position is set in the target position object (602e, 4). Next, the bit 4 in the control word object (602e, 1) is set for trigger the operation command. If the target location is valid, the controller will reply through the bit 12 in the object status word to locate the start of the operation. When the location is reached, the bit 10 in the status word will immediately set to a "1".



(0x6070 bit 4)

Target point confirmation (0x6071 bit 2)

Reach the target Point (0x6071 bit 10)



Other running commands can be stored in the cache (see point in time 1 in the figure below), and bit 12 in the status word object (602e, 2 sets the target point response) will be set to "0". During the motion to the target position, a second target position can be sent to the controller to prepare for it. At this point, you can reset all parameters, such as velocity, acceleration, deceleration, etc. (point in time 2). If the cache is idle again, the next point in time can enter the queue (point in time 3).

If the cache is full, the new target point will be ignored (point in time 4). If bit 5 in the control word object (602e, 1 bit: "change the target point now") is set, the controller will not use cache when working, and the new running command will be executed directly (point in time 5).

New target point (0x6070 bit 4)

Accept changes immediately (0x6070 bit 5)

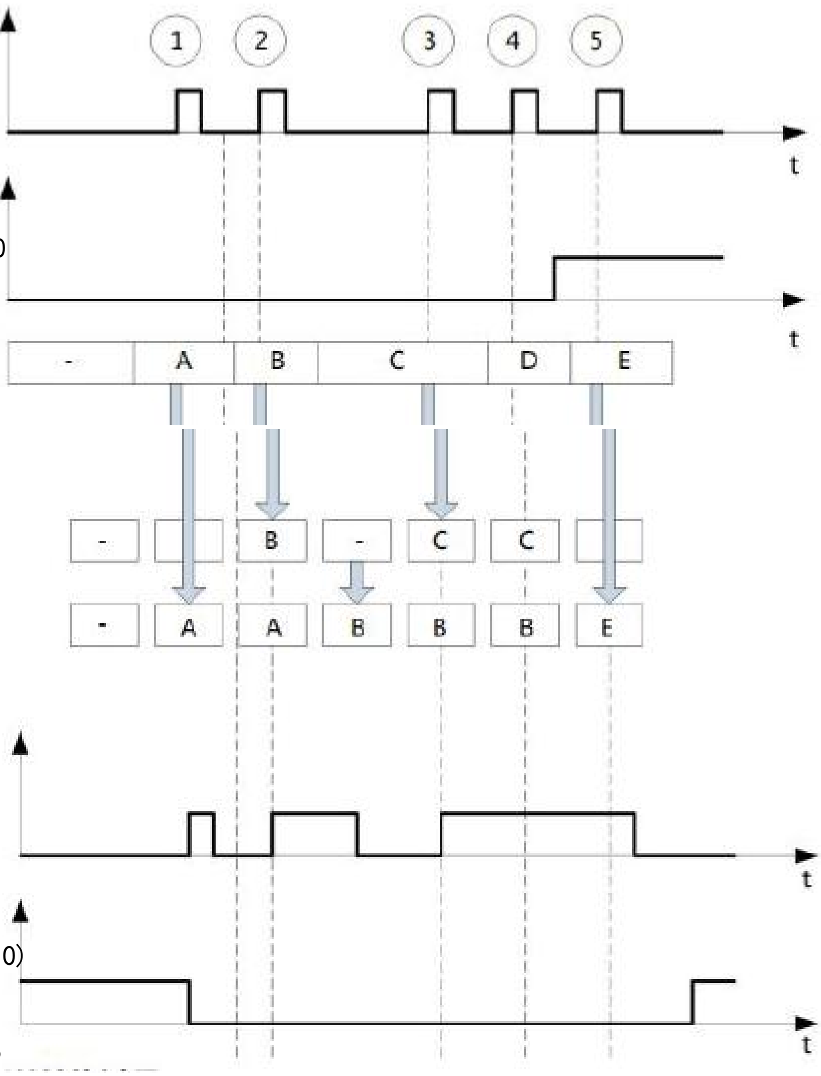
object point (0x6074..75)

Saved target point

Target point to be processed

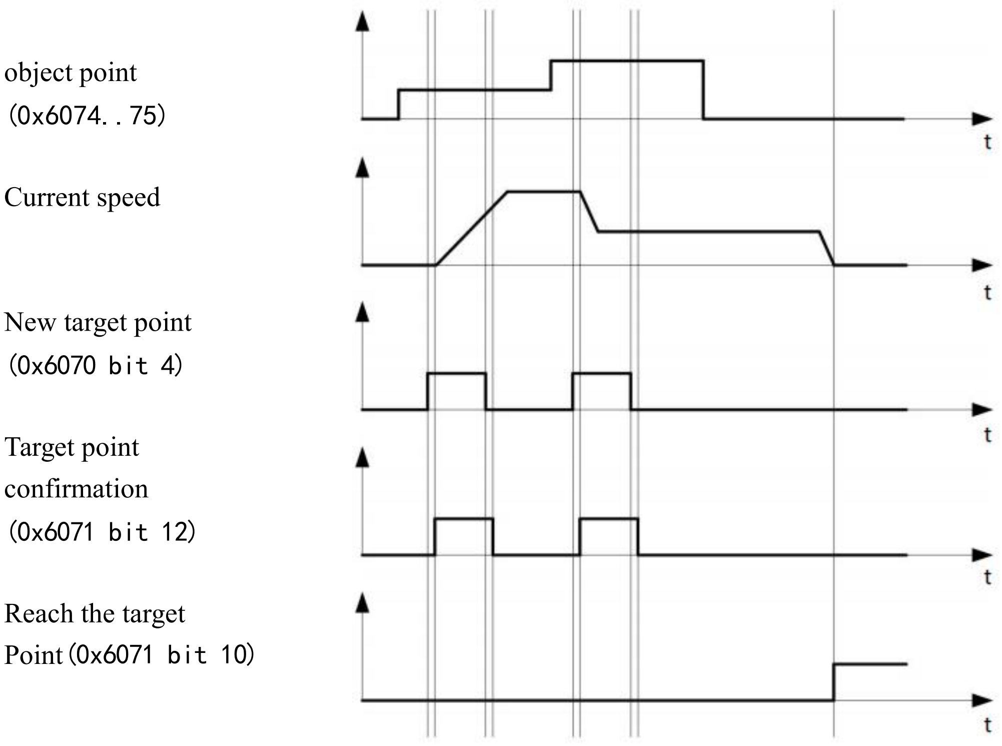
Target point confirmation (0x6071 bit 12)

Reach the target Point (0x6071 bit 10)



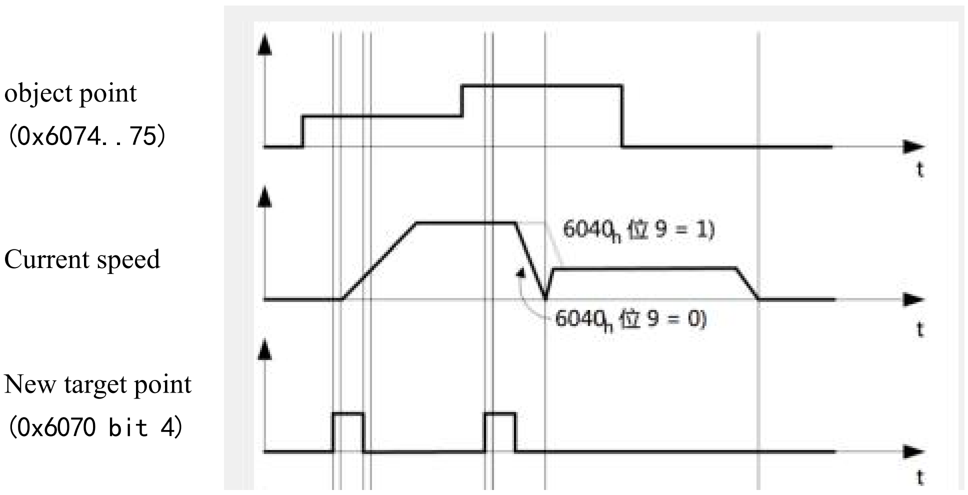
The conversion process of the second target position:

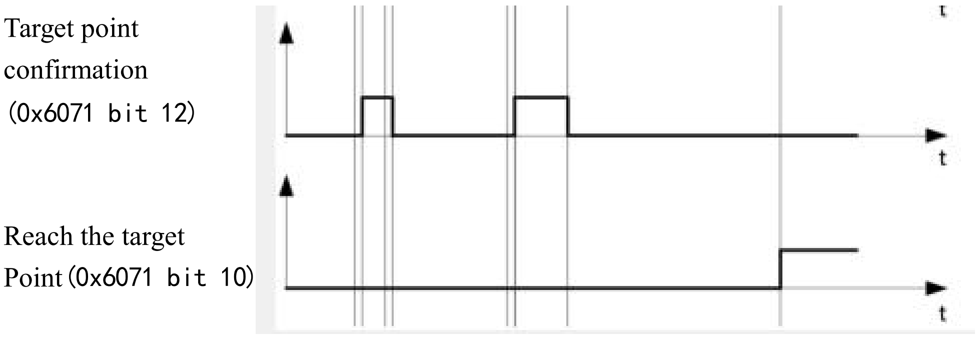
The following figure shows the conversion process of the second target position when moving to the first target position. In this figure, the bit 5 of the control word object (602e, 1) is set to "1" and the new target value will be accepted immediately.



The method of moving to the target position:

If the bit 9 in the control word object (602e,1) is a "0", it will first fully travel to the current target position. In this example, the final speed of the first target position is equal to zero. If bit 9 is set to "1", the final speed will be maintained until the target position is reached, and then the newly set motion parameters will take effect.




Target point confirmation (0x6071 bit 12)
Reach the target Point(0x6071 bit 10)
t
t
t

5.12 PV Mode

The working mode is set to 5 into (Profile Velocity Mode) PP mode, which adopts ladder acceleration and deceleration, and shares the starting speed, stop speed, acceleration, deceleration and running speed parameters with PP mode.

When the value of bit 8 (Halt) of the control word changes from "1" to "0", the motor will accelerate to the target speed with a preset starting speed in slope. When the value of the bit changes from "0" to "1", the motor slows down and stops moving. In the process of motion, a new running speed can be sent out, and the controller will smooth over to the newly set speed.

5.13 Analog positioning

PMC007C2 has an analog signal input port, and the internal 12-bit ADC, can be configured into analog positioning mode through software. First configure the analog positioning related parameters, and finally turn on the analog positioning enable. The following quart describes the analog related objects in detail.

5.13.1 Enable analog positioning

Enable analog positioning, 1 open, 0 closed

Object name	Enable analog positioning 1 open
Instruct address	0x6077
Object type	U8, rw
Range	ROM
Storage type	0、1
Default value	0

5.13.2 Analog initial AD code

Analog quantity start AD code, corresponding to the minimum value of the analog position

Object name	Analog quantity start AD code
Instruct address	0x6078
Object type	U16, rw

Range	ROM
Storage type	0-4096
Default value	0

5.13.3 Analog adjustment interval

Analog adjustment interval, Unit ms

The controller checks the analog input value at this time, and if the difference between the AD input value and the last input value is greater than the threshold value, the position will be adjusted once.

Object name	Analog adjustment interval
Instruct address	0x6079
Object type	U16, rw
Range	ROM
Storage type	0-65535
Default value	100

5.13.4 Analog regulating trigger value

The analog quantity adjusts the trigger value, and when the difference between the acquired AD code and the last acquired AD code is converted to a position greater than this value, the controller will adjust the position once.

Object name	Analog regulating trigger value
Instruct address	0x607a
Object type	U16, rw
Range	ROM
Storage type	0-65535
Default value	30

5.13.5 Minimum value of analog position

Subindex 0x05: Minimum value of analog position. Absolute position corresponding to the analog start AD code

Object name	Minimum value of analog position
Instruct address	0x607b..7c
Object type	S32, rw
Range	ROM
Storage type	-2^31~2^31
Default value	0

5.13.6 Maximum value of analog position

Analog position minimum. Absolute position corresponding to the analog start AD code

Object name	Maximum value of analog position
Instruct address	0x607b..7c
Object type	S32, rw
Range	ROM
Storage type	-2^31~2^31
Default value	0

5.14 Brake control

PMC007BXS supports brake control, and the output duty cycle can be adjusted by software, to avoid the serious problem of long-term brake heating.

Object name	Brake control
Instruct address	0x6038
Object type	U8, rw
Range	0~100
Storage type	RAM
Default value	0

5.15 Analogue input

PMC007BXS supports 0~24V voltage analog input, 12-bit ADC.

Object name	Analogue input
Instruct address	0x605f
Object type	U16, rw
Range	0~4095
Storage type	RAM
Default value	0

5.16 Power loss behavior

PMC007BXS can detect the power loss of the system and set the corresponding power loss behavior. The following is a detailed description of the power loss behavior settings related objects.

5.16.1 Power-down behavior control word

Object name	Power-down behavior control switch
Instruct	0x6082

address	
Object type	U16, rw
Range	ROM
Storage type	bit
Default value	0

Bit0: detects power loss and goes offline

Bit1: brake lock

The switch is turned on when the corresponding value is 1 and the switch is turned off for 0.

5.16.2 Power-down off-line voltage

Object name	Offline threshold voltage, unit mv
Instruct address	0x6083
Object type	U16, rw
Range	ROM
Storage type	0-65535
Default value	0

When the offline switch is turned on, the motor is offline when the power supply voltage is detected to be below this voltage.

5.16.3 Switching voltage

Object name	Offline threshold voltage, unit mv
Instruct address	0x6084
Object type	U16, rw
Range	ROM
Storage type	0-65535
Default value	0

When the power-down brake switch is opened, the brake brake is detected when the voltage of the power supply is lower than the voltage.

5.17 Reset back to zero

5.17.1 Reset command（Reset step distance）

Object name	Zero-back instruction (Back to zero step distance)
Instruct address	0x609a..9b
Object type	S32, rw
Storage type	RAM
Data type	-2147483647 ~ +2147483647
Default value	0

Reset command: After the reset command starts, the motor will move the specified distance in the forward direction until the operation is completed or the limit is triggered, so it is generally recommended that the reset distance should be set to be greater than or equal to the whole step. If it is at the limit and sends a reset command, the motor will reverse a short distance away from the limit and then re-trigger the limit again.

The bit8 of the controller status bit is the reset status bit, and the reset command will be automatically set to 1 when it is running, and it will be set to zero when the limit takes effect or the reset distance is completed.

5.17.2 Reset speed

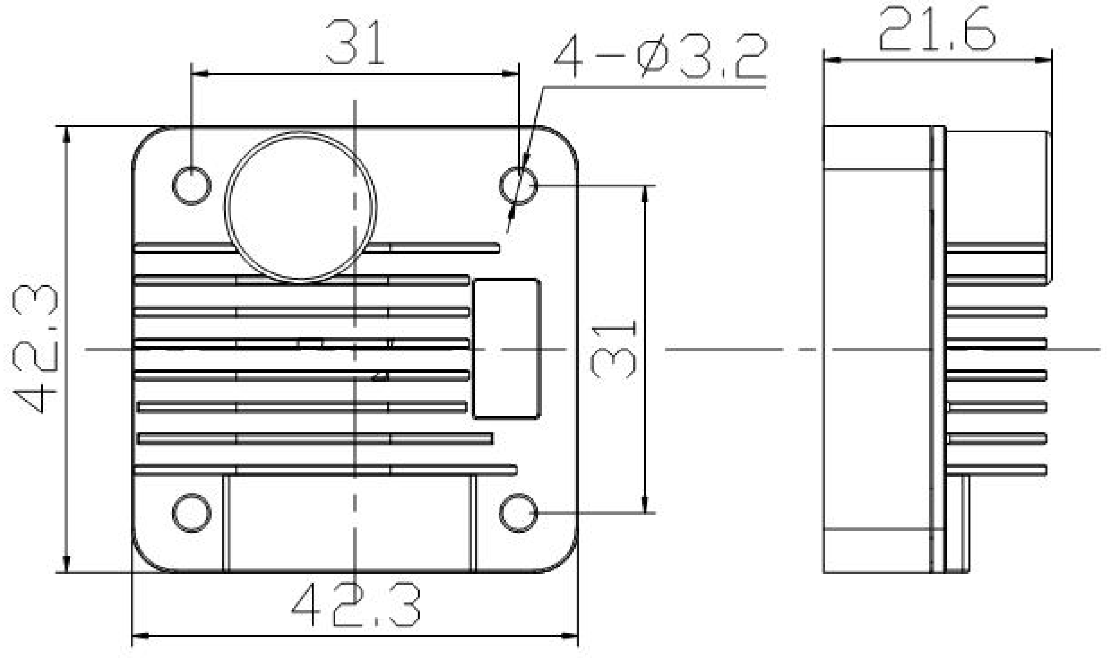
Object name	Reset speed
Instruct address	0x6098..99
Object type	S32, rw
Storage type	RAM
Data type	-2147483647 ~ +2147483647
Default value	32000

Set the reset speed is the running speed of the reset instructio n, the larger the value, the faster the running speed, there is a def ault value itself, which can be changed but cannot be saved by power-off. If there is a speed requirement for reset, it can be set in init ialization.

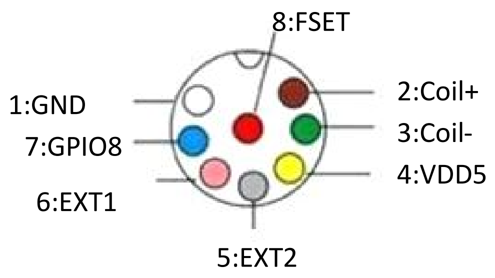
6 Electrical Characteristics

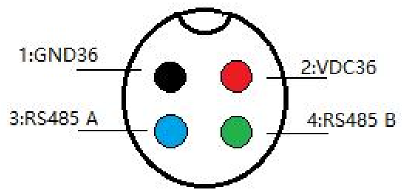
Parameter	Condition	Min	Typical	Max	Unit
Supply Power Voltage	Normal 25℃	12	24	47.5	V
Operation Temperature	12V DC	-20		55	℃
IO maximum current	source/sink current	0	10	20	mA
Output current	Normal 25℃	0.4	4	6	A
IO low Voltage	12V DC	-0.5		1.0	V
IO High Voltage	12V DC	3.0		5.5	V

7 Dimensions



For PMC007BXS IP64 Integrated motor with M12 connectors, the pinout is following:





8 Appendix 1 instruction table

modbus address	Name	Data Type	Attr.
2028	Slave address	UINT8	RW
202a..b	Baud rate	UINT32	RW
6000	Error state	UINT8	RW
6001	Controller status	UINT8	RW
6002	Rotation direction	UINT8	RW
6003..4	Max speed	INT32	RW
6005..6	Step command	INT32	RW
6007	Operation mode	UINT8	RW
6008	Start speed	UINT16	RW
6009	Stop speed	UINT16	RW
600a	Acceleration coefficient	UINT8	RW
600b	Deceleration coefficient	UINT8	RW
600c	Microstepping	UINT16	RW
600d	Max phase current	UINT16	RW
600e..f	Motor position	INT32	RW
6010	Current attenuation	UINT8	RW
6011	Motor enable	UINT8	RW
6013	External emergency stop enable	UINT8	RW
6014	Trigger mode of external emergency stop	UINT8	RW
6015	Type of sensor	UINT8	RW
6017	PVT operation control	UINT8	RW
6018	PVT mode control	UINT8	RW
6019	Max PVT point number	UINT16	RW
601a	PVT pointer	UINT16	RO
601b	Start index of PVT mode 1	UINT16	RW
601c	End index of PVT mode 1	UINT16	RW
601d	Start index of PVT model 2 acceleration stage	UINT16	RW
601e	End index of PVT model 2 acceleration stage	UINT16	RW

601f	Start index of PVT model 2 cycle stage	UINT16	RW
6020	End index of PVT model 2 cycle stage	UINT16	RW
6021	The times of PVT model 2 cycle stage	UINT16	RW
6022	Start index of PVT model 2 deceleration stage	UINT16	RW
6023	End index of PVT model 2 deceleration stage	UINT16	RW
6024	FIFO depth of PVT mode 3	UINT16	RW
6025	FIFO lower limit of PVT mode 3	UINT16	RW
6026	FIFO upper limit of PVT mode 3	UINT16	RW
6027..8	PVT position	INT32	RW
6029..a	PVT speed	INT32	RW
602b..c	PVT time	INT32	RW
602e	GPIO diriction	UINT16	RW
602f..30	GPIO configuration	UINT32	RW
6031	GPIO value	UINT16	RW
6038	Brake control	UINT8	RW
6042	Jitter delay of external emergency stop	UINT16	RW
6043	Locked-Rotor configuration	UINT8	RW
6044..5	Absolute position step	INT32	RW
6053	Termination step	UINT8	RW
6054	encoder CPR	UINT16	RW
6055..6	Position saving value power-down	INT32	RO
6057	Closed-loop parameter KP	UINT8	RW
6058	Closed-loop parameter KI	UINT8	RW
6059	Closed-loop parameter KD	UINT8	RW
605a	Closed-loop Pre-filter parameter	INT8	RW
605b	Closed-loop post-filter	INT16	RW

	parameter		
605c	Closed-loop stall length	INT16	RW
605d	Enable closed-loop torque loop	UINT8	RW
605e	Enable saving automatically when power is off	UINT8	RW
605f	Analogue input	UINT16	RW
6061	Step notification status	UINT8	RW
6062..3	Step notification position 1	INT32	RW
6064..5	Step notification position 2	INT32	RW
6067..8	PP/PV mode Accelerated speed	UINT32	RW
6069..a	PP/PV mode Dccelerated speed	UINT32	RW
606b..c	PP/PV mode Initial speed	UINT32	RW
606d..e	PP/PV mode Stop speed	UINT32	RW
6070	PP/PV mode control word	UINT16	RW
6071	PP/PV mode status word	UINT16	RW
6072..3	PP/PV mode running speed	INT32	RW
6074..5	PP/PV mode target location	INT32	RW
6077	Enable analog positioning	UINT8	RW
6078	Analog initial AD code	UINT16	RW
6079	Analog adjustment interval time	UINT16	RW
607a	Analog regulating trigger value	UINT16	RW
607b..c	Minimum value of analog position	INT32	RW
607d..e	Maximum value of analog position	INT32	RW
607f..80	real-time speed	INT32	RW
6082	Power-down behavior control word	UINT16	RW

6083	Power off motor enabling threshold	UINT16	RW
6084	Brake lock threshold	UINT16	RW
6089..608a	Encoder position	INT32	RW

9 Appendix 2 Communications Example

9.1 Modbus/RTU supported function codes

PMC007BxSxP currently supports the following function codes：

1、0x03：read holding register;

2、0x06：write single register;

3、0x10：write multiple registers;

9.2 Master Station communication parameter setting：

1）Baud rate：Same as slave station ;

2） Data bit：8;

3） Stop bit：1;

4） Parity bit：None;

9.3 Modbus master station message write operation

1） Function code 03:

E.g: below is an example of a read request and response of register 600c$_h$.

Send instruct: 01 03 60 0C 00 02 1A 08

Receive instruct: 01 03 04 00 00 04 0A 78 F4

2） Function code: 06:

E.g:Below is an example for write request and response in register 600c$_h$ with the value “0020$_h$”.

Send instruct: 01 06 60 0C 00 20 56 11

Receive instruct: 01 06 60 0C 00 20 56 11

3） Function code 10:

E.g:Below is an example for writing values “0000$_h$”, “0292$_h$”, “0000$_h$”, “0000$_h$” starting with register address 600c$_h$, number of registers is 4, length of the data is 8.

Send instruct: 01 10 60 0c 00 04 08 00 00 02 92 00 00 00 00 9f f5

Receive instruct: 01 10 60 0c 00 04 1f c9

9.4 Modbus/RTU control examples

9.4.1 Relative position control

Controller address	Function code	Register address	Data(0x)	Description	Remarks
01	06	600c	0020	Set microstep to 32	
01	06	600d	03e8	Set max phrase current to 1000	

01	06	6003	0000	Set the high 16 bits of speed to 0	
01	06	6004	7d00	Set the low 16 bits of speed to 32000(300RPM）	
01	06	6002	0000	Set the rotation direction to be positive	
01	06	6005	0000	Set the high 16 bits of relative position to 0	
01	06	6006	7d00	Set the low 16 bits of relative position to 32000	
01	03	6001	0000	Read controller statu,bit3 is busy status bit	The value of bit3: 1:busy 0:motor stop

9.4.2 Absolute position control

Controller address	Function code	Register address	Data(0x)	Description	Remarks
01	06	600c	0020	Set microstep to 32	
01	06	600d	03e8	Set max phrase current to 1000	
01	06	6003	0000	Set the high 16 bits of speed to 0	
01	06	6004	7d00	Set the low 16 bits of speed to 32000(300RPM）	
01	06	6044	0000	Set the rotation direction to be positive	
01	06	6045	7d00	Set the high 16 bits of relative position to 0	
01	03	6001	0000	Set the low 16 bits of relative position to 32000	The value of bit3: 1:busy 0:motor stop

9.4.3 Speed mode control

Controller	Function	Register	Data(0x)	Description	Remarks

address	code	address			
01	06	600c	0020	Set microstep to 32	
01	06	600d	03e8	Set max phrase current to 1000	
01	06	6007	0001	Set work mode to speed mode	
01	06	6003	0000	Set the high 16 bits of speed to 0	
01	06	6004	7d00	Set the low 16 bits of speed to 32000(300RPM）	
01	03	6001	0000	Read controller statu,bit3 is busy status bit	The value of bit3: 1:busy 0:motor stop

9.4.4 Commonly used fixed parameter settings

Controller address	Function code	Register address	Data(0x)	Description	Remarks
01	06	600c	0020	Set microstep to 32	
01	06	600d	03e8	Set max phrase current to 1000	
01	06	6008	0258	Set start speed to 600	
01	06	6009	0258	Set stop speed to600	
01	06	600a	0008	Set Acceleration coefficient to 8	
01	06	600b	0008	Set deceleration coefficient to 8	
01	06	6013	0003	Enable external stop	Both of EXT1/EXT2 enabled
01	06	6014	0000	Trigger mode of external emergency stop	Both of EXT1/EXT2 are drop edge .
01	06	200f	0002	Power-down saves all parameters	

10 Appendix 3 CRC code

Cyclic redundancy check CRC area is 2 bytes, containing a 16-bit binary data. the CRC value is calculated by the sending device, and the calculated value is attached to the information. when the receiving device receives the information, the CRC value is recalculated, and the calculated value is compared with the actual value received in the CRC area. if the two are not the same, an error is generated.

CRC start by setting all the 16 bits of the register as "1", and then putting the data of the adjacent 2 8-bit bytes into the current register, only the 8-bit data of each character is used as the starting bit CRC, the generation, and the stop bit and parity bit are not added to the CRC.

The result of every 8 bits of data is shifted one bit to the right (in the LSB direction) during the generation of CRC, and the MSB, detection is filled with "0". If the LSB is "1", it is different from the preset fixed value, and if the LSB is "0", it is not different or operated.

Repeat the above procedure until the shift is 8 times. After the 8th shift is completed, the next 8-bit data is different from the current value of the register, and after all the information is processed, the final value in the register is CRC value.

CRC generation process:

1. Set the 16-bit CRC register FFFFH.
2. A first 8-bit data performs an XOR operation with a CRC register 8 bits lower, putting the result into the CRC register.
3. Move one bit CRC register to the right, MSB fill zero, check LSB..
4. (if LSB is 1): CRC register performs an XOR operation with the A001H.
   (If LSB is 0): Repeat 3 and move one bit to the right.
5. Repeat 3 and 4 until 8 shifts are completed and 8 bytes are processed.
6. Repeat steps 2 to 5 to process the next 8-bit data until all bytes are processed.
7. the final value of the CRC register is the CRC value.
8. high 8 bits and low 8 bits should be placed separately when putting the CRC value into the information. Add CRC values to the information,send the 16 bit CRC value in the message, first send low 8 bit, then send high 8 bit.

11 Appendix 3 Modbus/RTU 16bits CRC check example

```
// CRC High byte value table
static const uint8_t s_CRCHi[] = {
        0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0,
        0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
        0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0,
        0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40,
        0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1,
        0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,
        0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1,
        0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
        0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0,
        0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40,
        0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1,
        0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40,
        0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0,
        0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40,
        0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0,
        0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40,
        0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0,
        0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
        0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0,
        0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
        0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0,
        0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40,
        0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1,
        0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
        0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0,
        0x80, 0x41, 0x00, 0xC1, 0x81, 0x40
} ;
// CRC Low byte value table
const uint8_t s_CRCLo[] = {
        0x00, 0xC0, 0xC1, 0x01, 0xC3, 0x03, 0x02, 0xC2, 0xC6, 0x06,
        0x07, 0xC7, 0x05, 0xC5, 0xC4, 0x04, 0xCC, 0x0C, 0x0D, 0xCD,
        0x0F, 0xCF, 0xCE, 0x0E, 0x0A, 0xCA, 0xCB, 0x0B, 0xC9, 0x09,
        0x08, 0xC8, 0xD8, 0x18, 0x19, 0xD9, 0x1B, 0xDB, 0xDA, 0x1A,
        0x1E, 0xDE, 0xDF, 0x1F, 0xDD, 0x1D, 0x1C, 0xDC, 0x14, 0xD4,
        0xD5, 0x15, 0xD7, 0x17, 0x16, 0xD6, 0xD2, 0x12, 0x13, 0xD3,
        0x11, 0xD1, 0xD0, 0x10, 0xF0, 0x30, 0x31, 0xF1, 0x33, 0xF3,
        0xF2, 0x32, 0x36, 0xF6, 0xF7, 0x37, 0xF5, 0x35, 0x34, 0xF4,
        0x3C, 0xFC, 0xFD, 0x3D, 0xFF, 0x3F, 0x3E, 0xFE, 0xFA, 0x3A,
        0x3B, 0xFB, 0x39, 0xF9, 0xF8, 0x38, 0x28, 0xE8, 0xE9, 0x29,
        0xEB, 0x2B, 0x2A, 0xEA, 0xEE, 0x2E, 0x2F, 0xEF, 0x2D, 0xED,
        0xEC, 0x2C, 0xE4, 0x24, 0x25, 0xE5, 0x27, 0xE7, 0xE6, 0x26,
        0x22, 0xE2, 0xE3, 0x23, 0xE1, 0x21, 0x20, 0xE0, 0xA0, 0x60,
        0x61, 0xA1, 0x63, 0xA3, 0xA2, 0x62, 0x66, 0xA6, 0xA7, 0x67,
        0xA5, 0x65, 0x64, 0xA4, 0x6C, 0xAC, 0xAD, 0x6D, 0xAF, 0x6F,
        0x6E, 0xAE, 0xAA, 0x6A, 0x6B, 0xAB, 0x69, 0xA9, 0xA8, 0x68,
        0x78, 0xB8, 0xB9, 0x79, 0xBB, 0x7B, 0x7A, 0xBA, 0xBE, 0x7E,
        0x7F, 0xBF, 0x7D, 0xBD, 0xBC, 0x7C, 0xB4, 0x74, 0x75, 0xB5,
        0x77, 0xB7, 0xB6, 0x76, 0x72, 0xB2, 0xB3, 0x73, 0xB1, 0x71,
        0x70, 0xB0, 0x50, 0x90, 0x91, 0x51, 0x93, 0x53, 0x52, 0x92,
        0x96, 0x56, 0x57, 0x97, 0x55, 0x95, 0x94, 0x54, 0x9C, 0x5C,
        0x5D, 0x9D, 0x5F, 0x9F, 0x9E, 0x5E, 0x5A, 0x9A, 0x9B, 0x5B,
        0x99, 0x59, 0x58, 0x98, 0x88, 0x48, 0x49, 0x89, 0x4B, 0x8B,
        0x8A, 0x4A, 0x4E, 0x8E, 0x8F, 0x4F, 0x8D, 0x4D, 0x4C, 0x8C,
        0x44, 0x84, 0x85, 0x45, 0x87, 0x47, 0x46, 0x86, 0x82, 0x42,
        0x43, 0x83, 0x41, 0x81, 0x80, 0x40
};
/*
*********************************************************************************************************
*       function name: CRC16_Modbus
*       description: CRC    _Modbus
*       Parameters: _pBuf : Data for validation
*                          _usLen :Data length
*    Return value: 16bit integer value。  For Modbus, this result, high bytes are transmitted first, low bytes are transmitted later..
******************************************************************************************************
*/
uint16_t CRC16_Modbus(uint8_t *_pBuf, uint16_t _usLen)
{
        uint8_t ucCRCHi = 0xFF; /* High CRC byte initial */
        uint8_t ucCRCLo = 0xFF; /* Low CRC byte initial */
        uint16_t usIndex;    /* CRCindex in the loop */

    while (_usLen--)
    {
    usIndex = ucCRCHi ^ *_pBuf++; /* calculate CRC */
    ucCRCHi = ucCRCLo ^ s_CRCHi[usIndex];
    ucCRCLo = s_CRCLo[usIndex];
    }
    return ((uint16_t)ucCRCHi << 8 | ucCRCLo);
}
```

Here's the calling method 1:

```
typedef struct
{
        uint8_t RxBuf[H_RX_BUF_SIZE];
        uint8_t RxCount;
        uint8_t RxStatus;
        uint8_t RxNewFlag;

        uint8_t RspCode;

        uint8_t TxBuf[H_TX_BUF_SIZE];
        uint8_t TxCount;

        uint16_t Reg01H;         /* Save the register header address sent by the host */
        uint16_t Reg02H;
        uint16_t Reg03H;
        uint16_t Reg04H;

        uint8_t RegNum;                 /* register number */

        uint8_t fAck01H;         /*    Response command flag 0 indicates execution failure 1 indicates execution success */
        uint8_t fAck02H;
        uint8_t fAck03H;
        uint8_t fAck04H;
        uint8_t fAck05H;
        uint8_t fAck06H;
        uint8_t fAck10H;

}MODH_T;
MODH_T g_tModH;
uint16_t crc;

g_tModH.TxBuf[0] = 0x31;
g_tModH.TxBuf[1] = 0x32;
g_tModH.TxCount = 2;
crc = CRC16_Modbus(g_tModH.TxBuf, g_tModH.TxCount);
```

Here's the calling method 2:

```
uint8_t _Data[10]; = { 0x31, 0x32};
uint8_t  usLen = 2;
crc = CRC16_Modbus(_Data, usLen);

```

12 Appendix 2 Error code table

MODBUS ERROR CODE		
Code	name	description
01	Illegal function	For the server (or slave), the function code received in the inquiry is an unallowed operation. This may be because the function code is only applicable to new devices and is not achievable in the selected unit. At the same time, it is also pointed out that the server (or slave) handles this kind of request in an error state, for example: because it is unconfigured, and it is required to return the register value.
02	Illegal data address	For the server (or slave), the data address received in the inquiry is an unallowable address. In particular, the combination of reference number and transmission length is invalid. For a controller with 100 registers, the request with offset 96 and length 4 will succeed, and the request with offset 96 and length 5 will generate exception code 02.
03	Illegal data value	For the server (or slave), the value included in the query is not allowed. This value indicates a failure in the remaining structure of the combined request, for example: the implied length is incorrect. It does not mean that because the MODBUS protocol does not know the significance of any special value of any special register, the data item submitted for storage in the register

		has a value that is not expected by the application.
04	Slave equipment failure	When the server (or slave) is trying to perform the requested operation, an unrecoverable error occurs.
05	Ensure	Used with programming commands. The server (or slave) has accepted the request and is processing the request, but it takes a long time to perform these operations. Returning this response prevents a timeout error in the client (or master). The client (or master station) can continue to send the polling procedure completion message to determine whether the processing is completed.
06	Slave equipment busy	Used with programming commands. The server (or slave) is processing long-duration program commands. When the Zhang server (or slave station) is idle, the user (or master station) should retransmit the message later.
08	Storage parity error	Used with function codes 20 and 21 and reference type 6 to indicate that the extended file area cannot pass the consistency check. The server (or slave) managed to read the log file, but found a parity error in the memory. The client (or master) can resend the request, but can request service on the server (or slave) device.
0A	Unavailable gateway path	Used with a gateway to indicate that the gateway cannot allocate an internal communication path from input port to output port for processing requests. Usually means that the gateway is misconfigured or overloaded.
0B	Gateway target device failed to respond	Used with the gateway to indicate that no response was obtained from the target device. Usually means that the device is not on the network.

Examples of client requests and server exception responses:

Request		Response	
domain name	Hex	domain name	Hex
Function code	01	Function code	81
Start address(Hi）	04	Error code	02
Start address(Lo）	A1		
Output number(Hi）	00		
Output number(Lo）	01		

In this example, the client addresses the server device's request. Function code (01) is used to read the output status. It will request the output status of address 1245 (hex 04A1). It is worth noting that, as explained in the output field (0001) number, only one output is available. If there is no output address in the server device, the server will return an exception response with an exception code (02). This shows the illegal data address of the slave.

Remarks: The abnormal response message has two fields different from the normal response:

Function code field: In a normal response, the server uses the response function code field to respond to the originally requested function code. The most significant bit (MSB) of all function codes is 0 (their values are all lower than 80 hex). In the abnormal response, the server sets the MSB of the function code to 1. This makes the function code value in the abnormal response higher than the function code value in the normal response by 80 hexadecimal.

Data field: In a normal response, the server can return the data or statistics table in the data field (any message requested in the request). In the exception response, the server returns the exception code in the data field. This defines the server state that caused the exception.