CANopen Controller Protocol

Catalogue

1：Introduction to CANopen Fundamentals

The full name of CAN is Controller Area Network, that is, controller area network. It was first proposed by Bosch Company in Germany and is one of the most widely used field buses in the world.

CANopen is an application layer protocol implemented on the CAN bottom layer protocol (logistics layer and link layer). As a field bus with advanced technology, high reliability, strong real-time, complete functions, reasonable cost and full openness, it is widely used in many industries.

This article focuses on the configuration of CANopen communication protocol and the commands that the controller receives from CANopen protocol. Help you use CANopen on the Santeng controller to configure the CAN communication parameters and ensure the efficient operation of CANopen mode.

This section contains the CANopen information specifications for the controller. For details about the CAN physical layer and CANopen protocol, please refer to the DS301 document.

1.1 Use and advantages of CANopen

CANopen enables multiple controllers to be interconnected on a scalable unified network. Its flexible configuration function provides a simple way to access the exposed parameters of the device, which can automatically (periodically or event driven) transmit data in real time.

The advantages of CANopen include:

1.EN50325-4 standard has been formed

2.Widely supported, independent of the manufacturer

3.Highly scalable

4.Flexible structure (widely used in various application fields)

5.Suitable for decentralized architecture

6.Extensive support for CANopen monitoring tools and solutions

1.2 General configuration

CAN mode: used to select one of the three working modes. Select Off to disable all CAN sending and receiving functions.

Node ID: the CAN node ID used by the controller when sending. Its value is between 1 and 126.

Bit Rate: optional bit rate. The available speed is 1000800500250125,50,25,10 kbit/s. The default is 250 kbit/s. The recommended speed of RawCAN and MiniCAN modes is 125 kbit/s

Heartbeat: The heartbeat frame cycle sent by the controller. This frame is compatible with CANopen 0x600+ID, with a data byte with a value of 0x05 (status: normal). Any mode selected will send a heartbeat frame. You can enter a value of 0 to disable.

1.3 CAN operating mode

1 - RawCAN

2 - MiniCAN

3 –CANopen

RawCAN is a low-level working mode, which can read and write CAN frames. It is recommended to be used in low data rate systems, which do not comply with any specific standards. Generally, MicoroBasic script language is used to construct and decode CAN frames.

MiniCAN is a greatly simplified subset of CANopen, which can integrate the controller into the existing CANopen network. This mode requires MicroBasic scripts to prepare and use CAN data.

CANopen is a complete standard from CAN in Automation (CIA), based on DS301 specification. This mode can be used when full compliance with CANopen standards is required.

1.4 Introduction to self startup

When automatic startup is enabled, the controller automatically enters CANopen working mode. Automatic startup of the controller is enabled by default. Disabling this parameter prevents the controller from starting automatically after reset. After disabling, the controller can only be enabled after receiving the CANopen management command.

1.5 ID Introduction

The CANopen network device must have a unique node ID between 1 and 126. A value of 0 is used to send broadcast messages. It cannot be used for network nodes.

1.6 Rate Introduction

The bit rate supported by the CAN bus ranges from 10 Kbps to 1 Mbps. The default bit rate used in the current CANopen is 250kbps. The effective bit rate supported by the controller is 1000K, 800K, 500K, 250K, 125K, 50K, 25K, 10K.

1.7 Heartbeat Introduction

Heartbeat messages are sent to the bus at several millisecond intervals. Heartbeat is very useful for detecting the existence of network nodes. The default value is set to 1000ms.

1.8 Frame Type

In CAN bus, there are four frame types: data frame, remote frame, error frame and overload frame.

(1) Data frame: data frame transmits application data

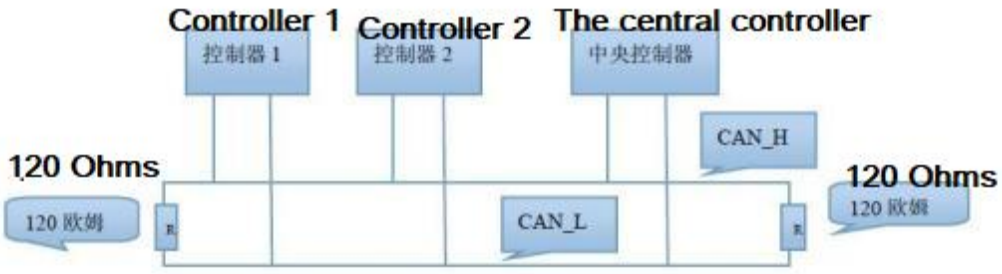
(2) Remote frame: by sending a remote frame, you can request data from the network and start other resource nodes to transmit their respective data.

(3) Error frame: The error frame can report the error of each node. It consists of two different domains. The first domain is the superposition of error marks provided by different stations, and the second domain is the error delimiter;

(4) Overload frame: if the node is not ready for receiving, it will transmit the overload frame, which is composed of two different domains. The first domain is the overload flag, and the second domain is the overload delimiter.

1.9 CAN bus connection

Connecting to the CAN bus is as simple as the figure above. A 120 Ohm resistor must be inserted at both ends of the bus cable. The CAN bus network can reach 1000 meters at most, and the length of the branch structure shall not exceed 0.3 m. See CAN bus specification for maximum length under various bit rates.



2.CANopen configuration of the controller

Through the upper computer software of MotorMonitor, the controller configuration communication parameters must be connected to the PC through the RS232 port, and CANopen mode can be enabled using the CAN menu in the configuration tab. In addition, the utility can be used to configure the following parameters:

- Node ID
- Baud rate
- Heartbeat (ms)
- Auto start

2.1 Exit default configuration

(1) Standard frame

(2) CAN mode：CANopen

(3) Baud rate： 250 kbps

(4) Node ID: 1 by default

(5) Heartbeat cycle：1000ms

(6) TPDO enable and transmission rate: 1500ms by default

(7) Auto start: default enable

2.2 CAN bus pin

Depending on the controller model, the CAN signal is located in the 15 pin hole connector or the 25 pin connector.

Table 1. Pin position of DB15 connector

Pin number	signal	explain
6	CAN L	CAN bus low
7	CAN H	CAN bus high

Table 2. Pin Position of DB25 Connector

Pin number	signal	explain
23	CAN H	CAN bus low
24	CAN L	CAN bus high

Note: See the port definition in the product manual for details

3. CANopen Controller Protocol

In fact, all real-time queries and real-time commands of the controller that can be accessed through serial port/USB communication can also be accessed through CANopen. All

commands supported by the controller have been sorted into the command table, as shown in the appendix.

Note: CAN and USB of most controllers cannot work at the same time. For the controller with USB connector, if simultaneous connection is not allowed, the controller will enable CAN when USB is not connected.

As long as USB is plugged into PC controller, USB will be enabled automatically and CAN will be prohibited. The CAN connection will remain disabled until the USB unplugged controller restarts.

3.1 CANopen Message Type

Controllers operating in CANopen mode can receive the following types of commands:

(1) Service data object, or SDO message, used to read and write parameter values

(2) Process data objects, or PDO mapped messages, used to automatically send parameters and/or receive commands at runtime

(3) Network management, or NMT defined according to CANopen specification

3.2 Service Data Object (SDO) read/write messages

You can use accelerated SDO messages to send runtime queries and runtime commands to the controller in real time. SDO messages provide a general method to access object dictionaries, and parameter values can be obtained in an irregular form, because generating each

SDO request and response message will consume too much network traffic.

3.3 Send Process Data Object (TPDO) messages

Sending PDO (TPDO) messages is one of the two types of PDO messages used at runtime. TPDO periodically sends runtime working parameters from the controller to other nodes. TPDO does not change object data; It only reads the internal parameters of the controller and sends them to the CAN bus.

3.4 Receive process data (RPDO) messages

RPDOs are configured to capture runtime data for the controller.

The CANopen implementation of the controller supports RPDOs. The data received by RPDOs is stored in 8 user variables and processed by MicroBasic scripts.

3.5.CANopen data sending and receiving

Using CANopen to send and receive data is not a simple way to send commands and parameters after specifying IDs. It needs to conform to certain loading specifications.

3.5.1Read Node Parameters

ID loading method: 0x600+target address ID (0x01-0x7e), range: 0x601-0x67E

The data frame loading method is as follows

Table 1. Read Node Commands

Data_0	Data_1、2	Data_3	Data_4	Data_5	Data_6	Data_7
0x40	Indexes	Subindex	00	00	00	00

CAN data frame format returned by the target node

Data_0	Data_1、2	Data_3	Data_4	Data_5	Data_6	Data_7
0x4F	Indexes	Subindex	XX	00	00	00
0x4B	Indexes	Subindex	XX	XX	00	00
0x47	Indexes	Subindex	XX	XX	XX	00
0x43	Indexes	Subindex	XX	XX	XX	XX

Note: 0x4F indicates that the returned message contains one byte of valid data, and by analogy, 0x43 indicates that the returned message contains four bytes of valid data

3.5.2 Write Node Parameters

ID loading method: 0x600+target address ID (0x01-0x7e), range: 0x601-0x67E

The data frame loading method is as follows:

Table 2. Write Node Commands

Data_0	Data_1、2	Data_3	Data_4	Data_5	Data_6	Data_7
0x2F	Indexes	Subindex	XX	00	00	00
0x2B	Indexes	Subindex	XX	XX	00	00

0x27	Indexes	Subindex	XX	XX	XX	00
0x23	Indexes	Subindex	XX	XX	XX	XX

Note: 0x2F indicates that the message sends one byte of valid data, and by analogy, 0x23 indicates that the message sends four bytes of valid data

CAN data frame format returned by the target node

Data_0	Data_1、2	Data_3	Data_4	Data_5	Data_6	Data_7
0x60	Indexes	Subindex	00	00	00	00

matters needing attention

- **"XX" represents a valid byte of data. "00" means invalid byte, and its data has no specification.**
- **The data fields sent and received by CANopen are all eight bytes, and the extra bytes are invalid.**
- **Multi byte indexes and data are all low byte first.**
- **If an invalid command is sent, an error frame will be returned. Relevant knowledge is beyond the scope of this agreement.**

3.5.3 Example of instructions

1. Speed control mode:

Note: The command index is 0x2000, the speed set value range is (- 1000 -- 1000), which is the maximum forward speed when 1000, the output is zero when 0, and the maximum reverse speed when - 1000.

If you want to control the speed of the controller with ID 1 and open loop mode, you need to send the following messages

ID： 0x601

Data:

Speed setting value is 500 (0x01F4)

0x23 0x00 0x20 0x01 0xf4 0x01 0x00 0x00

Speed setting value is 1000 (0x03E8)

0x23 0x00 0x20 0x01 0Xe8 0x03 0x00 0x00

The speed setting value is - 500 (0x0FE0C, that is, the complement of 0x01F4+1)

0x23 0x00 0x20 0x01 0x0C 0xFE 0x00 0x00

The speed setting value is - 1000 (0x0FC18, i.e. the complement of 0x03E8+1)
0x23 0x00 0x20 0x01 0x18 0xFC 0x00 0x00

The reply message is as follows

ID : 0x581

Data ：

0x60 0x00 0x20 0x01 0x00 0x00 0x00 0x00

2、 Position counting mode:

Set the target value of position mode as 4000000 (0x3D0900)

0x23 0x01 0x20 0x01 0x00 0x09 0x3D 0x00

Set the running speed of position mode 1000 (0x03E8)

0x23 0x02 0x20 0x01 0xE8 0x03 0x00 0x00

Read position count value

0x40 0x04 0x21 0x01 0x00 0x00 0x00 0x00

Read running speed

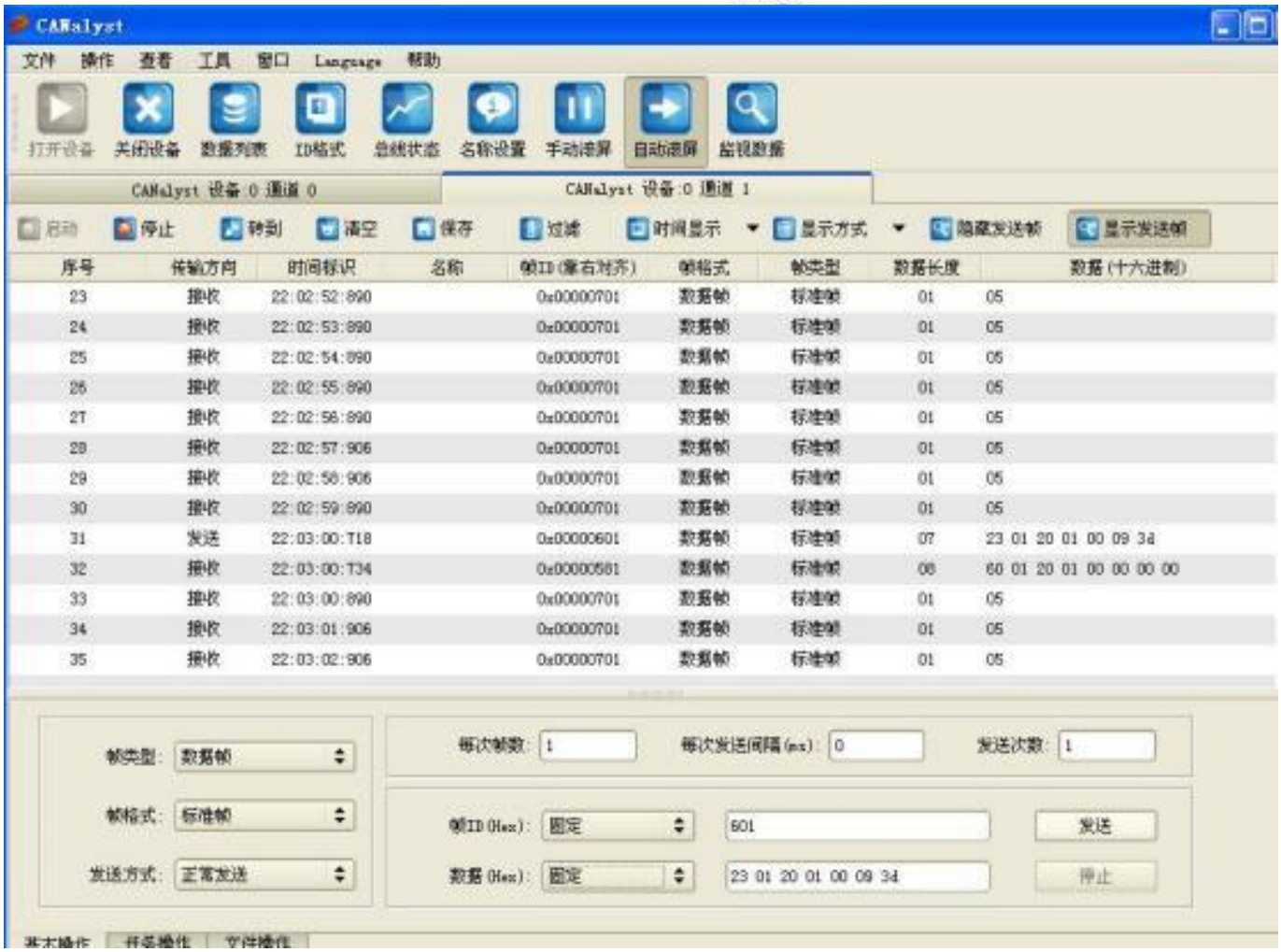
0x40 0x03 0x21 0x01 0x00 0x00 0x00 0x00

Stop command

0x2C 0x0E 0x20 0x01 0x00 0x00 0x00 0x00

During the test, it is recommended to use the CAN card to send data packets and monitor the bus status


Motor Monitor
Intelligent controller
选择串口:
COM6
当前版本:
常规双机版
启停操作:
启动
停止
重启
11/06/2019 11:33:18
语言版本:
Chinese
配置(C)
运行(R)
控制器+电机
控制器
电机1配置及输出
电机2配置及输出
其他
运行项
编码器1
编码器2
CAN模式
CANOpen
节点ID
1
波特率
M125
心跳周期(ms)
1000
CANOpen自启动
打开
数字输入
数字输出
脉冲
从控制器读取
保存到控制器
从文件读取
保存到文件


CANalyst
文件
操作
查看
工具
窗口
Language
帮助
打开设备
关闭设备
数据列表
ID格式
总线状态
名称设置
手动滚屏
自动滚屏
监视数据
CANalyst 设备:0 通道 0
CANalyst 设备:0 通道 1
启动
停止
转到
清空
保存
过滤
时间显示
显示方式
隐藏发送帧
显示发送帧
序号
传输方向
时间标识
名称
帧ID(靠右对齐)
帧格式
帧类型
数据长度
数据(十六进制)
23 接收 22:02:52:890 0x00000701 数据帧 标准帧 01 05
24 接收 22:02:53:890 0x00000701 数据帧 标准帧 01 05
25 接收 22:02:54:890 0x00000701 数据帧 标准帧 01 05
26 接收 22:02:55:890 0x00000701 数据帧 标准帧 01 05
27 接收 22:02:56:890 0x00000701 数据帧 标准帧 01 05
28 接收 22:02:57:906 0x00000701 数据帧 标准帧 01 05
29 接收 22:02:58:906 0x00000701 数据帧 标准帧 01 05
30 接收 22:02:59:890 0x00000701 数据帧 标准帧 01 05
31 发送 22:03:00:718 0x00000601 数据帧 标准帧 07 23 01 20 01 00 09 34
32 接收 22:03:00:734 0x00000581 数据帧 标准帧 08 60 01 20 01 00 00 00 00
33 接收 22:03:00:890 0x00000701 数据帧 标准帧 01 05
34 接收 22:03:01:906 0x00000701 数据帧 标准帧 01 05
35 接收 22:03:02:906 0x00000701 数据帧 标准帧 01 05
帧类型:
数据帧
帧格式:
标准帧
发送方式:
正常发送
每次帧数:
1
每次发送间隔(ms):
0
发送次数:
1
帧ID(Hex):
固定
601
发送
数据(Hex):
固定
23 01 20 01 00 09 34
停止

Appendix

The object dictionary given in the following table contains runtime queries and runtime

Commands that can be accessed using SDO/PDO messages during controller operation.
Table 3 Command Index Table (Runtime Commands)

Indexes	Subindex	Entry Name	Data Type& visit	RS232 command reference resources
0x2000	01	Set motor command, channel 1	S16 WO	“G”
	02	Set motor command, channel 2		
0x200C	00	Emergency shutdown	U8 WO	“EX”
0x200D	00	Emergency shutdown release	U8 WO	“MG”
0x200E	00	Shutdown in all modes	U8 WO	“MS”
0x2017	00	Flash Save configuration to Flash	U8 WO	“EES”

Table 4 Command Index Table (Position Mode Runtime Commands)

Indexes	Subindex	Entry Name	Data Type&Access	RS232 Command Reference
0x2001	01	Set position command, channel 1	S32 WO	“ P”
	02	Set position command, channel 2		
0x2002	01	Set position speed, channel 1	S16 WO	“ S”
	02	Set position speed, channel 2		
0x2003	01	Set encoder line number, channel 1	S32 WO	“ C”
	02	Set encoder line number, channel 2		
0x2006	01	Set acceleration, channel 1	S32 WO	“ MAC”
	02	Set acceleration, channel 2		
0x2007	01	Set deceleration, channel 1	S32 WO	“ MDEC”
	02	Set deceleration, channel 2		

Table 5 Instruction index table (runtime query)

Indexes	Subindex	Entry Name	Data Type&Access	Command Reference
0x2100	01		S16 RO	“ A”
		Read motor current, channel 1		
	02	Read motor current, channel 2	S16 RO	
0x2101	01	Read current motor command, channel 1	S16 RO	“ M”
	02	Read current motor command, channel 2		
0x2103	01	Read encoder motor speed, channel 1	S16 RO	“ S”
	02	Read encoder motor speed, channel 2	S16 RO	

0x2104	01	Read absolute encoder count, channel 1	S32 RO	“ C”
	02	Read absolute encoder count, channel 2	S32 RO	
0x2105	01		S32 RO	“ CB”
		Read Absolute Brushless Count, Channel 1		
	02	Read Absolute Brushless Count, Channel 2	S32 RO	
0x2107	01	Read encoder motor speed at 1/1000 of the maximum value, channel 1	S16 RO	“ SR”
	02	Read encoder motor speed at 1/1000 of the maximum value, channel 2		

0x2108	01	Read encoder count relative value, channel 1	S32 RO	“ CR”
	02	Read encoder count relative value, channel 2		
0x2109	01	Read relative value of brushless count, channel 1	S32 RO	“ CBR”
	02	Read relative value of brushless count, channel 2		
0x210A	01	Press RPM to read brushless motor speed, channel 1	S16 RO	“ BS”
	02	Press RPM to read brushless motor speed, channel 2		
0x210C	01	Read battery current, channel 1	S16 RO	“ BA”

	02	Read battery current, channel 2		
0x210D	01	Read internal 13.5V pushing voltage (v)	U16 RO	“ V”
	02	(v) Read battery voltage （v）	U16 RO	
	03	Read internal 5V voltage (mV)	U16 RO	
0x210E	00	Read all digital inputs	U32 RO	“ D”
0x210F	01	Read enclosure&internal temperature (MCU temperature)	S8 RO	“ T”
	02	Read Housing&Internal Temperature (Channel 1)		
	03	Read Housing&Internal Temperature (Channel 2)		
0x2111	00	Read status flag	U8 RO	“ FS”
0x2112	00	Read fault flag	U8 RO	“ FF”

0x2113	00	Read the current digital output	U8 RO	“ DO”
0x2119	00	Reading time	U32 RO	“ TM”
0x2121	00	Read the motor status flag	U8 RO	“ FM”

Detailed description of common agreements:

- **EX Emergency stop CANOpen id: 0x200C**

Description:

The EX command will cause the controller to enter the emergency stop state, just like the hardware emergency stop is detected on the input pin. In case of emergency, the controller shall be reset or the MG shall receive the release command.

Syntax Serial Number：!EX

- **MG Emergency stop release CANOpen ID: 0x200D**

Description：

The MG command will release the emergency stop condition and allow the controller to resume normal operation. Before sending this command, make sure that the fault status has been cleared.

Grammatical sequence：!MG

- **MS Stop CANOpen id in all modes: 0x200E**

Description：

MS command is similar to EX emergency stop command, it is applied to the specified motor channel syntax serial number:! MS [cc] cc=Motor channel

- **ESS - Save configuration in EEPROM CANOpen id: 0x2017**

Description：

This command saves any changes to the controller configuration to Flash. After the controller is powered on next time, the saved configuration will be loaded again. This command is a duplicate of the EESAV maintenance command. It is provided as a real-time command so that configuration changes can be saved from scripts.

Grammatical sequence：!EES

- **P - Turn to the set position CANOpen id: 0x2001**

Description：

Use this command in the position counting mode to move the motor to the specified encoder position count value.

Grammatical sequence：!P [cc] nn

cc=motor channel nn=absolute count value (encoder resolution * 4=1 turn. Setting range ± 2147483648)

Example：

! P 1 10000: Make the motor reach the absolute count value 10000

- **S - Set motor speed CANOpen id: 0x2002**

Description：

In closed loop speed mode, this command will cause the motor to rotate RPM at the desired speed. In closed loop position mode, this command determines the speed at which the motor will move from one position to another.

Syntax sequence:! S [cc] nn cc=motor channel nn=speed value in RPM

Example：

！ S 2500: Set the position speed of motor 1 to 2500 RPM. (Minimum: - 500000 Maximum: 500000)

- **C - Set encoder count value CANOpen id: 0x2003**

Description：

This command is used to set the encoder count value of the selected motor channel. Note that changing the value of the controller during operation will have adverse effects in closed loop mode.

Syntax sequence:! C [cc] nn

cc=motor channel

nn=counter value

Example：

！ C 2 - 1000: Load encoder counter to - 1000 in channel 2

！ C 1 0: Clear encoder counter 1

- **AC Set acceleration CANOpen id: 0x2006**

Description：

Set the speed change rate when the motor channel accelerates. This command is the same as the MACC configuration command, but the difference is that it can be changed quickly during motor operation. The acceleration value is 0.1 * RPM per second. When in use, the controller is equipped with an encoder, and the speed and acceleration values are the actual speed. The brushless motor controller uses Hall sensors to measure the actual speed and acceleration, also in real RPM/s. When using a controller without a speed sensor, the acceleration value is relative to the Max RPM configuration parameter, which itself is the value that the user provided speed is usually expected to be at full power. Assume that the Max RPM parameter is set to 1000, and the acceleration value of 10000 indicates that the motor will reach full speed within 1 second.

Syntax sequence:! AC cc nn

cc=motor channel

nn=acceleration value of 0.1 * RPM/s (min: 0; max: 500000)

Example：

! AC 1 2000: If the speed measurement channel is 1, increase the speed of motor 1 by 200 RPM per second.

! AC 20000: If there is no speed sensor and RPM is set to 1000, the time from 0 to full power is 0.5s.

- **DC Set deceleration time CANOpen id: 0x2007**

Description：

Set the speed change rate during motor channel deceleration. This command is the same as the MDEC configuration command, but the difference is that it can be changed quickly during motor operation. The acceleration value is 0.1 * RPM per second. When using a controller equipped with an encoder, the speed and deceleration values are the actual RPM. The brushless motor controller uses Hall sensor to measure the actual speed and deceleration, which will also be measured in actual RPM/s. When using a controller without a speed sensor, the acceleration value is relative to the maximum RPM configuration parameter, which is the value of the user provided speed that is usually expected to be at full power. Assuming that the maximum RPM parameter is set to 1000, the deceleration value of 10000 means that the motor will change from full speed to 0 within 1 second regardless of the actual motor speed.

Syntax sequence:! DC cc nn

cc=motor channel

nn=deceleration value of 0.1 * RPM/s

Example：

! DC 1 2000: If the speed is measured, reduce the speed of motor 1 by 200 RPM every second.

! DC 2 20000: If there is no speed sensor and the maximum RPM is set to 1000, the time from full power to stop is 0.5s.

- **A-reading motor output current CANOpen id: 0x2100**

Description：

Measure and report the motor current of all operating channels. Please note that the current flowing through the motor is often higher than the current flowing through the battery.

Syntax sequence:? A [cc] cc=Motor channel

Reply：

A=aaType: Minimum: 0

aa=Current per channel * 10

Example：

Q:? A

Reply: A=100:200

Q: ?A 2

Reply: A=200

Note: A single channel controller will report a value

- **Y-reading voltage CANOpen ID: 0x210D**

Description：

Report the voltage measured inside the controller at three positions: the main battery voltage, the internal voltage of the motor drive stage, and the available voltage DB 15 or 25, the 5V output

on the front connector. For safe operation, the drive stage voltage must be higher than 12V. The 5V output shows that the internal regulation of the controller is 5V, minus the voltage drop of the diode used for protection, and will be within the range of 4.7V. Monitor the battery voltage to detect undervoltage or overvoltage conditions.

Syntax sequence:? V [ee] Reply: V=nn

ee =

1：Internal voltage

2：Battery voltage

3：5V output

For internal voltage and battery voltage, nn=V * 10. The 5V output is mv.

Example：

Q: ?V

R:V=135:246:4730

Q: ?V 3

R:V=4730

- **C - Read the absolute value of encoder counter CANOpen id: 0x2104**

Description：

Returns the encoder value as an absolute number. The counter is a 32 bit range of ± 2147483648 counts.

Syntax sequence:? C [cc]

Reply：

C=nn min: - 2147483648; Maximum:+2147483648

Definition：

cc=encoder channel number

nn=absolute counter value (nn/4 times frequency/encoder resolution=motor running cycles)

- **T - Reading temperature CANOpen id: 0x6403**

Description：

Report the temperature chip of each heat sink edge and

internal MCU silicon chip. The reported value resolution is

Celsius.

Syntax sequence:? T [cc]

Reply：

T=cc min: - 40; Maximum: 125

cc =

1：MCU temperature

2：Channel 1 side

3：Channel 2 side

tt=in ° C

Note：

On some controller models, additional temperature values may be reported. These were measured at different points and were not recorded. You can safely ignore these additional data. Other controller models have only one radiator temperature sensor, so only one value other than the internal IC temperature can be reported.

- **FS Read status flag CANOpen id: 0x6405**

Description：

Reports the status of the controller using status flags to indicate the conditions under which some internal states operate normally. The response to this query is all individual numeric status flags. The status of each flag is read by converting this number to a binary number.

Syntax sequence:? FS

Reply：

FS = f1 + f2 * 2 + f3 * 4 + ... + fn * 2 ^ n-1；

Minimum: 0 Maximum: 255

Definition：

f1 =Serial port mode

f2 =Pulse mode

f3 =Analog mode

f4 =Power off

f5 =Stall detected

f6 =At Limit

f7 =Not used

f8 =Run Script

- **FF Read fault flag CANOpen id: 0x6406**

Description：

Reports the status of controller failures that may occur during operation. The response to this query is a single number that must be converted to a binary number to evaluate each status bit.

Syntax sequence:? FF

Reply：

FF = f1 + f2 * 2 + f3 * 4 + ... + fn * 2n-1

Minimum: 0; Maximum: 255

Definition :
f1 = overheated

f2 = Overpressure

f3 = Undervoltage

f4 = short circuit

f5 = Emergency stop

f6 = Brushless sensor fault

f7 = MOSFET fault

f8 = Default configuration loaded at startup

example：

Q：?FF

R：FF = 2：Overvoltage fault

- **FM Read running status flag CANOpen id: 0x2122**

Description：

Report the operation status of each motor. The response to this query is that a single number must be converted to a binary number to evaluate each status bit.

Syntax sequence:? FM [cc]

Reply：

FM = f1 + f2 * 2 + f3 * 4 + ... + fn * 2n-1

Minimum: 0; Maximum: 255

Definition：

cc =Motor channel

f1 =Overcurrent limiting

f2 =Motor stops

f3 =Loop error detected

f4 =Safe Stop Activation

f5 =Trigger positive limit

f6 =Trigger reverse limit

f7 =Overcurrent protection stop

example：

Q：?FM 1

R: FM=6: Motor 1 stops and loop error is detected

Noted：

When the motor command returns to 0, f2, f3 and f4 are cleared. When f5 or f6 is open, the motor can only be commanded to run in reverse.