Control command

The 4 motor interfaces on the module correspond to motors on robot car, as shown below

M1 -> Upper left motor (left front wheel of the car)
M2 -> Lower left motor (left rear wheel of the car)
M3 -> Upper right motor (right front wheel of the car)
M4 -> Lower right motor (right rear wheel of the car)

Serial port configuration

Baud rate 115200, no parity, no hardware flow control, 1 stop bit

1.Configure motor type

Command	Explanation	Example	Remark	Firmware Defaults	Power off save
$mtype:x#	Motor model	$mtype:1#	The motor model is 520 motor	520 motor	Y
Note					

0. Motor type selection. If the motor encoder A is connected to the A port of the board, then B is connected to B. You need to select the 310 motor model. Otherwise, you need to select the 520 motor or TT motor.
1. The command can be sent in all uppercase or lowercase letters.

2. If the above command is successful, it will return the message **command+OK**. If no message is returned, check the serial port connection.
3. x: is the type of motor.
   The motor types represented by different values are as follows:
   1: 520 motor
   2: 310 motor
   3: TT motor (with encoder)
   4: TT motor (without encoder)

Note: If you are using a motor without an encoder, you can select type 4, that is, the command is: $mtype:4#
If you are using a motor with an encoder, you can select one of 1, 2, and 3.

2.Configuring motor deadband

Command	Explanation	Example	Remark	Firmware Defaults	Power off save
$deadzone:xxxx#	Configure the motor pwm pulse dead zone	$deadzone:1650#	When controlling PWM, the dead zone value will be added by default so that the motor will not have an oscillation area.	1600	Y
Note					

1. The command can be sent in all uppercase or all lowercase
2. If the above command is successful, it will return the message of **command+OK**. If no message is returned, check the serial port connection
3. xxxx: is the value of the dead zone, which needs to be measured. By changing this value, the minimum vibration of the motor can be eliminated
4. The range of the dead zone value (0-3600)

3.Configuring motor phase lines

Command	Explanation	Example	Remark	Firmware Defaults	Power off save
$mline:xx#	Configure the motor phase lines	$mline:13#	Configure the phase of the motor Hall encoder to 13 lines	11	Y
Note					

Note					

1. The command can be sent in all uppercase or all lowercase
2. The above command will return **command+OK** information if it is successful. If no information is returned, check the serial port connection
3. xx: This is the phase of the Hall encoder for one turn. This value needs to be obtained by checking the merchant's motor parameter table
4. For motors with encoders: This value plays a **main role** in controlling speed. This value needs to be correct
5. Motors without encoders: This value configuration can be ignored

4.Configure motor reduction ratio

Command	Explanation	Example	Remark	Firmware Defaults	Power off save
$mphase:xx#	Configure the motor reduction ratio	$mphase:40#	Configure the motor reduction ratio to 40	30	Y
Note					

1. The command can be sent in all uppercase or all lowercase
2. The above command will return **command+OK** information if it is successful. If no information is returned, check the serial port connection
3. xx: This is the motor reduction ratio parameter. This value needs to be obtained by checking the merchant's motor parameter table
4. For motors with encoders: This value plays a **main role** in controlling speed. This value needs to be correct
5. Motors without encoders: This value configuration can be ignored

5.Configure the wheel diameter (Optional)

Command	Explanation	Example	Remark	Firmware Defaults	Power off save
$wdiameter:xx#	Configure the wheel diameter	$wdiameter:50#	The diameter of the wheel is 50mm	67 mm	Y
Note					

1. The command can be sent in all uppercase or all lowercase
2. The above command will return **command+OK** information if it is successful. If no information is returned, check the serial port connection
3. xx: This is the diameter of the wheel. This value can be measured or obtained using the merchant's information

4. For motors with encoders: This value plays a **main role** in controlling speed. This value needs to be correct in millimeters (mm); if this value is incorrect, it will only affect the speed data inaccurately, and will not affect the encoder data
5. Motors without encoders: This value configuration can be ignored

6.Configure PID parameters for motor control

Command	Explanation	Example	Remark	Firmware Defaults	Power off save
$MPID:x.xx,x.xx,x.xx#	Configure PID parameters	$MPID:1.5,0.03,0.1#	The configuration control speed is P: 1.5, I: 0.03, D: 0.1	P:0.8 I:0.06 D:0.5	Y
Note					

1. The command can be sent in all uppercase or all lowercase
2. The above command will return **command+OK** if it is successful. If no information is returned, check the serial port connection
3. x.xx, x.xx, x.xx: These are the parameters for controlling motor p, i, d respectively. **Every time the value is changed, the chip will restart and stop the moving motor. This is a normal situation**
4. For motors with encoders: the pid parameter is valid, and this value needs to be correct. **Generally, there is no need to modify the pid, and the default value can be used**
5. Motors without encoders: the pid parameter is invalid, and this value configuration can be ignored

7.Reset all variables to defaults value

Command	Explanation	Example	Remark	Firmware Defaults	Power off save
$flash_reset#	Restore factory defaults value	-	-	-	-
Note					

1. The command can be sent in all uppercase or all lowercase
2. If the above command is successful, it will return the message **command+OK**. If no message is returned, check the serial port connection
3. Execute this command and the module will restart once

8.Control speed command

Command	Explanation	Example	Remark	Firmware Defaults	Power off save
$spd:0,0,0,0#	Control the speed of 4 motors	$spd:100,-100,0,50#	Control the speed of 4 motors M1:100 M2:-100 M3:0 M4:50	-	N
Note					

1. The command can be sent in all uppercase or all lowercase letters
2. If the above command is successful, the motor will move, and the serial port will not return anything
3. This command is only valid for encoder type motors.
4. The speed range is (-1000~1000) and it is invalid if it exceeds the range
5. 0,0,0,0: represents M1, M2, M3, M4 on the board screen
6. When the spd command parameter is 0, PID still works. At this time, the hand cannot turn the wheel. To turn the wheel, use the pwm command to set it to 0.

9.Direct control PWM instruction

Command	Explanation	Example	Remark	Firmware Defaults	Power off save
$pwm:0,0,0,0#	Control 4 motors with PWM output	$spd:0,-520,300,800#	Control the PWM output of 4 motors M1:0 M2:-520 M3:300 M4:800	-	N
Note					

1. The command can be sent in all uppercase or all lowercase
2. If the above command is successful, the motor will move, and nothing will be returned from the serial port
3. The speed range is (-3600~3600) and it is invalid if it exceeds the range
4. For motor control without encoder, **you can control it through this command**
5. 0,0,0,0: represents M1,M2,M3,M4 on the board screen

10.Report encoder data (This command is only valid for motors with encoder)

Command	Explanation	Example	Remark	Firmware Defaults	Power off save
$upload:0,0,0#	Receive encoder data	$upload:1,0,0#	Receive the total encoder data of wheel rotation 1: open, 0: do not receive	-	N
Note					

1. The command can be sent in all uppercase or all lowercase
2. $upload:0,0,0#: The first 0 represents: the total encoder data of the wheel rotation The second 0 represents: the real-time encoder data of the wheel rotation (10ms) The third 0 represents: the speed of the wheel
3. The corresponding information can be received at the same time

- The total encoder data of the wheel rotation returns the information: "$MAll:M1,M2,M3,M4#"
- The real-time encoder data of the wheel rotation returns the information: "$MTEP:M1,M2,M3,M4#"
- The speed of the wheel returns the information: "$MSPD:M1,M2,M3,M4#"

11.Query flash variables

Command	Explanation	Example	Remark	Firmware Defaults	Power off save
$read_flash#	Query flash variables	-	-	-	N
Note					

1. The command can be sent in all uppercase or all lowercase
2. If the above command is successful, it will return the message **command+OK**. If no message is returned, check the serial port connection

12.Check battery level

Command	Explanation	Example	Remark	Firmware Defaults	Power off save
$read_vol#	Check battery level	-	-	-	N

Command	Explanation	Example	Remark	Firmware Defaults	Power off save
Note	level				

1. The command can be sent in all uppercase or all lowercase
2. If the above command is successful, it will return the information of **battery power ($Battery:7.40V#)**. If no information is returned, please check the serial port connection

IIC protocol control

4-channel motor driver board IIC device address: 0x26

Register address	R/W	Type	Range	Explanation	Example

Register address	R/W	Type	Range	Explanation	Example
0x01	w	uint8_t	1: 520 motor 2: 310 motor 3: TT motor (with encoder) 4: TT motor (without encoder)	Write motor type	Device address + register address + motor type
0x02	w	uint16_t	0-3600	Configuring motor deadband	Device address + register address + motor dead zone value
0x03	w	uint16_t	0-65535	Configure the number of motor magnetic ring lines	Device address + register address + number of motor magnetic ring lines
0x04	w	uint16_t	0-65535	Configure motor reduction ratio	Device address + register address + motor reduction ratio
0x05	w	float	-	Enter the wheel diameter, unit: mm	Device address + register address + wheel diameter (need to convert float to bytes - little endian first)

Register address	R/W	Type	Range	Explanation	Example
0x06	w	int16_t	-1000~1000	Speed control, this register is only effective for motors with encoders, unit: mm	Device address + register address + speed (transmitting data of 4 motors each time) **Big-endian mode**Each speed occupies 2 bits For uint8_t eg: m1 motor speed 200, m2 motor speed -200, m3 motor speed 0, m4 motor speed 500 That is: [0x00 0xC8 0xFF 0x38 0x00 0x00 0x01 0xf4]
0x07	w	int16_t	-3600~3600	PWM control, this control does not require encoder data and can directly control the motor rotation	Device address + register address + speed (transmitting data of 4 motors each time) **Big-endian mode**Each speed occupies 2 bits For uint8_t eg: m1 motor pwm200, m2 motor pwm is -200, m3 motor pwm is 0, m4 motor pwm is 500 That is: [0x00 0xC8 0xFF 0x38 0x00 0x00 0x01 0xf4]
0x08	R	uint16_t	-	Reading battery level	A correct data: data = (buf[0] <<8|buf[1])/10.0
0x10	R	int16_t	-	Read M1 encoder real-time pulse data - 10ms	A correct data: data = buf[0] <<8|buf[1]

Register address	R/W	Type	Range	Explanation	Example
0x11	R	int16_t	-	Read M2 encoder real-time pulse data - 10ms	A correct data: data = buf[0] <<8|buf[1]
0x12	R	int16_t	-	Read M3 encoder real-time pulse data - 10ms	A correct data: data = buf[0] <<8|buf[1]
0x13	R	int16_t	-	Read M4 encoder real-time pulse data - 10ms	A correct data: data = buf[0] <<8|buf[1]
0x20	R	int16_t	-	Read the total pulse data of the M1 motor encoder (High bit)	-
0x21	R	int16_t	-	Read the total pulse data of the M1 motor encoder (Low bit)	The acquired data needs to be shifted to get the correct data. High bit represents: buf[0] buf[1] Low bit represents: bf[0] bf[1] (data = buf[0] <<24|buf[1] <<16|bf[0] <<8|bf[1])
0x22	R	int16_t	-	Read the total pulse data of the M2 motor encoder (High bit)	-
0x23	R	int16_t	-	Read the total pulse data of the M2 motor encoder (Low bit)	Same calculation method as M1
0x24	R	int16_t	-	Read the total pulse data of the M3 motor encoder (High bit)	-
0x25	R	int16_t	-	Read the total pulse data of the M3 motor encoder (Low bit)	Same calculation method as M1

Register address	R/W	Type	Range	Explanation	Example
0x26	R	int16_t	-	Read the total pulse data of the M4 motor encoder (High bit)	-
0x27	R	int16_t	-	Read the total pulse data of the M4 motor encoder (Low bit)	Same calculation method as M1