

优必选舵机 60kg/25kg 舵机通讯协议整理

目录

优必选舵机 60kg/25kg 舵机通讯协议整理.....	1
总述.....	2
舵机角度及速度控制（运动控制）.....	3
舵机零点修改.....	3
舵机锁定、解除.....	3
舵机状态查询.....	3
舵机 ID 修改.....	4
PS:	4
硬件准备:	5
24V 电源、导线、接线器若干.....	5
一代优必选舵机 60kg/25kg 舵机，需要是从第一代克鲁泽机械臂拆卸下来的舵机：（二代的发热量高，不推荐）.....	5
带线端子，端子采用 4pin 5264 的带线端子，不知道到哪里购买的可参考淘宝链接: .5	
● 舵机引脚定义：（附图.....	6
硬件接线如图.....	7
上位机调试及参数设定.....	8
本文所使用的 USB 转 CAN 模块的上位机具有配置模式、数据模式，接下来以这个模块为例进行 CAN 总线参数的设置.....	8
选择正确串口，打开串口，读取模块目前参数，按下图进行参数设置.....	9
进入数据模式发送命令.....	10
舵机通信协议解析及示例.....	10
指令包.....	10
应答包.....	11
指令类型.....	11
示例 1：运动控制.....	11
示例 2：查询:	12
示例 3：舵机 ID 修改:	13
示例 4：舵机锁定与解锁(舵机 ID 为 3):	14

部分资料搜集自互联网

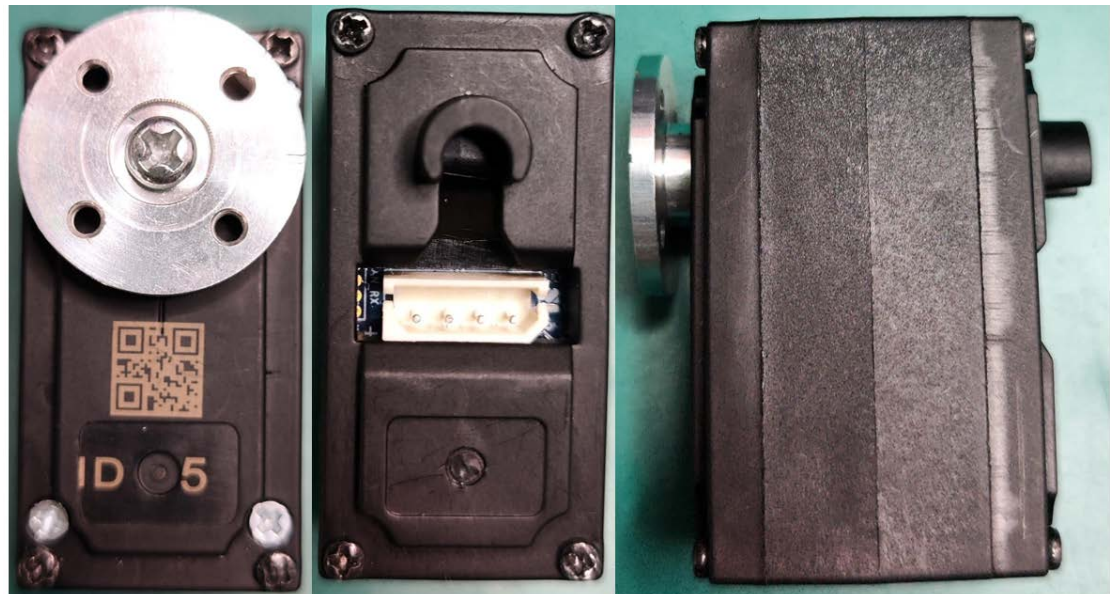
本文仅供参考

禁止用于商业用途，违者后果自负

2023. 4. 30



60kg 舵机



25kg 舵机

总述

舵机采用 ARM 32 位单片机为主控制核芯，位置感应采用 360 度 12 位精度的磁铁感应角度方案，通信采用 CAN 协议

舵机目前已经探索出了以下功能：

舵机角度及速度控制（运动控制）

舵机的角度控制范围：0-360 度，控制精度：0.09 度

舵机的速度控制范围为：0-280 度/s。控制精度：1 度/s

舵机零点修改

当舵机的零点角度有偏移时可以进行舵机零点的修正

舵机锁定、解除

当舵机锁定时，舵机在收到外界作用力时不会运动（除非外力超过舵机最大输出力矩）

当解除锁定时，舵机在收到外界作用力时会发生运动（除非外力小于减速机自身阻力）

舵机状态查询

此命令可以查询总线内舵机的 ID 以及对应位置

舵机 ID 修改

此命令可以修改舵机的 ID

PS:

- 360 度 12 位精度意味着对 360 度进行 2^{12} (4096) 份细分，也就是说传感器的精度可以达到 ($360 \div 4096 \approx 0.09$ 度) 的水平
- 在一个 CAN 总线中，可以连接多个舵机，每个舵机都有一个独特的 ID 号码。控制器发出的指令包含 ID 信息，只有与指令中 ID 号匹配的舵机才能完整地接收并回应该指令。

硬件准备：

24V 电源、导线、接线器若干

一代优必选舵机 60kg/25kg 舵机，需要是从第一代克鲁泽机械臂拆卸下来的舵机：（二代的发热量高，不推荐）

二代表面有罗马数字
II，一代则没有

一代

二代



带线端子，端子采用 4pin 5264 的带线端子，不知道到哪里购买的可参考淘宝链接：

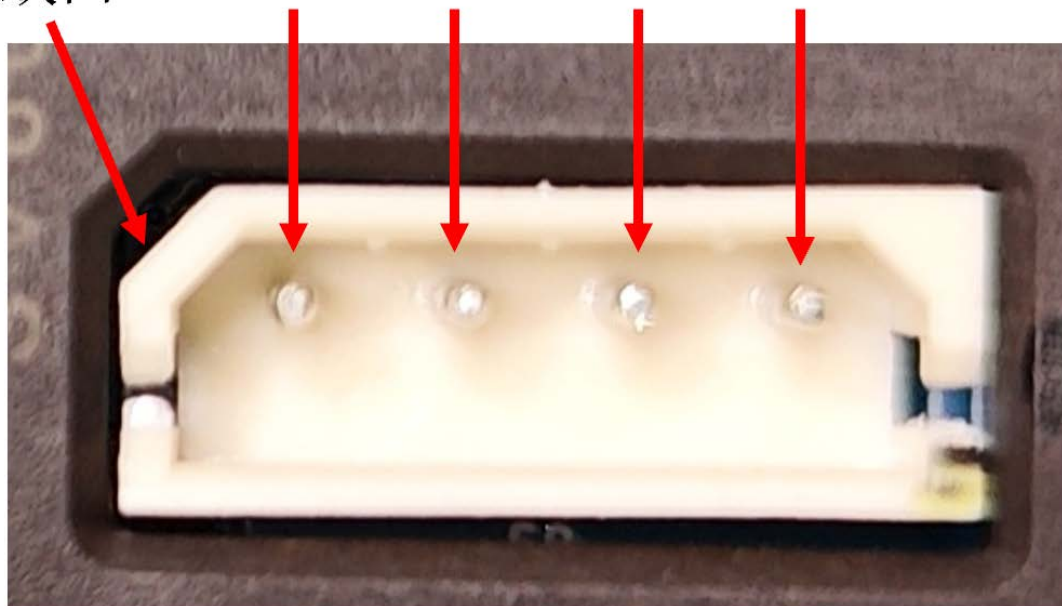
[鐳喲搨璇㄰儁 CZ3457 「5264 端子线 单双头彩色连接线 2P-6P 间距 2.54mm 电子线 26awg」](#) 点击链接直接打开 或者 淘宝搜索直接打开

选择 4P 即可



- 舵机引脚定义：（附图

缺口 24V GND CANH CANL



注意好缺口位置，按图接线，CANH 对应 USB 转 CAN 模块的 CANH，CANL 对应 USB 转 CAN 模块的 CANL

- 因为需要和舵机进行 CAN 通信，所以需要准备相应硬件（单片机，USB 转 CAN 模块），本文的例子都是由电脑

端控制，通过 USB 转 CAN 模块对舵机进行控制，CAN 模块需要购买，视频同款模块淘宝链接：

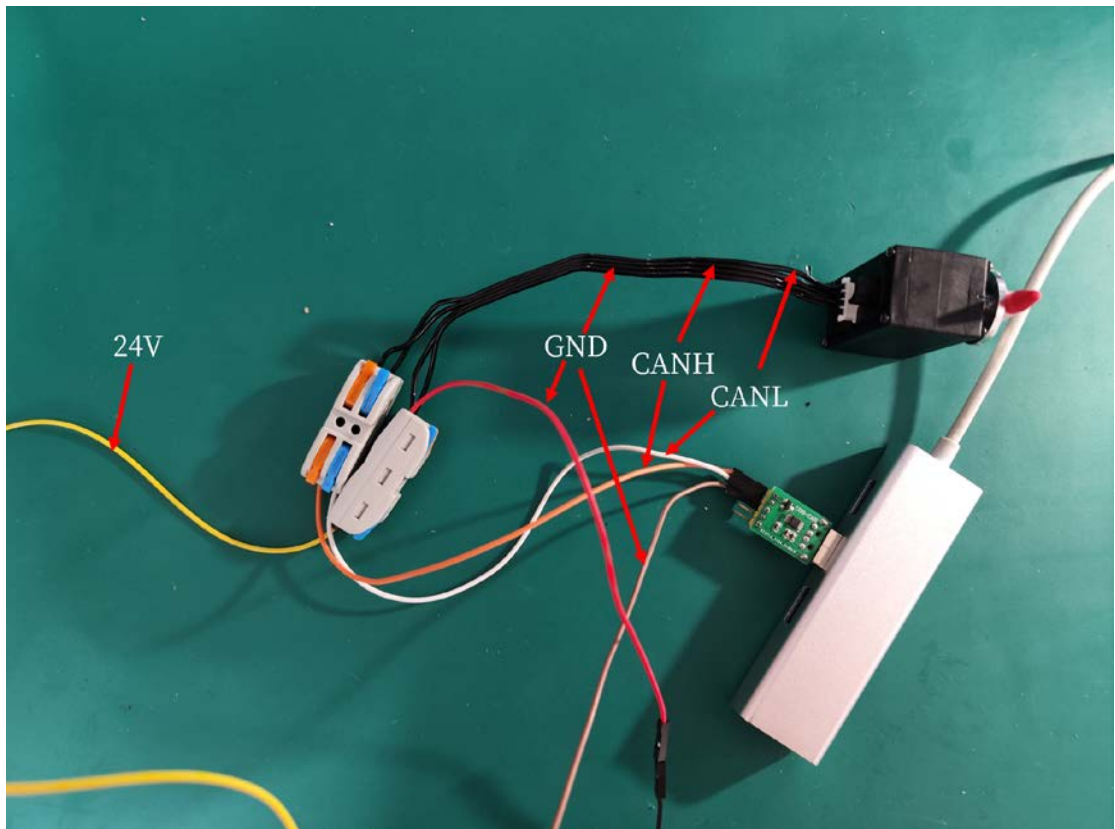
【淘宝】

[USB 转 CAN modbus CANOpen 工业级转换器 CAN 分析仪 串口转 CAN TTL-淘宝网](#)
CZ0001 「USB 转 CAN modbus CANOpen 工业级转换器 CAN 分析仪 串口转 CAN TTL」

点击链接直接打开 或者 淘宝搜索直接打开

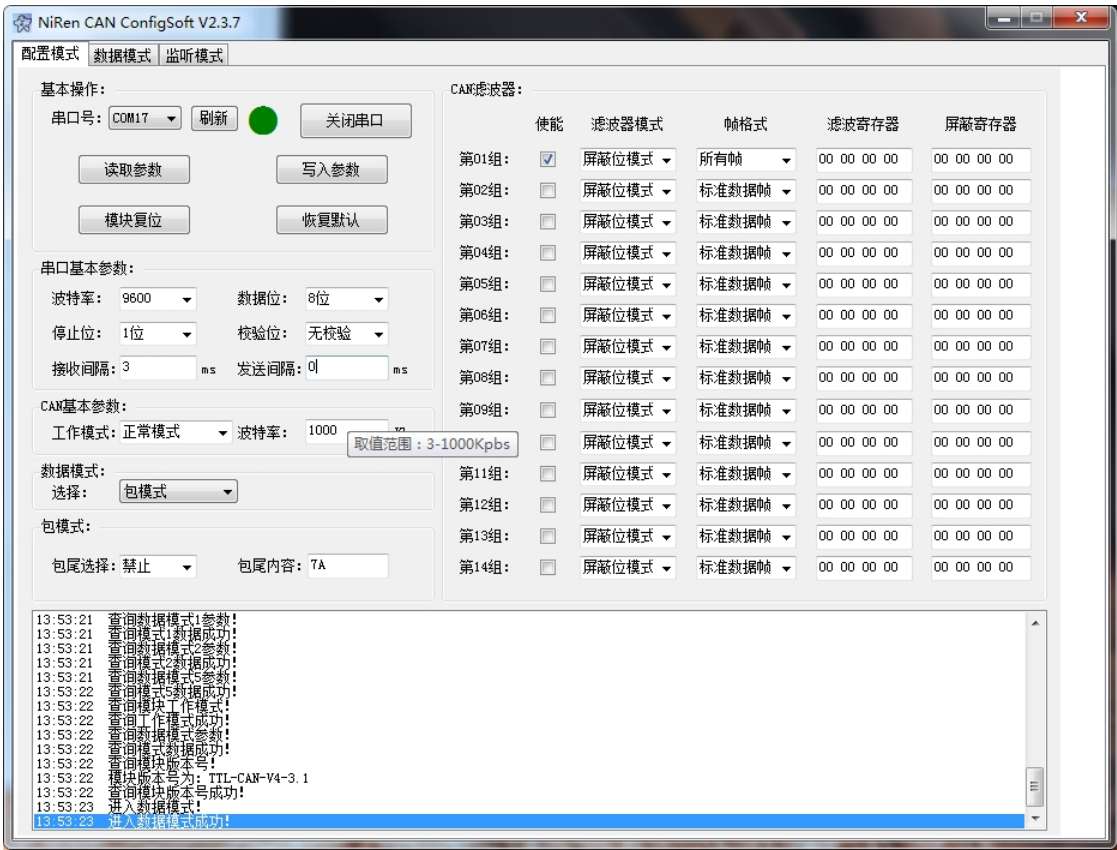
，只要可以进行 CAN 通信，买哪款都可以。

硬件接线如图



上位机调试及参数设定


本文所使用的 USB 转 CAN 模块的上位机具有配置模式、数据模式，接下来以这个模块为例进行 CAN 总线参数的设置



选择正确串口，打开串口，读取模块目前参数，按下图进行参数设置

1. 选择正确的串口

基本操作：

串口号：COM17 刷新  关闭串口

读取参数 写入参数

模块复位 恢复默认

串口基本参数：

波特率：9600 数据位：8位

停止位：1位 校验位：无校验

接收间隔：3 ms 发送间隔：0 ms

CAN基本参数：

工作模式：正常模式 波特率：1000 取值范围：3-1000000

数据模式：

选择：包模式

包模式：

包尾选择：禁止 包尾内容：7A

2. 打开串口

5. 写入参数

3. 读取参数

4. 参数设置

进入数据模式发送命令



舵机通信协议解析及示例

指令包

字头	帧信息	ID	参数
0xAA	数据的长度	ID 号	对应参数

PS:

- 在通讯协议中，“0x”是一个常用的前缀，表示接下来的数字是十六进制（base 16）的。例如，“0xFF”表示一个十六进制的数，其中“F”在十六进制中对应于十进制的15，所以“FF”在十六进制中表示的数值就是 $(15 \times 16) + 15 = 255$ 。类似的，“0x10”在十六进制中表示的数值就是 $(1 \times 16) + 0 = 16$ 。使用十六进制的一个主要原因是它可以更方便地表示二进制数，因为每一个十六进制的位都可以精确地对应四个二进制的位。例如，二进制的“1111”就等于十六进制的“F”。在串口通信中，数据通常是以二进制形式传输的，所以经常用十六进制来表示和操作这些数据。例如，你可能会看到类似“0xFF”或“0x00”这样的表达，分别对应于二进制的“11111111”和“00000000”。
- 二进制（binary）是一种计数系统，只使用两个数字0和1来表示数值。在计算机系统中，二进制被广泛应用，因为计算机处理器只能理解二进制数据。十进制

(decimal) 是我们平时使用的计数系统，使用 10 个数字 0~9 来表示数值。十六进制 (hexadecimal) 也是一种计数系统，使用 16 个数字 09 和字母 AF (分别表示 10~15) 来表示数值。在计算机系统中，十六进制被广泛应用，因为它可以方便地表示二进制数，而且十六进制数的位数比二进制数少得多，便于人们阅读和输入。在计算机编程中，常常使用二进制、十进制和十六进制来表示数据和内存地址等信息。

应答包

字头	帧信息	ID	参数	尾
0xAA	数据的长度	ID 号	对应参数	固定格式

不同的指令包对应的应答包的功能参数不同

指令类型

功能指令	功能	值	参数长度 (字节)
运动控制	位置及速度控制	0x03	4
读取信息	查询 ID、位置	0x01	0
锁定	锁定舵机	0x11	0
解锁	解锁舵机	0x2f	0
ID 修改	修改 ID	0x07	5

示例 1：运动控制

■ 让 ID 为 13 的舵机以 150 度/s 的速度运动到 180 度

发送：AA 00 00 05 00 00 00 0D 03 00 08 96 00 00 00 00

接收：AA 00 00 03 00 00 00 0D 03 04 AA 96 00 00 00 00

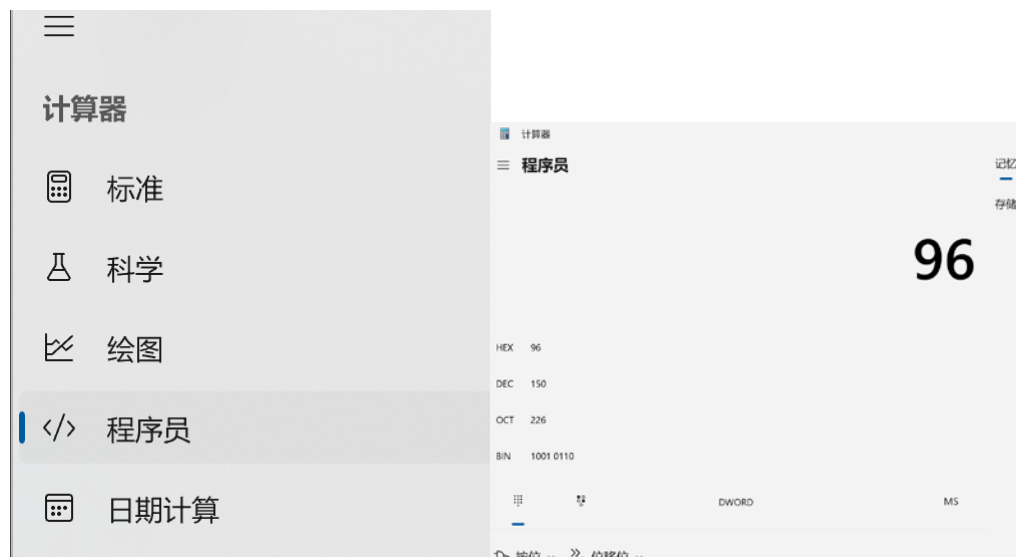
解析：
00 00 05 对应数据长度 (03 00 08 96 00) ；
00 00 00 0D 对应舵机 ID (0x0D=13) ；
03 对应功能指令
00 08 表示位置，对应 0x00 0x08(位置低字节以及高字节)，转化为 10 进制可表示为 2048，此通讯协议中角度转化计算需要移位(因为其低字节在前，高字节在后)，即 0x00

0x08 变为 0x08 0x00, 转化为十进制后为 2048 ($8 \times 16 \times 16 + 0 \times 16 + 0 = 2048$), 转化角度为 180 度 ($2048 \times \frac{360}{4096} = 180$ 度);

96 00 对应速度为 11 度/s;

Ps:

- 在串口通讯中, 16 进制数 0x00 0x08 在命令中需要调换顺序, 变成 0x08 0x00, 是因为串口通讯传输的数据是按照字节(byte)为单位传输的。在大端字节序中, 数据的高字节在前、低字节在后; 而在小端字节序中, 数据的低字节在前、高字节在后。通常, CPU 的内部数据存储方式都采用一种字节序, 而通讯协议中定义的数据字节序可能与 CPU 内部的字节序不同。因此, 为了避免字节序的不一致, 通讯协议中规定了数据的字节序。在该通讯协议中, 发送命令中的数据采用小端字节序, 即低字节在前、高字节在后。因此, 16 进制数 0x00 0x08 在发送的命令中需要调换顺序, 变成 0x08 0x00。这样接收方才能正确地解析出数据。
- Windows 计算器可以进行进制换算, Hex 为 16 进制、DEC 为 10 进制



示例 2: 查询:

查询命令是查询总线上所有的舵机, 如果总线上有多个舵机会收到多个回复, 示例里总线上有一个 ID 为 13 的舵机:

发送: AA 00 00 01 00 00 00 00 01 00 00 00 00 00 00 00

接收: AA 00 00 08 00 00 00 0D 02 03 08 62 60 17 10 10

01: 查询命令

00 00 00 0D: 舵机 ID

03 08: 舵机位置, 可转换为 179.3 度 (2040)

示例 3: 舵机 ID 修改:

舵机的 ID 修改填写舵机的唯一码, 如果不知道舵机唯一码需要发两条命令——查询唯一码及修改 ID, 本例将 ID 为 13 的舵机修改为 ID 为 3

首先获取舵机的唯一码

发送: AA 00 00 01 00 00 00 0D 07 00 00 00 00 00 00 00

接收: AA 00 00 06 00 00 00 0D 08 00 5A 8A 8C 74 10 10

5A 8A 8C 74: 舵机唯一码

修改 ID:

发送: AA 00 00 06 00 00 00 0D 07 03 5A 8A 8C 74 00 00

接收: AA 00 00 02 00 00 00 03 08 03 5A 8A 8C 74 10 10

可以看到 ID 已经成功从 13 修改到 3

示例 4: 舵机锁定与解锁(舵机 ID 为 13):

■ 锁定:

发送: AA 00 00 01 00 00 00 0D 11 00 00 00 00 00 00 00

接收: AA 00 00 01 00 00 00 0D 12 0D 5A 8A 8C 74 10 10

11: 锁定指令

■ 解锁:

发送: AA 00 00 01 00 00 00 0D 2f 00 00 00 00 00 00 00

接收: AA 00 00 01 00 00 00 0D 30 0D 5A 8A 8C 74 10 10

2f: 解锁指令

如果你知道更多功能欢迎私信我补充更新