WJ-S8 控制器用户手册

更新记录：

1. 2017-11-12 v1.0 创建文档
2. 2017-12-14 v1.1 修订
3. 2017-12-16 v1.2 重新整理 IO 分配
4. 2019-01-20 v3.4 与固件一同升级
5. 2019-07-27 v3.6 使用新板卡，修正 IO
6. 2019-09-22 v3.7 删除旧板卡
7. 2019-11-10 v3.8 修正通用输入引脚分配

一 概述

WJ-S8 控制器是无极电子工作室在总结本工作室几大热门步进电机加减速算法的基础上，结合当前市场最新技术，为客户步进电机应用提供一站式服务的平台。

这是给步进电机驱动器发脉冲的装置，它从客户端获取指令，然后控制步进电机运行。控制器的外观如图 1 所示。

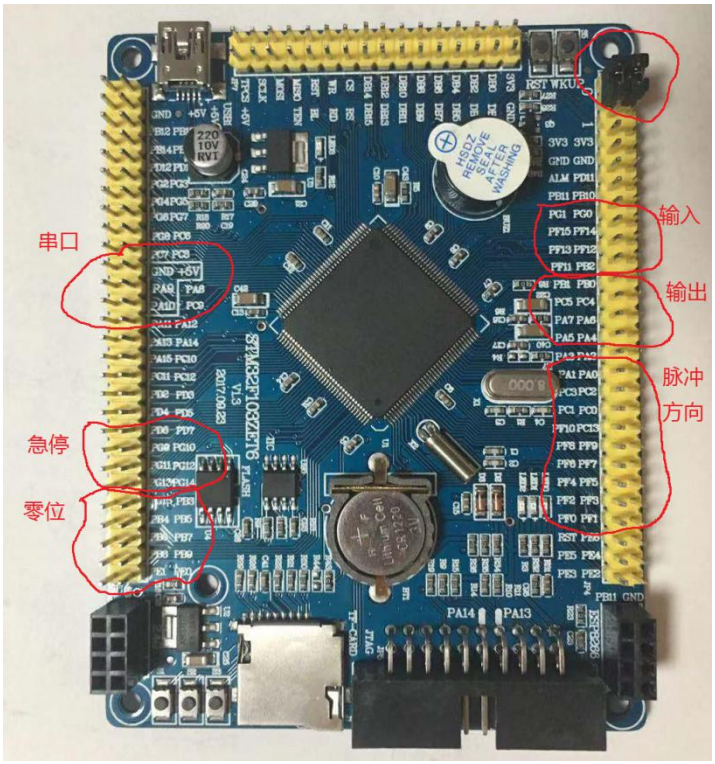


图 1 WJ-S8 控制器

该控制器特色如下：

1. 8 轴独立定时器控制
2. 平滑加减速
3. 串口指令控制
4. 加速度，减速度，最大速度，脉冲数可设置
5. 速度模式或者脉冲模式
6. 支持急停与缓停
7. 基于 ARM 内核

8. 16 路通用 IO(8 路输入，8 路输出)
9. 定点优化算法
10. 专用测试工具
11. 参数掉电保存
12. 参数一键恢复
13. 运行状态查询
14. 脉冲数目实时监控
15. 一键回零

下面从步进电机系统的组成原理开始讲解使用过程，如果您希望全面了解控制器的指令，建议从头到位阅读；如果您想快速让自己的系统动起来，可以跳过前面的内容，直接从第七部分《快速上手》，开始阅读

二 系统组成

以步进电机为执行器的系统，除去客户端，一般包含 3 大部分，如图 2 所示：

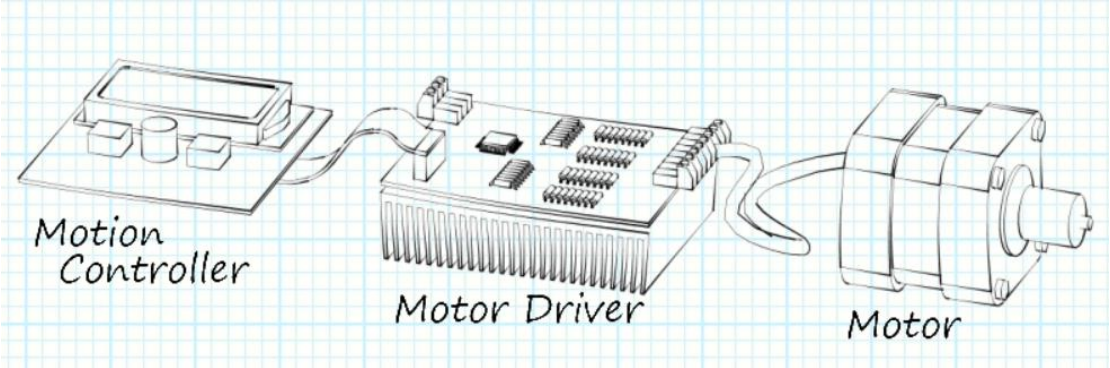


图 2 步进电机系统常见配置

1. 步进电机本体（Motor）
   最为常见的是两相四线步进电机，步距角 1.8 度，不细分的话，就是 200 个脉冲转一圈。这种电机随处可买，价格根据功率大小与制作工艺而变。
2. 步进电机驱动器（Motor Driver）
   驱动器是给电机线圈通电的装置，它接受从电机控制器的脉冲和方向信号，进而转换为电机线圈的励磁电流，从而驱动电机动作。常见的驱动器包含电流选择与步距角细分功能，好一点的带自测功能，不需要外接脉冲就可以转动电机。
3. 步进电机控制器（Motion Controller）
   电机控制器是用来给驱动器发脉冲的，脉冲的频率由控制器来决定，如果要实现电机平滑加减速，就要根据需要产生变频率脉冲，这些计算都在控制器内部进行。
4. 操作客户端（Client）
   通过某种通信方式，比如串口给电机控制器发指令，从而完成控制步进电机的功能。常见的客户端如串口助手。无极电子开发了专门的客户端，详见后面描述。

以上四个部分，只是最常见的划分方式。在实际使用的时候，特别是小规模的应用，经常把驱动器和控制器做在一块板上。本工作室推出的是控制器，后面需要连接驱动器才能正常工作，后续无极电子会推出驱动控制一体化的板卡，敬请期待…

三 指令格式

目前，控制器支持四个大类，第一类是参数设置类(4 条)，第二类是运动类(5 条)，第三类是通用 IO 类(2 条)，第四类是查询类(3 条)。在指令讲解的时候，以通用串口工具来进行，这种方法也是支持二次开发的，只要您给串口发送对应的字符串。

一条指令分为两个字段，每个字段中间用一个空格分割，指令的格式如下所示：

指令类型+编号	参数
M01	200 300 -400 500 200 300 400 -500

第一个字段是指令类型与编号，比如参数设置类为 S(Set)，运动类为 M(Move)；第二个字段是参数，如果指令中需要带参数，在这个段中填写，每个参数用空格分割。

需要特别注意的是，指令最后需要加上回车和换行符号，对应的 16 进制数据为 0xD 和 0xA，否则下位机解析命令帧不对。

四 指令列表

4.1 运动类

运动类指令一共包括：

4.1.1 M01 位置模式运动

该运动指令带参数，就是脉冲数，对于一圈 800 脉冲的电机，如果要转一圈，需要执行如下指令。

M01 800 800 -800 -1600 -800 800 -400 800

脉冲数接受负数，表示电机往另外一个方向旋转。这个指令有需要注意的地方，首先八个参数必须完整，如果不转的电机，对应参数位置写 0，不能省略；其次，最后一个参数后面，也要跟一个空格。当然，如果您使用无极电子开发的专用界面，不需要考虑这么多。

4.1.2 M02 速度模式运动

该指令接受四个参数，最大速度，比如

M02 100 300 500 700 200 300 200 300

这样对应电机就朝着一个方向不停运行，直到发送急停或缓停命令，该参数同样可以用负数来表示反向，速度的单位是 r/min。速度模式的时候，如果设置的速度过大，电机会采用逐步加速的方式启动，因此不用担心速度设置过大导致无法启动。建议使用 60r/min，这样刚好每秒钟 1 转，方便观察。

4.1.3 M03 急停

该命令用于停止电机，不论什么时候，想要停止电机都可以用这个命令。

M03 1 1 1 1 1 1 1 1

该命令的参数为 1 表示需要停止某个电机，如果不想停止，用 0。

4.1.4 M04 缓停

指令的格式与 M03 相同，不同的是，这个时候电机会慢慢停下来。缓停的减速度用 S05 指令来调整，参考后面。

4.1.5 M05 回零

指令格式 M05 100 100 100 100 100 100 100 100
表示以 100r/min 的速度向一个方向运动，直到检测到零点，然后会自动向反方向运动一小段距离，作为轴初始位置。零点对应的端口分别为（轴 0 到轴 7）：PD7，PG9~PG15，回零端口当从高电平变为低的时候，触发一次。默认端口已经上拉到高电平，所以在运动过程中，变为低就触发。

4.2 参数设置类

参数设置类指令用于修改电机的运行参数，比如加速度，减速度，最大速度，每圈脉冲数，对于用户来说，每圈脉冲数是根据驱动器拨码开关设置来定的，因此这个参数是最需要修改的，其余可以使用默认，不修改。

4.2.1 S01 设置加速度

S01 200 200 200 200 200 200 200 200
该指令带一个参数，加速度值 200rad/s^2

4.2.2 S02 设置减速度

S02 200 200 200 200 200 200 200 200
参数含义同 S01 指令

4.2.3 S03 设置最大速度

S03 100 100 100 100 100 100 100 100
参数的含义是最大速度，因为加减速过程中，由于脉冲总数和加速度的设置，能否达到最大速度不定，因此这里的速度是最大速度，表示 100rad/s。举个例子，如果你的加速度足够小，脉冲数又很小，那么设置一个超级大的速度 1000rad/s，电机都没到这个速度，就已经用完脉冲数了，当然不可能实现。

4.2.4 S04 设置每圈脉冲数

这个需要参考实际的驱动器设置，一般是 200 的倍数，例如
S04 1600 800 800 1600 3200 800 800 3200

4.2.5 S05 设置缓停减速程度

S05 100 100 100 100

这个数据影响的是 M04 指令，数值越大，缓停过程越长

指令成功执行后，控制器回复 S05 三个字符。

4.2.6 S06 设置 LED 是否闪烁

该指令不带参数，可以切换 LED 灯是否闪烁，一般情况下建议保留原来数值，用于反馈固件运行状态。如果对控制器应答时间有要求，可以关闭。

指令成功执行后，控制器回复 S06 三个字符。

4.2.7 S07 脉冲数目清零

S07 2

表示把第二轴的脉冲数清零，可以把四个轴的脉冲数逐个清零

指令成功执行后，控制器回复 S07 三个字符。

4.2.8 S08 参数恢复

使用 S08，不带参数，可以恢复出厂设置

4.2.9 S09 设置速度模式的加速程度

S09 100 100 100 100

设置 M02 模式运动的加速过程，数值越大，加速过程越长

指令成功执行后，控制器回复 S09 三个字符。

4.2.10 S10 设置回零脉冲数

S10 200 200 200 200 100 100 100 100

设置回零运动的反转脉冲数，M05 执行过程中，遇到零位信号，轴会反向旋转一定脉冲数，S10 可以改变这个数值，默认 200

指令成功执行后，控制器回复 S10 三个字符。

⑾内部保留命令

⑿内部保留命令

4.2.13 S13 设置端口配置模式

S13 5

一共有 16 个 GPIO 端口，分别是 A4~A7，B0~B2，C4~C5，F11~F15，G0~G1

用户可以根据需要，把这些端口配置为输入或者输出，配置过程如下：

8 组端口的配置不是任意的，而是根据下面表格进行

模式编号	输出	输入
1	A4，A5，A6，A7，B0，B1，B2，C4，C5，F11，F12，F13，F14，F15，G0，G1	无
2	A4，A5，A6，A7，B0，B1，B2，C4，C5，F11，F12，F13，F14，F15，G0，	G1
3	A4，A5，A6，A7，B0，B1，B2，C4，C5，F11，F12，F13，F14，F15，	G0，G1
4	A4，A5，A6，A7，B0，B1，B2，C4，C5，F11，F12，F13，F14，	F15，G0，G1
5	A4，A5，A6，A7，B0，B1，B2，C4，C5，F11，F12，F13，	F14，F15，G0，G1
6	A4，A5，A6，A7，B0，B1，B2，C4，C5，F11，F12，	F13，F14，F15，G0，G1
7	A4，A5，A6，A7，B0，B1，B2，C4，C5，F11，	F12，F13，F14，F15，G0，G1
8	A4，A5，A6，A7，B0，B1，B2，C4，C5，	F11，F12，F13，F14，F15，G0，G1
9	A4，A5，A6，A7，B0，B1，B2，C4，	C5，F11，F12，F13，F14，F15，G0，G1
10	A4，A5，A6，A7，B0，B1，B2，	C4，C5，F11，F12，F13，F14，F15，G0，G1
11	A4，A5，A6，A7，B0，B1，	B2，C4，C5，F11，F12，F13，F14，F15，G0，G1
12	A4，A5，A6，A7，B0，	B1，B2，C4，C5，F11，F12，F13，F14，F15，G0，G1
13	A4，A5，A6，A7，	B0，B1，B2，C4，C5，F11，F12，F13，F14，F15，G0，G1
14	A4，A5，A6，	A7，B0，B1，B2，C4，C5，F11，F12，F13，F14，F15，G0，G1
15	A4，A5，	A6，A7，B0，B1，B2，C4，C5，F11，F12，F13，F14，F15，G0，G1

16	A4，	A5，A6，A7，B0，B1，B2，C4，C5，F11，F12，F13，F14，F15，G0，G1
17	无	A4，A5，A6，A7，B0，B1，B2，C4，C5，F11，F12，F13，F14，F15，G0，G1

另外，需要注意的是，输出通道依次从 B0 到 B15 为通道 1 到通道 16，输入通道从 B15 反过来到 B0 为通道 1 到通道 8，例如对于默认的模式 9，输出通道为：

A4-输出 1，A5-输出 2，A6-输出 3，A7-输出 4 等

G1-输入 1，G0-输入 2，F15-输入 3，F14-输入 4 等

这个在 I01 获取输入数据指令的时候需要注意

4.3 通用 IO 类

4.3.1 P01 通用输出

P01 2 1

要想利用控制器输出一个 bit 的信号，可以采用 P01 指令，后面跟着两个参数，第一个参数指定轴号，从 1 到 8，第二个参数，为 0 表示输出低电平，为 1 表示输出高电平，其余值表示不变。

通用输出一共有 8 路，分别对应物理通道为(从低到高)：PA4，PA5，PA6，PA7，PC4，PC5，PB0，PB1

4.3.2 I01 通用输入

要想读取 8 路通用输入的电平值，用 I01，串口将返回 8 路输入电平值。通用输入对应通道为(从低到高)：PB2，PF11，PF12，PF13，PF14，PF15，PG0，PG1。

需要注意的是，这两个指令受到 S13 指令的影响

4.4 查询类

查询类指令包含：

4.4.1 Q01 查询固件版本号

查询固件版本号用于确认控制器的固件版本，不带参数。

4.4.2 Q02 查询参数

Q02 带一个参数，就是轴号，从 1 到 8

可以查询指定的轴参数，排列顺序为加速度，减速度，最大速度，细分

4.4.3 Q03 查询轴运动状态

返回八个电机的运动状态，1 表示正在运行中，0 表示停止运行

4.4.4 Q04 查询轴脉冲数

不带参数，返回当前各个轴的脉冲数目

五 SDK 函数说明

5.1 通信函数

函数原型	INT32 WJ_Open(int num_scom)
函数功能	手动打开串口/USB，并做相关初始化工作
输入参数	num_scom 表示串口号，0 表示打开 USB
输出参数	无
函数返回	0 函数执行成功；非 0 函数执行失败

函数原型	INT32 WJ_Close()
函数功能	手动关闭串口/USB，并做相关释放析构工作
输入参数	无
输出参数	无
函数返回	0 函数执行成功；非 0 函数执行失败

5.2 查询函数

函数原型	INT32 WJ_Get_Axis_Acc(INT32 AxisNUM, INT32* pValue)
函数功能	查询轴加速度
输入参数	AxisNUM 电机轴控制序号
输出参数	pValue 指定轴的加速度（单位 rad/s2）地址
函数返回	0 函数执行成功；非 0 函数执行失败

函数原型	INT32 WJ_Get_Axis_Dec(INT32 AxisNUM, INT32* pValue)
函数功能	查询轴减速度
输入参数	AxisNUM 电机轴控制序号
输出参数	pValue 指定轴的减速度（单位 rad/s2）地址
函数返回	0 函数执行成功；非 0 函数执行失败

函数原型	INT32 WJ_Get_Axis_Vel(INT32 AxisNUM, INT32* pValue)
函数功能	查询轴最大速度
输入参数	AxisNUM 电机轴控制序号
输出参数	pValue 指定轴的速度（单位 Hz/s）地址
函数返回	0 函数执行成功；非 0 函数执行失败

函数原型	INT32 WJ_Get_Axis_Subdivision(INT32 AxisNUM, INT32* pValue)
函数功能	查询轴细分数
输入参数	AxisNUM 电机轴控制序号
输出参数	pValue 指定轴的细分地址
函数返回	0 函数执行成功；非 0 函数执行失败

函数原型	INT32 WJ_Get_Axis_Status(INT32 AxisNUM, INT32* pValue)
函数功能	查询轴状态
输入参数	AxisNUM 电机轴控制序号
输出参数	pValue 指定轴的状态地址，1 表示正在运动，0 表示停止。
函数返回	0 函数执行成功；非 0 函数执行失败

函数原型	INT32 WJ_Get_Axes_Status(INT32* pValueAxes)
函数功能	查询所有轴状态
输入参数	无
输出参数	pValueAxes 所有轴状态首地址（需根据轴数传入四或八元素数组首地址）
函数返回	0 函数执行成功；非 0 函数执行失败

函数原型	INT32 WJ_Get_Axis_Pulses(INT32 AxisNUM, INT32* pValue)
函数功能	查询轴脉冲计数
输入参数	AxisNUM 电机轴控制序号

输出参数	pValue 指定轴的脉冲数地址
函数返回	0 函数执行成功；非 0 函数执行失败

函数原型	INT32 WJ_Get_Axes_Pulses(INT32* pValueAxes)
函数功能	查询所有轴脉冲计数
输入参数	无
输出参数	pValueAxes 所有轴脉冲数首地址（需根据轴数传入四或八元素数组首地址）
函数返回	0 函数执行成功；非 0 函数执行失败

5.3 运动函数

函数原型	INT32 WJ_Move_Axis_Pulses(INT32 AxisNUM, INT32 Value)
函数功能	轴运动一段脉冲
输入参数	AxisNUM 电机轴控制序号，Value 设定轴要走的脉冲数
输出参数	无
函数返回	0 函数执行成功；非 0 函数执行失败

函数原型	INT32 WJ_Move_Axes_Pulses(INT32* pValueAxes)
函数功能	所有轴同时走脉冲
输入参数	pValueAxes 所有轴脉待走脉冲数组首地址（需根据轴数传入四或八元素数组首地址）
输出参数	无
函数返回	0 函数执行成功；非 0 函数执行失败

函数原型	INT32 WJ_Move_Axis_Vel(INT32 AxisNUM, INT32 Value)
函数功能	轴以一定速度运动
输入参数	AxisNUM 电机轴控制序号，Value 设定轴要走的脉冲速度（单位 rad/s）
输出参数	无
函数返回	0 函数执行成功；非 0 函数执行失败

函数原型	INT32 WJ_Move_Axes_Vel(INT32* pValueAxes)
函数功能	所有轴同时走速度

输入参数	pValueAxes 所有轴脉待走速度数组首地址（需根据轴数传入四或八元素数组首地址）
输出参数	无
函数返回	0 函数执行成功；非 0 函数执行失败

函数原型	INT32 WJ_Move_Axis_Emergency_Stop(INT32 AxisNUM)
函数功能	轴急停
输入参数	AxisNUM 电机轴控制序号，0 表示四轴急停
输出参数	无
函数返回	0 函数执行成功；非 0 函数执行失败

函数原型	INT32 WJ_Move_Axis_Slow_Stop(INT32 AxisNUM)
函数功能	轴缓停
输入参数	AxisNUM 电机轴控制序号，0 表示四轴缓停
输出参数	无
函数返回	0 函数执行成功；非 0 函数执行失败

函数原型	INT32 WJ_Move_Axis_Home(INT32 AxisNUM, INT32 Value)
函数功能	轴回零
输入参数	AxisNUM 电机轴控制序号，0 表示四轴回零
输出参数	无
函数返回	0 函数执行成功；非 0 函数执行失败

5.4 设置函数

函数原型	INT32 WJ_Set_Axis_Acc(INT32 AxisNUM, INT32 Value)
函数功能	设置轴加速度
输入参数	AxisNUM 电机轴控制序号（根据控制器不同 1-4 或 1-8）；Value 设置指定轴的加速度 rad/s2
输出参数	无
函数返回	0 函数执行成功；非 0 函数执行失败

函数原型	INT32 WJ_Set_Axis_Dec(INT32 AxisNUM, INT32 Value)
函数功能	设置轴减速度

输入参数	AxisNUM 电机轴控制序号（根据控制器不同 1-4 或 1-8）；Value 设置指定轴的减速度 rad/s2
输出参数	无
函数返回	0 函数执行成功；非 0 函数执行失败

函数原型	INT32 WJ_Set_Axis_Vel(INT32 AxisNUM, INT32 Value)
函数功能	设置轴最大速度
输入参数	AxisNUM 电机轴控制序号（根据控制器不同 1-4 或 1-8）；Value 设置指定轴的速度 rad/s
输出参数	无
函数返回	0 函数执行成功；非 0 函数执行失败

函数原型	INT32 WJ_Set_Axis_Subdivision(INT32 AxisNUM, INT32 Value)
函数功能	设置轴细分
输入参数	AxisNUM 电机轴控制序号（根据控制器不同 1-4 或 1-8）；Value 设置指定轴的细分
输出参数	无
函数返回	0 函数执行成功；非 0 函数执行失败

函数原型	INT32 WJ_Set_Axis_Slow_Stop(INT32 AxisNUM, INT32 Value)
函数功能	设置轴缓停减速程度值
输入参数	AxisNUM 电机轴控制序号（根据控制器不同 1-4 或 1-8）；Value 设置缓停减速程度（默认 100）
输出参数	无
函数返回	0 函数执行成功；非 0 函数执行失败

函数原型	INT32 WJ_Set_Led_Twinkle()
函数功能	设置 Led 灯闪烁
输入参数	无
输出参数	无
函数返回	0 函数执行成功；非 0 函数执行失败

函数原型	INT32 WJ_Set_Axis_Pulses_Zero(INT32 AxisNUM)
函数功能	轴脉冲计数清零
输入参数	AxisNUM 电机轴控制序号（根据控制器不同 1-4 或 1-8）
输出参数	无

函数返回	0 函数执行成功；非 0 函数执行失败

函数原型	INT32 WJ_Set_Default()
函数功能	所有参数恢复出厂设置
输入参数	无
输出参数	无
函数返回	0 函数执行成功；非 0 函数执行失败

函数原型	INT32 WJ_Set_Move_Axis_Vel_Acc(INT32 AxisNUM, INT32 Value)
函数功能	设置速度模式的加速程度
输入参数	AxisNUM 电机轴控制序号（根据控制器不同 1-4 或 1-8）；Value 速度模式的加速程度
输出参数	无
函数返回	0 函数执行成功；非 0 函数执行失败

函数原型	INT32 WJ_Set_Axis_Home_Pulses(INT32 AxisNUM, INT32 Value)
函数功能	设置轴回零后反向运动的脉冲数
输入参数	AxisNUM 电机轴控制序号（根据控制器不同 1-4 或 1-8）；Value 设置回零后反向运动的脉冲数（目前默认 200）
输出参数	无
函数返回	0 函数执行成功；非 0 函数执行失败

5.5 IO 函数

函数原型	INT32 WJ_IO_Output(INT32 IONUM, INT32 Value)
函数功能	设置通用输出电平
输入参数	IONUM 通用输出序号；Value 1 表示高电平，0 表示低电平
输出参数	无
函数返回	0 函数执行成功；非 0 函数执行失败

函数原型	INT32 WJ_IO_Input(INT32 IONUM, INT32* pValue)
函数功能	读取通用输入电平
输入参数	IONUM 通用输出序号

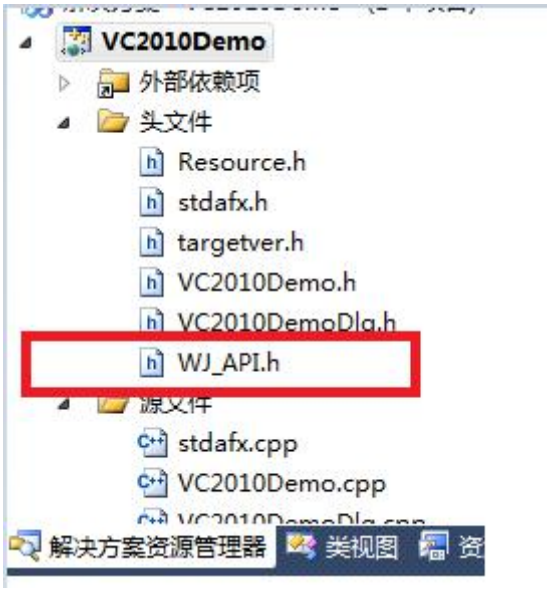
输出参数	pValue1 表示高电平，0 表示低电平；-1 表示无此端口
函数返回	0 函数执行成功；非 0 函数执行失败

六 SDK 函数二次开发方法

6.1 VS2010 二次开发

第一步：将接口文件里的 dll 文件、lib 引导文件、头文件拷贝到自己的 VC 工程文件目录：

第二步：在 VC 工程里手动添加头文件：



第三步：在需要引用的文件中包含此头文件：

第四步：调用函数，本例中调用打开 USB 函数：

```
void CVC2010DemoDlg::OnBnClickedButton2()
{
    // TODO: 在此添加控件通知处理程序代码

    if(!WJ_Open(0))//0表示打开的是USB

        MessageBox("USB连接成功");
    else
        MessageBox("USB连接失败");

}
```

七 串口控制卡操作流程

本工作室秉承为用户节约时间的精神，开发了专用的测试工具，如图 7 所示。当然，您可以根据上面的指令集，通过别的串口工具来测试也是完全没问题的，但是注意一点，该串口工具需要能够在指令末尾添加回车换行符号，无极电子赠送的工具 xcom 就有这个特点，如图 6 所示。

7.1 硬件接线

您拿到 WJ-S8-UART 控制器后，可以看到上面有很多的未连接头，这些都是可以用杜邦线接出的接线端，经常玩单片机的用户，应该非常清楚这样的端子怎么来用。具体怎么把控制器与驱动器连接呢？参考图 4，除了电机的脉冲和方向端口外，还有串口的连接端口，串口线白线接控制器 PA9，绿线接控制器 PA10，5V 与串口的 5V 连接，G 表示 GND，串口位置见图 1。注意：脉冲与方向口的具体引脚名称参考后面的 IO 分配表，图中只是示意。

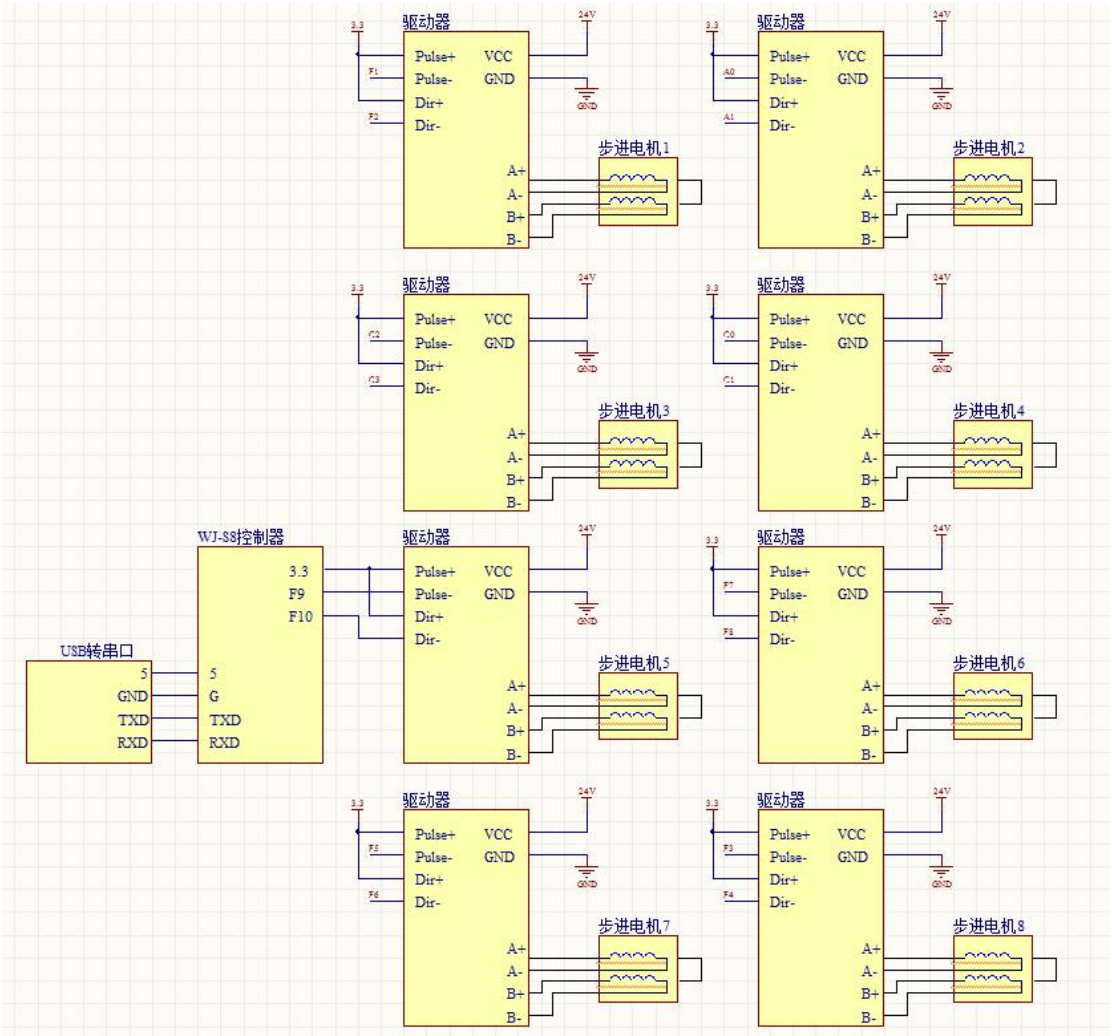


图 4 控制器接线

如图 4 所示，驱动器与控制器的接线主要有 4 路，分别是脉冲正负(Pulse)，方向正负(Dir)，按照共阳极的接法，把脉冲正和方向正接到控制器的 3.3V 上，负极分别接一个 IO 口。另外，驱动器需要供电，用 24V，这个电压范围可调节，具体根据驱动器上的电压范

围来定，比如我们使用的范围是 9~42V。最后，把电机接上，这里需要注意的是，电机的接线不能乱，一定要 A，B 相分开，否则电机是不会转的。怎么判别电机哪两根线是一相呢？很简单，拿个万用表量一下电阻，如果有电阻显示，一般是 5 欧姆以下，那就是一相绕组，否则是断开的。需要注意的是，USB 转串口的工具一头要求是杜邦线接口的，如图 5 所示。

对于 USB 线的另外一头，需要注意，红色接 5V，黑色接 GND，白色接控制器的 TXD，绿色接 RXD。这样就完成所有的连接了。

图 5 USB 转串口

7.2 软件操作

软件操作上，主要分两个流程，第一个是采用通用的串口工具 xcom

①通用串口工具

一般的串口工具都有数据发送和显示功能，如图 6 所示。

图 6 串口工具

首先设置波特率，115200，其余默认；接着选择你计算机中识别的串口号；然后打开。

在打开成功后，在文本输入区输入前面介绍过的指令中任何一条，然后点击发送就可以了，注意，发送新行需要勾上。如果控制器接收成功，会回送串口指令，因此在串口助手的接收区，会显示刚才发送的指令，这里再强调一次，指令的末尾要加一个空格。

②专用工具

无极电子为客户节约时间，因此专门开发了测试工具，如图 7 所示，8 轴控制器支持的所有指令，都可以测试。

界面主要分为：串口设置，版本查询，参数设置，运动设置，IO 采集几个部分。操作的方式类似，如下进行。

选择合适的串口号之后，串口状态会自动切换为红色，表明连接成功。

点击查询版本号，大约 2 秒后，版本号框会显示目前控制器的固件版本。

在各个电机的参数框中填写合适的信息后，点击后边设参按钮，参数就设置成功。

给定电机脉冲数后，点击开始，电机就转指定的脉冲数，如果写 0，表示不转。

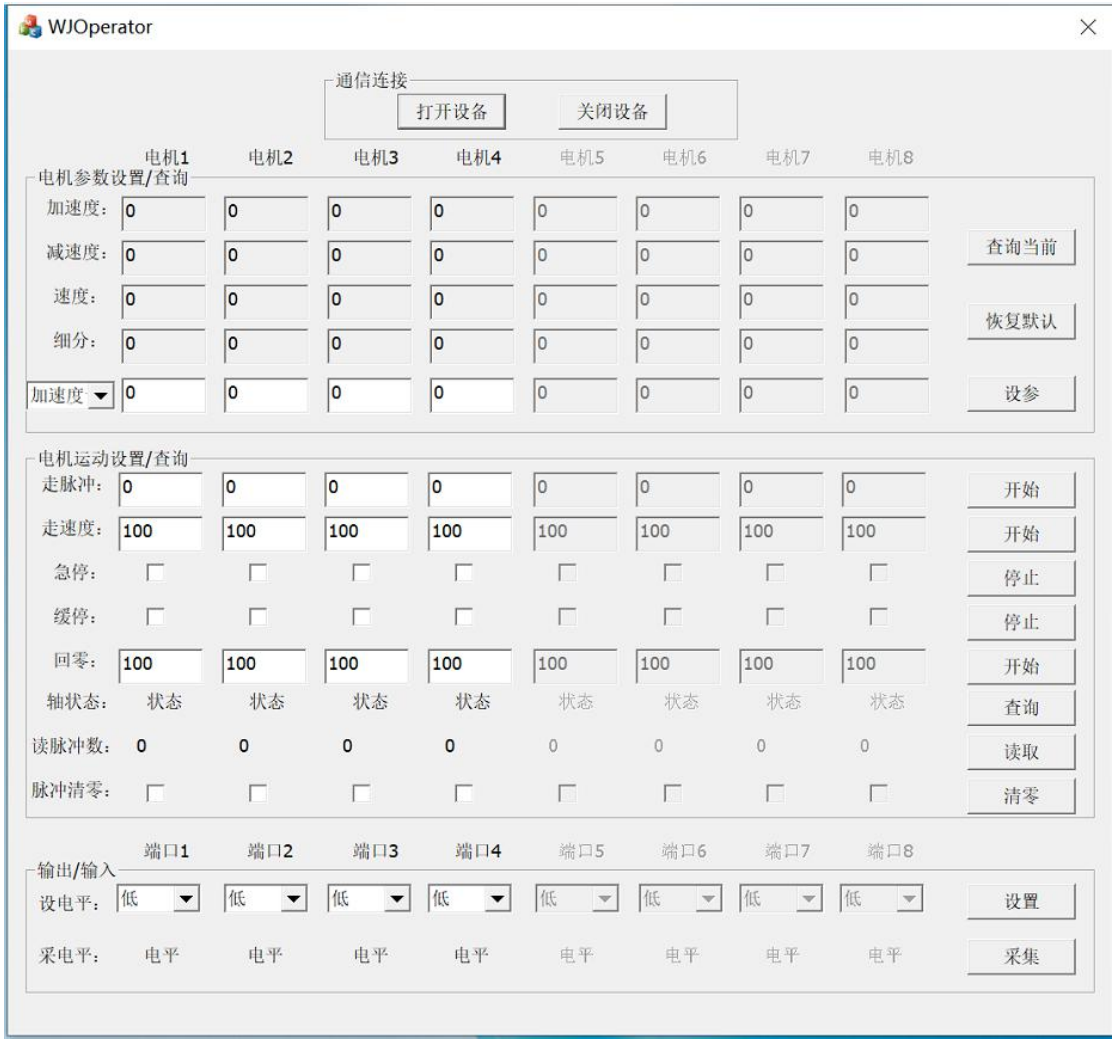


图 7 无极电子 8 轴工具

八 USB 控制卡操作流程

本工作室秉承为用户节约时间的精神，开发了专用的测试工具，如图 11 所示。当然，您可以根据上面的指令集，通过别的 USB 工具来测试也是完全没问题的。

8.1 硬件接线

您拿到 WJ-S8-USB 控制器后，可以看到上面有很多的未连接头，这些都是可以用杜邦线接出的接线端，经常玩单片机的用户，应该非常清楚这样的端子怎么来用。具体怎么把控制器与驱动器连接呢？参考图 9，除了电机的脉冲和方向端口外，控制器的 USB 口如图 8 所示，就是常见的 miniUSB。

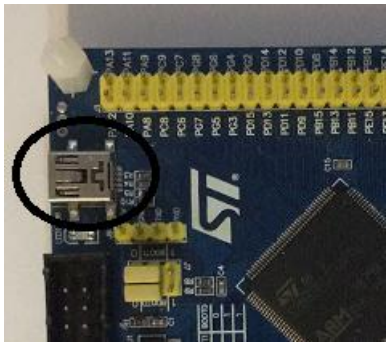


图 8 控制器 USB 口

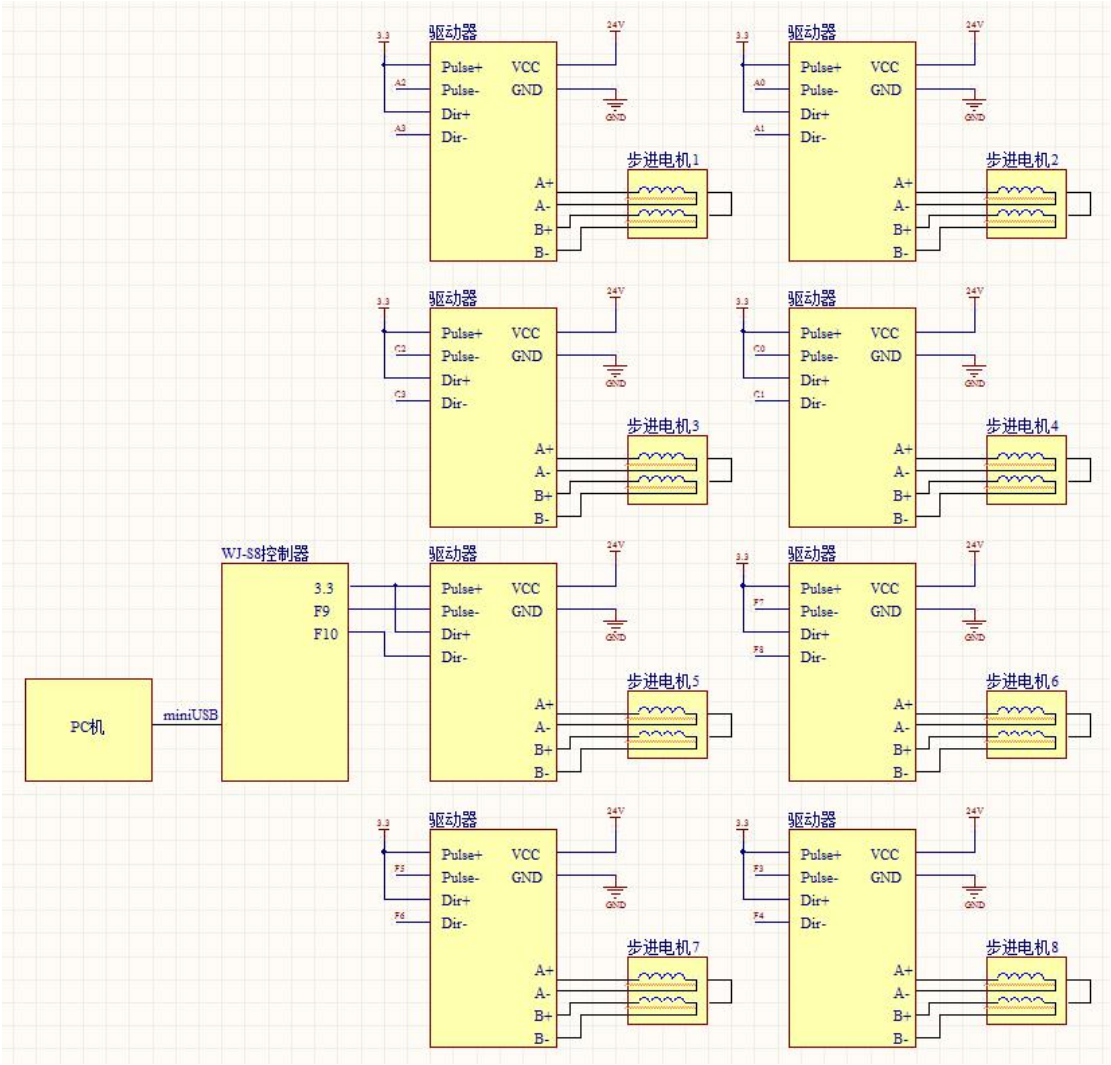


图 9 控制器接线

如图 8 所示，驱动器与控制器的接线主要有 4 路，分别是脉冲正负(Pulse)，方向正负(Dir)，按照共阳极的接法，把脉冲正和方向正接到控制器的 3.3V 上，负极分别接一个 IO 口。另外，驱动器需要供电，用 24V，这个电压范围可调节，具体根据驱动器上的电压范

围来定，比如我们使用的范围是 9~42V。最后，把电机接上，这里需要注意的是，电机的接线不能乱，一定要 A，B 相分开，否则电机是不会转的。怎么判别电机哪两根线是一相呢？很简单，拿个万用表量一下电阻，如果有电阻显示，一般是 5 欧姆以下，那就是一相绕组，否则是断开的。

图 10 USB 线

8.2 软件操作

无极电子为客户节约时间，因此专门开发了测试工具，如图 11 所示，8 轴控制器支持的所有指令，都可以测试。

界面主要分为：版本查询，参数设置，运动设置，IO 采集几个部分。操作的方式类似，如下进行。

1. 打开设备
2. 查询版本号，查询后，显示目前固件版本，有数据表示 USB 口通信正常。
3. 在各个电机的参数框中填写合适的信息后，点击后边设参按钮，参数就设置成功。

给定电机脉冲数后，点击开始，电机就转指定的脉冲数，如果写 0，表示不转。

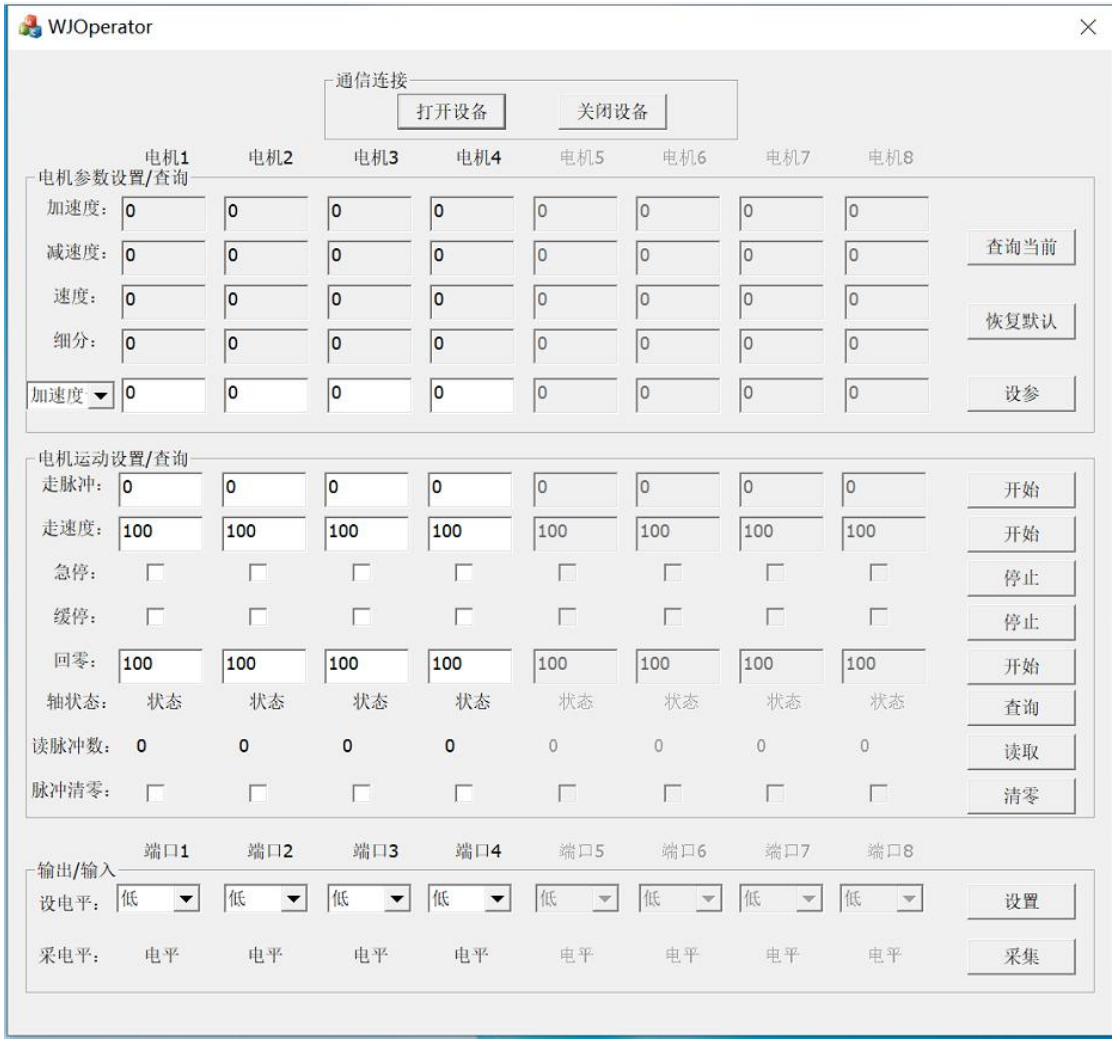


图 11 无极电子 8 轴工具

九 常见问题列表

1. 这个控制器上位机支持 labview 吗？支持 C#的上位机吗？

控制器对于上位机的要求只有串口，所以，不论你是 windows 平台，还是 linux 和 iOS，Android 都无所谓，也不管用什么开发工具，VS，labview，Java 都可以，只要你能开发对应的串口工具，都可以操作该控制器。

2. 控制器带回零功能吗？

带的，参考 M05 指令。

3. 控制器可以接收编码器信号吗？

该控制器是针对步进电机，或者脉冲型伺服电机开发的，一般用于开环控制，因此目前不支持编码器信号输入。

4. 控制器可以刷新固件吗？

该控制器的固件是无极电子经过长时间打造的精品，如果您一定要自己更新，那么无极电子不负责后续的维护。

5. 控制器带插补功能吗？

对插补算法感兴趣的朋友，可以看看店铺内的其余产品，有直线圆弧插补类的，无极电子后续根据需求来移植插补功能。

6. 控制器能控制伺服电机吗？

本控制器发出脉冲来驱动电机，只要是需要脉冲的电机，不论是步进电机，伺服电机还是直线电机，都能使用。

7. 控制器的 ENA 用来干什么？

步进电机的 ENA 用于控制电机使能，这是一种硬件上的急停保护，一般接机械限位开关，本控制器端口资源已经优化，不提供单独的 ENA 接口。

8. 控制器需要开关电源吗？

控制器不需要接开关电源，最方便的使用方式是接上配套的 usb 转串口线，由它来给整个控制器供电，电机驱动器的供电需要开关电源。具体电压见控制器上说明，一般 42 伏特以内。

9. 控制器可以控制哪种步进电机？

原则上可以控制任何一种步进电机，只要驱动器的 IO 信号与控制器匹配。经过测试，除了少数控制器以外，比如山社的 MD 系列，需要 5V 电平驱动。其余的驱动器，都可以使用 3.3V 电平，及时控制器 IO 口上标注了 5V，也还是可以使用本控制器的 3.3V 来驱动的。

10. 串口协议是什么？

本控制器的串口协议是无极电子自己开发的，指令格式为指令码+ID+参数的形式，详细见指令解读部分。

11. 控制器灯不闪，发送指令没反应，为什么？

首先要确认，跳线冒是正常启动状态，如图 17 所示

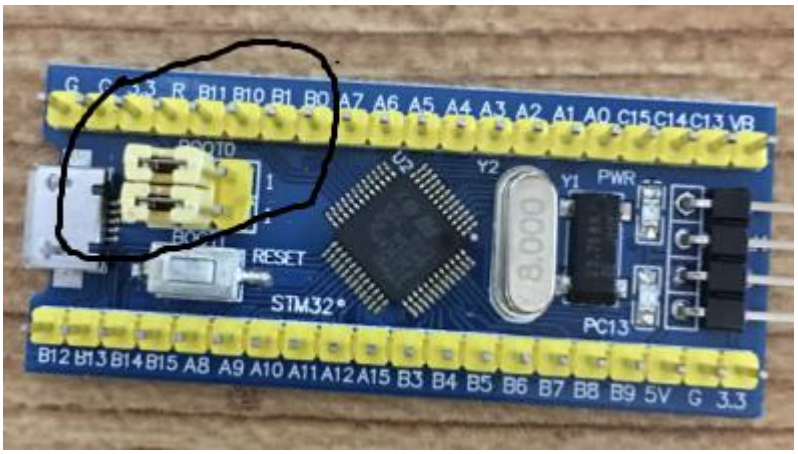


图 17 跳线帽正常状态

12. 我的电机转的过程到最后会嗡嗡叫，突然停止，怎么回事？

首先要保证你的驱动器设置与参数一致，默认是 3200 细分。这里需要注意的是，驱动器的拨码开关，一般是向下为 ON，如下图所示。

13. 为啥我自己开发的程序，发送对应字符串后，板卡无响应

以上的字符串，都有一个结束标记，在 xcom 的使用过程中，其实已经提到了就是回车换行，因此，对应于某个命令，最后发送给串口的应该是以 0xD，0xA 结尾的一帧数据。例如 Q01 命令，最后发送给串口的是'Q' '0' '1' '\r' '\n'，16 进制的话，就是 0x51，0x30，0x31，0xD，0xA

14. 位置模式和速度模式下，速度的单位是什么

位置模式下，速度单位是 rad/s，速度模式下是 r/min，建议测试的时候，用速度模式 60r/min，这样刚好对应 1 转每秒，方便观察。

15. 细分的设置

界面上的细分，真实含义是每圈脉冲数，并不是驱动器上的细分数。例如普通的步进电机，200 个脉冲一圈，8 细分下就是 1600 个脉冲每圈，因此需要设置 1600

16. 关于驱动器

淘宝上有些模块，写的是驱动器，多采用 ULN2003 的那种。无极电子在此提醒，这多半是放大器，没有脉冲分配功能，是不能接本控制器的。

17. 关于运动控制指令的发送周期

M 开头的几个运动控制指令，例如 M01，要求是轴停止的状态下才接受，可以通过查询轴状态 Q03 获取；M02 指令支持在线更新速度，不需要先停止；M05 回零运动，也需要停止的时候发，而且只发一次就能自动回零

18. 指令中的参数范围是多少？

int32 类型，32 位有符号整型数

19. 拍的套件包含啥？

所有物品单独售卖，可以自由组合，比如轴卡+杜邦线+通信线

20. 设置后，怎么确认是否生效

使用 Q02 指令查询轴参数来确认

十 快速上手

当您收到控制器以后，就可以接入您之前的步进电机系统，开始测试了，这里首先要准备如下工具：一套可用的步进电机系统，无极电子开发的客户端，无极电子的控制器。下面详细说说。

1. 一套可用的步进电机系统

这个是要求您的步进电机系统可用，因为如果步进电机驱动器和电机本身就不能动，那么我们的控制器也没法让它动起来，是吧。所以，您需要用别的方式确保步进电机系统能正常转。

2. 无极电子的控制器

在控制器的接线上，最少要求串口线，脉冲，方向线，一共 7 根，如图 13 所示

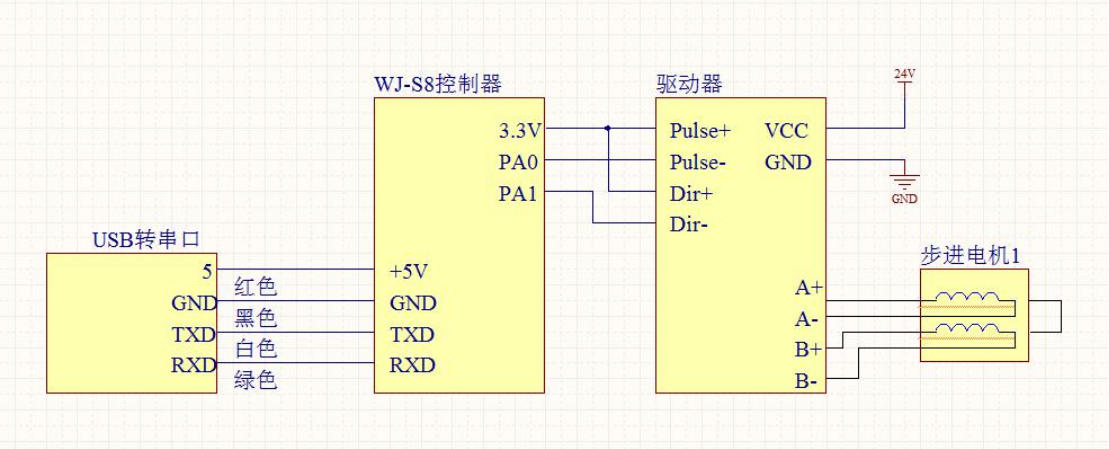


图 13 控制器接线图

其中 USB 转串口线要求是后面开口的杜邦线接头(图 5)，可以与控制器直接连接。连接的顺序为白色连 TXD，绿色连 RXD，红色连 5V，黑色连 GND。

3. 硬件连接准备完毕，打开串口工具(图 7)

可以点击右上角的查询，返回版本号。点击走脉冲行对应的按钮，电机就可以旋转了，如果有问题，欢迎联系客服咨询。

USB 控制卡的操作类似。

十一 IO 分配表

为了方便各位买家接线，这里把所有的 IO 引脚分配整理一下，顺序都是从 0 到 7 列举。

1. 脉冲脚：

PA0 PC2 PC0 PF9 PF7 PF5 PF3 PF1

2. 方向脚

PA1 PC3 PC1 PF10 PF8 PF6 PF4 PF2

3. 通用输出脚（出厂设置）

PA4 PA5 PA6 PA7 PB0 PB1 PB2 PC4

4. 通用输入脚（出厂设置）

PC5 PF11 PF12 PF13 PF14 PF15 PG0 PG1

5. 急停脚

PG9 PG10 PG11 PG12 PG13 PG14 PD6 PD7

6. 零位脚

PE0 PE1 PB4 PB5 PB6 PB7 PB8 PB9

这些引脚的分布如图 14 所示：

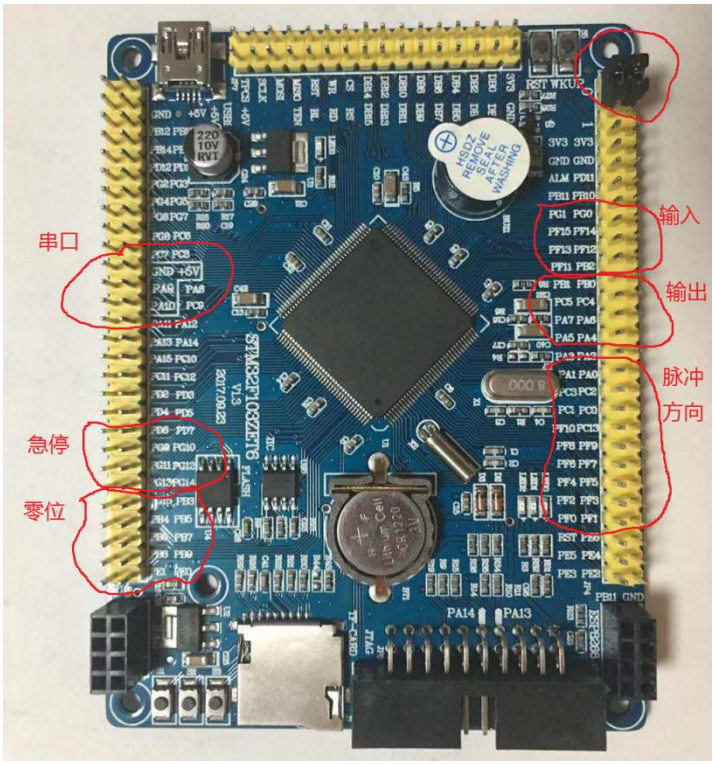


图 14 功能引脚分配

好了，介绍就到这里，更多详情，请关注无极电子。