WLKATA

C++

用户手册

(技术开发文档)

文档版本：V0.1.0

发布日期：2025/10/30

目录

1. WlkataC SDK 简介 - 2 -
1. WlkataC SDK 概述 - 2 -
2. WlkataC SDK 控制函数介绍 - 4 -
1. 调用库 - 4 -
2. 创建机械臂对象(Mirobot) - 4 -
3. 设置机械臂地址 - 4 -
4. 多功能控制器重启 - 5 -
5. 回零 - 6 -
6. 机械臂回到初始位置 - 7 -
7. 执行离线文件 - 8 -
8. 机械臂停止 - 9 -
9. 气泵控制 - 11 -
10. 夹爪控制 - 12 -
11. PWM 控制 - 14 -
12. 透传指令 - 14 -
13. 获取机械臂运动状态 - 14 -
14. 获取机械臂各轴角度 - 16 -
15. 获取机械臂坐标 - 17 -
16. 笛卡尔坐标控制机械臂(Mirobot/Harobot) - 19 -
17. 角度控制机械臂(Mirobot/Harobot) - 23 -

18. 机械臂速度 ..... - 27 -
19. 机械臂拓展轴控制 ..... - 27 -
20. 创建机械臂对象(E4) ..... - 28 -
21. 笛卡尔坐标控制机械臂(E4) ..... - 28 -
22. 角度控制机械臂(E4) ..... - 31 -
23. 创建机械臂对象(MT4) ..... - 34 -
24. 笛卡尔坐标控制机械臂(MT4) ..... - 34 -
25. 角度控制机械臂(MT4) ..... - 37 -
26. 创建步进电机控制器对象（MS4220） ..... - 40 -
27. 设置地址（MS4220） ..... - 40 -
28. 重启（MS4220） ..... - 40 -
29. 回零（MS4220） ..... - 41 -
30. 移动（位置模式）（MS4220） ..... - 42 -
31. 移动（速度模式）（MS4220） ..... - 42 -
32. 获取步进电机控制器状态（MS4220） ..... - 43 -
33. 获取步进电机当前位置（MS4220） ..... - 44 -
34. 获取传感器的状态（MS4220） ..... - 44 -
35. 创建 Mirobot_Serial_GUI 对象 ..... - 45 -
36. 调取 Mirobot_GUI ..... - 45 -

必　读

如实现本手册所示的各种功能，本手册描述的 WlkataC 库版本为 **0.1.0，Mirobot/E4/MT4** 需将多功能控制器固件升级至 **20230710 及以上版本**，Mirobot 机械臂固件升级至 **20231129 及以上版本**，E4 机械臂升级至 **20231031 及以上版本，**MT4 机械臂升级至 **2024082 及以上版本。**Haro380 需将机械臂固件升级到 **20250804 及以上版本，**多功能控制器需升级到 **20250701 及以上版本，Mirobot/E4/MT4** 可下载 **WlkataStudio-Windows-V3.X.X** 配合使用，Haro380 可下载 **Wlkata Lab** 进行下载使用，并对机械臂及多功能控制器进行升级。

更多信息请参照：https://github.com/wlkata/WlkataC.git

1. WlkataC SDK 简介

1. WlkataC SDK 概述

本模块提供 Mirobot 机械臂、E4 机械臂、MT4 机械臂、Haro380 机械臂，步进电机控制器（RS485）的控制方式，需要注意的是控制端需要连接到多功能控制器进行通信，不能控制端直接连接到机械臂本体。控制端与多功能控制器之间可采用 RS485、UART 进行通信。

本库是基于 serial 库进行通信的，故使用本模块的同时需要加载 serial 库。

本库暂不支持 C++98 (ISO/IEC 14882:1998)标准，其它标准无发现问题。

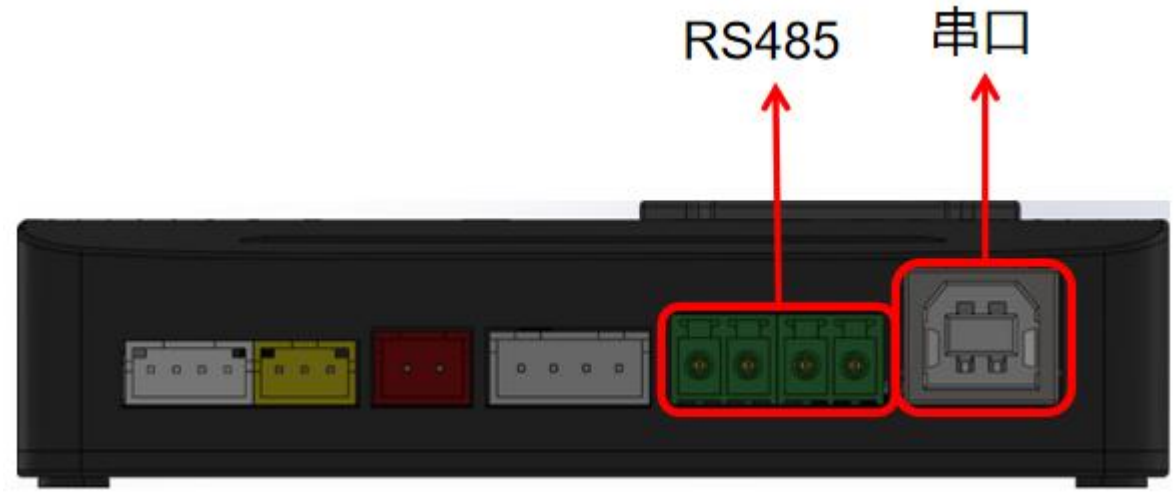
本手册所描述内容对应的模块版本为：

wlkataC	0.1.0
serial	1.2.1

函数所给出的示例程序都是针对控制端与多功能控制器 RS485 通信进行的。

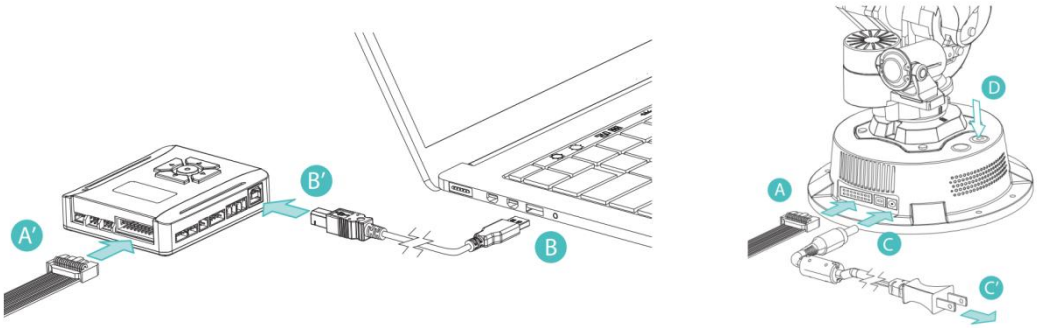
- Mirobot_UART 类主要控制的是 Mirobot 机械臂及其拓展轴。
- E4_UART 类主要控制的是 E4 机械臂及其拓展轴。
- MT4_UART 类主要控制的是 MT4 机械臂及其拓展轴。
- MS4220 类主要控制的是 MS4220 步进电机控制器。
- Harobot_UART 类主要控制的是 Harobot 机械臂及其拓展轴。

Haro380/MT4/E4 类是继承自 Mirobot_UART 类，故除笛卡尔坐标控制和角度控制中的参数不同外其他控制函数通用。详细用法见下文。

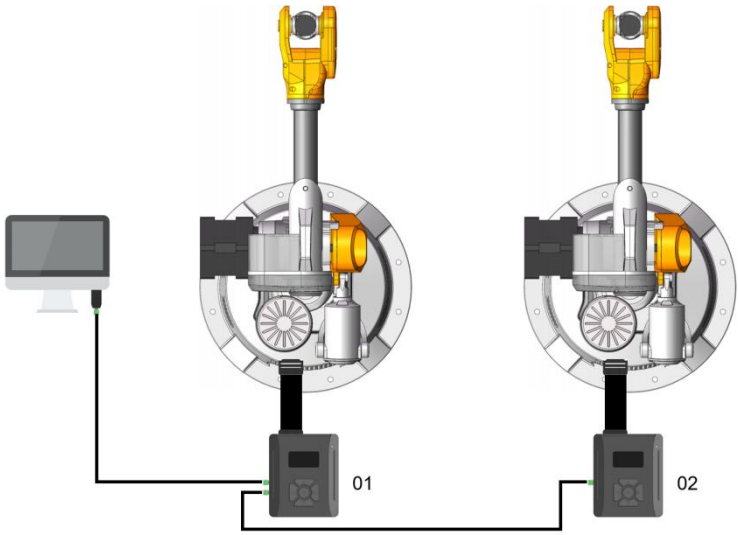


多功能控制器 RS485 设置方法请参照：[多功能控制器用户手册]

UART 接线示意图：



RS485 连接双机械臂接线示意图：



2. WlkataC SDK 控制函数介绍

1. 调用库

函数	#include "wlkataC.h"
函数说明	导入 WLKATA 库文件。
参数说明	无
返回值	无
备注	无

2. 创建机械臂对象(Mirobot)

函数	void Mirobot_UART
函数说明	创建机械臂对象。
参数说明	无

3. 设置机械臂地址

函数	void init(serial::Serial *pserial, int addr, bool block_flag = false, bool message_flag = false);
函数说明	将串口和多功能控制器地址进行绑定。
参数说明	*pserial:串口对象； adr:多功能控制器地址，RS485 模式使用实际地址，UART 模式使用 -1； block_flag：函数是否为阻塞模式；

	message_flag：是否终端打印给机械臂发送和接收的数据；

4. 多功能控制器重启

函数	void restart();
函数说明	控制多功能控制器重启。
参数说明	无

示例（RS485）	#include "serial/serial.h"
	#include "wlkataC.h"
	int main() {
	serial::Serial my_serial("com13", 38400);
	Mirobot_UART Mirobot;
	Mirobot.init(&my_serial, 1， True,True);
	// Mirobot.message_print(true);
	Mirobot.restart();
	my_serial.close();
	system("pause");
	return 0;
	}

示例(UART)	#include "serial/serial.h"
	#include "wlkataC.h"
	int main() {
	serial::Serial my_serial("com13", 115200);
	Mirobot_UART Mirobot;
	Mirobot.init(&my_serial, -1， True,True);
	// Mirobot.message_print(true);
	Mirobot.restart();

	my_serial.close();
	system("pause");
	return 0;
	}

5. **回零**

函数	void homing();
函数说明	机械臂回零操作。
参数说明	无

示例(RS485)	#include "serial/serial.h"
	#include "wlkataC.h"
	int main() {
	serial::Serial my_serial("com13", 38400);
	Mirobot_UART Mirobot;
	Mirobot.init(&my_serial, -1， True,True);
	// Mirobot.message_print(true);
	Mirobot.homing();
	my_serial.close();
	system("pause");
	return 0;
	}

示例（UART)	#include "serial/serial.h"
	#include "wlkataC.h"
	int main() {
	serial::Serial my_serial("com13", 115200);
	Mirobot_UART Mirobot;

	Mirobot.init(&my_serial, 1， True,True);
	// Mirobot.message_print(true);
	Mirobot.homing();
	my_serial.close();
	system("pause");
	return 0;
	}

6. 机械臂回到初始位置

函数	void zero();
函数说明	控制机械臂回到初始位置。
参数说明	无

示例(RS485)	i#include "serial/serial.h"
	#include "wlkataC.h"
	int main() {
	serial::Serial my_serial("com13", 38400);
	Mirobot_UART Mirobot;
	Mirobot.init(&my_serial, 1， True,True);
	// Mirobot.message_print(true);
	Mirobot.zero();
	my_serial.close();
	system("pause");
	return 0;
	}

示例（UART)	#include "serial/serial.h"

	#include "wlkataC.h"
	int main() {
	serial::Serial my_serial("com13", 115200);
	Mirobot_UART Mirobot;
	Mirobot.init(&my_serial, -1， True,True);
	// Mirobot.message_print(true);
	Mirobot.zero();
	my_serial.close();
	system("pause");
	return 0;
	}

7. 执行离线文件

函数	void runFile(string file_name, bool num = false);
函数说明	执行多功能控制器中的离线文件；
参数说明	fileName,：离线文件名称； num：是否循环执行，False 为只执行 1 次，True 为循环执行该离线文件。该参数为选填，不填写默认为 False。

示例(RS485)	#include "serial/serial.h"
	#include "wlkataC.h"
	int main() {
	serial::Serial my_serial("com13", 38400);
	Mirobot_UART Mirobot;
	Mirobot.init(&my_serial, -1， True,True);
	// Mirobot.message_print(true);
	Mirobot.runFile(“test”);

	my_serial.close();
	system("pause");
	return 0;
	}

示例（UART）	#include "serial/serial.h"
	#include "wlkataC.h"
	int main() {
	serial::Serial my_serial("com13", 115200);
	Mirobot_UART Mirobot;
	Mirobot.init(&my_serial, 1，True,True);
	// Mirobot.message_print(true);
	Mirobot.runFile(“test”);
	my_serial.close();
	system("pause");
	return 0;
	}

8. 机械臂停止

函数	void cancellation();
函数说明	机械臂以减速形式停止当前动作，并清理掉缓存动作。注意使用次函数需要将函数设为非阻塞模式。
参数说明	无
示例（RS485）	#include <iostream> #include <chrono> #include <thread> #include "serial/serial.h" #include "wlkataC.h"

	int main() {
	serial::Serial my_serial("com13", 38400);
	Mirobot_UART Mirobot;
	Mirobot.init(&my_serial, 1， false,true);
	// Mirobot.message_print(true);
	Mirobot.runFile(“test”);
	std::this_thread::sleep_for(std::chrono::milliseconds(2000));
	Mirobot.cancellation();
	my_serial.close();
	system("pause");
	return 0;
	}

示例（UART)	#include <iostream>
	#include <chrono>
	#include <thread>
	#include "serial/serial.h"
	#include "wlkataC.h"
	int main() {
	serial::Serial my_serial("com13",115200);
	Mirobot_UART Mirobot;
	Mirobot.init(&my_serial, -1， false,true);
	// Mirobot.message_print(true);
	Mirobot.runFile(“test”);
	std::this_thread::sleep_for(std::chrono::milliseconds(2000));
	Mirobot.cancellation();
	my_serial.close();
	system("pause");
	return 0;

	}

9. 气泵控制

函数	void pump(int num);
函数说明	控制气泵吸气、吹气、关闭。
参数说明	num:0—关闭； 1—吸气； 2—吹气。

示例（RS485）	#include <iostream>
	#include <chrono>
	#include <thread>
	#include "serial/serial.h"
	#include "wlkataC.h"
	int main() {
	serial::Serial my_serial("com13", 38400);
	Mirobot_UART Mirobot;
	Mirobot.init(&my_serial, -1， True,True);
	// Mirobot.message_print(true);
	Mirobot.pump(1);
	std::this_thread::sleep_for(std::chrono::milliseconds(2000));
	Mirobot.pump(2);
	std::this_thread::sleep_for(std::chrono::milliseconds(2000));
	Mirobot.pump(0);
	std::this_thread::sleep_for(std::chrono::milliseconds(2000));
	my_serial.close();
	system("pause");
	return 0;

	}

示例（UART）	#include <iostream>
	#include <chrono>
	#include <thread>
	#include "serial/serial.h"
	#include "wlkataC.h"
	int main() {
	serial::Serial my_serial("com13",115200);
	Mirobot_UART Mirobot;
	Mirobot.init(&my_serial, 1， True,True);
	// Mirobot.message_print(true);
	Mirobot.pump(1);
	std::this_thread::sleep_for(std::chrono::milliseconds(2000));
	Mirobot.pump(2);
	std::this_thread::sleep_for(std::chrono::milliseconds(2000));
	Mirobot.pump(0);
	std::this_thread::sleep_for(std::chrono::milliseconds(2000));
	my_serial.close();
	system("pause");
	return 0;
	}

10. 夹爪控制

函数	void gripper(int num);
函数说明	控制夹爪张开以及闭合。
参数说明	num:夹爪状态——0 夹爪断电，1 夹爪打开，2 夹爪闭合。
示例（RS485）	#include <iostream>

	#include <chrono>
	#include <thread>
	#include "serial/serial.h"
	#include "wlkataC.h"
	int main() {
	serial::Serial my_serial("com13", 38400);
	Mirobot_UART Mirobot;
	Mirobot.init(&my_serial, -1， True,True);
	// Mirobot.message_print(true);
	Mirobot.gripper(1);
	std::this_thread::sleep_for(std::chrono::milliseconds(2000));
	Mirobot.gripper(2);
	std::this_thread::sleep_for(std::chrono::milliseconds(2000));
	Mirobot.gripper(0);
	std::this_thread::sleep_for(std::chrono::milliseconds(2000));
	my_serial.close();
	system("pause");
	return 0;
	}

示例（UART)	#include <iostream>
	#include <chrono>
	#include <thread>
	#include "serial/serial.h"
	#include "wlkataC.h"
	int main() {
	serial::Serial my_serial("com13", 115200);
	Mirobot_UART Mirobot;
	Mirobot.init(&my_serial, 1， True,True);

```
// Mirobot.message_print(true);
Mirobot.gripper(1);
std::this_thread::sleep_for(std::chrono::milliseconds(2000));
Mirobot.gripper(2);
std::this_thread::sleep_for(std::chrono::milliseconds(2000));
Mirobot.gripper(0);
std::this_thread::sleep_for(std::chrono::milliseconds(2000));
my_serial.close();
system("pause");
return 0;
 }
```

11. PWM 控制

函数	void pwmWrite(int num);
函数说明	向 PWM 口输出信号。
参数说明	num:PWM 输出值(0-1000)。

12. 透传指令

函数	void sendMsg(string string_data)；
函数说明	向机械臂传输指令。
参数说明	string_data：需要传输的指令。

13. 获取机械臂运动状态

函数	string getState();
函数说明	查询机械臂状态。

参数说明	无
返回值	返回机械臂运动状态。

示例（RS485）	#include "serial/serial.h"
	#include "wlkataC.h"
	int main() {
	serial::Serial my_serial("com13", 38400);
	Mirobot_UART Mirobot;
	Mirobot.init(&my_serial, -1， True,True);
	// Mirobot.message_print(true);
	cout<<Mirobot.getState()<<endl;
	my_serial.close();
	system("pause");
	return 0;
	}

示例（UART）	#include "serial/serial.h"
	#include "wlkataC.h"
	int main() {
	serial::Serial my_serial("com13", 115200);
	Mirobot_UART Mirobot;
	Mirobot.init(&my_serial, 1， True,True);
	// Mirobot.message_print(true);
	cout<<Mirobot.getState()<<endl;
	my_serial.close();
	system("pause");
	return 0;
	}

14. 获取机械臂各轴角度

函数	float getAngle(int num);
函数说明	获取机械臂各轴角度。
参数说明	Num:1/2/3/4/5/6/7，表示 1-6 轴的角度，7 表示第 7 轴一般是滑轨的角度。
返回值	返回机械臂各轴角度及滑轨的角度。

示例（RS485）	#include "serial/serial.h"
	#include "wlkataC.h"
	int main() {
	serial::Serial my_serial("com13", 38400);
	Mirobot_UART Mirobot;
	Mirobot.init(&my_serial, 1， True,True);
	// Mirobot.message_print(true);
	cout<<Mirobot.getAngle(1)<<endl;
	cout<<Mirobot.getAngle(2)<<endl;
	cout<<Mirobot.getAngle(3)<<endl;
	cout<<Mirobot.getAngle(4)<<endl;
	cout<<Mirobot.getAngle(5)<<endl;
	cout<<Mirobot.getAngle(6)<<endl;
	my_serial.close();
	system("pause");
	return 0;
	}

示例（UART）	#include "serial/serial.h" #include "wlkataC.h" int main() {

	serial::Serial my_serial("com13",115200);
	Mirobot_UART Mirobot;
	Mirobot.init(&my_serial, -1， True,True);
	// Mirobot.message_print(true);
	cout<<Mirobot.getAngle(1)<<endl;
	cout<<Mirobot.getAngle(2)<<endl;
	cout<<Mirobot.getAngle(3)<<endl;
	cout<<Mirobot.getAngle(4)<<endl;
	cout<<Mirobot.getAngle(5)<<endl;
	cout<<Mirobot.getAngle(6)<<endl;
	my_serial.close();
	system("pause");
	return 0;
	}

15. 获取机械臂坐标

函数	float getcoordinate(int num);
函数说明	获取机械臂坐标。
参数说明	Num:1/2/3/4/5/6，分别表示机械臂 X//Y/Z/RX/RY/RZ 坐标。
返回值	返回机械臂 X//Y/Z/RX/RY/RZ 坐标。

示例(RS485)	#include "serial/serial.h"
	#include "wlkataC.h"
	int main() {
	serial::Serial my_serial("com13", 38400);
	Mirobot_UART Mirobot;
	Mirobot.init(&my_serial, 1， True,True);
	// Mirobot.message_print(true);

	cout<<Mirobot.getcoordinate(1)<<endl;
	cout<<Mirobot.getcoordinate(2)<<endl;
	cout<<Mirobot.getcoordinate(3)<<endl;
	cout<<Mirobot.getcoordinate(4)<<endl;
	cout<<Mirobot.getcoordinate(5)<<endl;
	cout<<Mirobot.getcoordinate(6)<<endl;
	my_serial.close();
	system("pause");
	return 0;
	}

示例(UART)	#include "serial/serial.h"
	#include "wlkataC.h"
	int main() {
	serial::Serial my_serial("com13", 115200);
	Mirobot_UART Mirobot;
	Mirobot.init(&my_serial, -1， True,True);
	// Mirobot.message_print(true);
	cout<<Mirobot.getcoordinate(1)<<endl;
	cout<<Mirobot.getcoordinate(2)<<endl;
	cout<<Mirobot.getcoordinate(3)<<endl;
	cout<<Mirobot.getcoordinate(4)<<endl;
	cout<<Mirobot.getcoordinate(5)<<endl;
	cout<<Mirobot.getcoordinate(6)<<endl;
	my_serial.close();
	system("pause");
	return 0;
	}

16. 笛卡尔坐标控制机械臂(Mirobot/Harobot)

函数	void writecoordinate(int motion, int position, float x, float y, float z, float a, float b, float c);
函数说明	笛卡尔坐标控制机械臂。
参数说明	Motion:0—快速运动; 1—直线运动; 2—门型运动。 Position：0—绝对值运动; 1—增量值运动。 x：x 点坐标; y：y 点坐标; z：z 点坐标; a：a 点坐标; b：b 点坐标; c：c 点坐标;
返回值	无

示例（RS485）	#include "serial/serial.h"
	#include "wlkataC.h"
	int main() {
	serial::Serial my_serial("com4", 38400);
	Mirobot_UART Mirobot;
	Mirobot.init(&my_serial, 1, true,true);
	Mirobot.homing();
	for (int i = 0; i < 5; i++) {

	Mirobot.writecoordinate(0, 0, 255, 0, 200, 0, 0, 0);
	Mirobot.writecoordinate(0, 0, 255, 0, 100, 0, 0, 0);
	}
	my_serial.close();
	system("pause");
	return 0;
	}

示例（UART）	#include "serial/serial.h"
	#include "wlkataC.h"
	int main() {
	serial::Serial my_serial("com4", 115200);
	Mirobot_UART Mirobot;
	Mirobot.init(&my_serial, -1, true,true);
	Mirobot.homing();
	for (int i = 0; i < 5; i++) {
	Mirobot.writecoordinate(0, 0, 255, 0, 200, 0, 0, 0);
	Mirobot.writecoordinate(0, 0, 255, 0, 100, 0, 0, 0);
	}
	my_serial.close();
	system("pause");
	return 0;
	}

Mirobot 示例 (UART)单独控制某一坐标 (params.set_a= false 后对用的值	#include "serial/serial.h"
	#include "wlkataC.h"
	int main() {
	serial::Serial my_serial("com4", 115200);
	Mirobot_UART Mirobot;
	Mirobot.init(&my_serial, -1, true,true);
	Mirobot.homing();

将不再起作用)	for (int i = 0; i < 5; i++) {
	Mirobot_UART::CoordinateParams params;
	params.x = 255;
	params.y = 50;
	params.z = 200;
	params.a = 20;
	params.b = 0;
	params.c = 0;
	
	params.set_x = params.set_y = params.set_z = true;
	params.set_a = params.set_b = params.set_c = false;
	Mirobot.writecoordinate(0, 0, params);
	
	Mirobot_UART::CoordinateParams params2;
	params2.x = 255;
	params2.y = 50;
	params2.z = 100;
	params2.a = 20;
	params2.b = 20;
	params2.c = 20;
	params2.set_x = params2.set_y = params2.set_z = true;
	params2.set_a = params2.set_b = params2.set_c = false;
	Mirobot.writecoordinate(0, 0, params2);
	}
	my_serial.close();
	system("pause");
	return 0;
	}

Harobot 示例	#include "serial/serial.h"

(UART)单独控制某一坐标
(params.set_a=false 后对用的值将不再起作用**)**
	#include "wlkataC.h"
	int main() {
	serial::Serial my_serial("com4", 115200);
	Harobot_UART Harobot;
	Harobot.init(&my_serial, -1, true,true);
	Harobot.homing();
	for (int i = 0; i < 5; i++) {
	Harobot_UART::CoordinateParams params;
	params.x = 255;
	params.y = 50;
	params.z = 200;
	params.a = 20;
	params.b = 0;
	params.c = 0;
	
	params.set_x = params.set_y = params.set_z = true;
	params.set_a = params.set_b = params.set_c = false;
	Harobot.writecoordinate(0, 0, params);
	
	Harobot_UART::CoordinateParams params2;
	params2.x = 255;
	params2.y = 50;
	params2.z = 100;
	params2.a = 20;
	params2.b = 20;
	params2.c = 20;
	params2.set_x = params2.set_y = params2.set_z = true;
	params2.set_a = params2.set_b = params2.set_c = false;
	Harobot.writecoordinate(0, 0, params2);

	}
	my_serial.close();
	system("pause");
	return 0;
	}

17. **角度控制机械臂**(Mirobot/Harobot)

函数	void writeangle(int position, float x, float y, float z, float a, float b, float c);
函数说明	笛卡尔坐标控制机械臂。
参数说明	Position：0—绝对值运动； 1—增量值运动。 x：1 轴角度； y：2 轴角度； z：3 轴角度； a：4 轴角度； b：5 轴角度； c：6 轴角度；
返回值	无

示例(RS485)	#include "serial/serial.h"
	#include "wlkataC.h"
	int main()
	{
	serial::Serial my_serial("com4", 38400);
	Mirobot_UART Mirobot;
	Mirobot.init(&my_serial, 1, true, true);

	Mirobot.homing();
	for (int i = 0; i < 5; i++)
	{
	Mirobot.writeangle(0, 10, 0, 0, 0, 0, 0);
	Mirobot.writeangle(0, -10, 0, 0, 0, 0, 0);
	}
	my_serial.close();
	system("pause");
	return 0;
	}

示例(UART)	#include "serial/serial.h"
	#include "wlkataC.h"
	int main()
	{
	serial::Serial my_serial("com4", 115200);
	Mirobot_UART Mirobot;
	Mirobot.init(&my_serial, -1, true, true);
	Mirobot.homing();
	for (int i = 0; i < 5; i++)
	{
	Mirobot.writeangle(0, 10, 0, 0, 0, 0, 0);
	Mirobot.writeangle(0, -10, 0, 0, 0, 0, 0);
	}
	my_serial.close();
	system("pause");
	return 0;
	}

Mirobot 示例	#include "serial/serial.h" #include "wlkataC.h"

(UART)单独控制某一轴
(params.set_a= false 后对用的值将不再起作用**)**
	int main() {
	serial::Serial my_serial("com4", 115200);
	Mirobot_UART Mirobot;
	Mirobot.init(&my_serial, -1, true, true);
	Mirobot.homing();
	for (int i = 0; i < 5; i++) {
	Mirobot_UART::AngleParams params;
	params.x = 0;
	params.y = 20;
	params.z = 0;
	params.a = 20;
	params.b = 0;
	params.c = 0;
	params.set_x = params.set_y = params.set_z = true;
	params.set_a = params.set_b = params.set_c = false;
	Mirobot.writeangle(0, params);
	Mirobot_UART::AngleParams params2;
	params2.x = 0;
	params2.y = -20;
	params2.z = 0;
	params2.a = 20;
	params2.b = 20;
	params2.c = 20;
	params2.set_x = params2.set_y = params2.set_z = true;
	params2.set_a = params2.set_b = params2.set_c = false;
	Mirobot.writeangle(0, params2);
	}
	my_serial.close();
	system("pause");

	return 0; }

Harobot 示例(UART)单独控制某一轴(params.set_a= false **后对用的值将不再起作用)**	#include "serial/serial.h"
	#include "wlkataC.h"
	int main() {
	serial::Serial my_serial("com4", 115200);
	Harobot_UART Harobot;
	Harobot.init(&my_serial, -1, true, true);
	Harobot.homing();
	for (int i = 0; i < 5; i++) {
	Harobot_UART::AngleParams params;
	params.x = 0;
	params.y = 20;
	params.z = 0;
	params.a = 20;
	params.b = 0;
	params.c = 0;
	params.set_x = params.set_y = params.set_z = true;
	params.set_a = params.set_b = params.set_c = false;
	Harobot.writeangle(0, params);
	Harobot_UART::AngleParams params2;
	params2.x = 0;
	params2.y = -20;
	params2.z = 0;
	params2.a = 20;
	params2.b = 20;
	params2.c = 20;
	params2.set_x = params2.set_y = params2.set_z = true;
	params2.set_a = params2.set_b = params2.set_c = false;

```
            Harobot.writeangle(0, params2);
        }
        my_serial.close();
        system("pause");
        return 0;
    }
```

18. 机械臂速度

函数	void speed(int num);
函数说明	机械臂速度设置。
参数说明	num：机械臂速度，取值范围 0~4000。
返回值	无

19. 机械臂拓展轴控制

函数	void writeexpand(int motion, int position, float d);
函数说明	控制机械臂拓展轴运动。
参数说明	Motion:0—快速运动； 1—直线运动。 Position：0—绝对值运动； 1—增量值运动。 d:移动多少距离。
返回值	无

20. **创建机械臂对象**(E4)

函数	E4_UART();
函数说明	创建机械臂对象。
参数说明	无

21. **笛卡尔坐标控制机械臂**(E4)

函数	void writecoordinate(int motion, int position, float x, float y, float z, float a);
函数说明	笛卡尔坐标控制机械臂。
参数说明	Motion:0—快速运动； 1—直线运动； 2—门型运动。 Position：0—绝对值运动； 1—增量值运动。 x：x 点坐标； y：y 点坐标； z：z 点坐标； a：a 点坐标；
返回值	无

示例(RS485)	#include "serial/serial.h"
	#include "wlkataC.h"
	int main() {
	serial::Serial my_serial("com4", 38400);

```
    E4_UART E4_1;

    E4_1.init(&my_serial, 1, true,true);

    E4_1.homing();

    for (int i = 0; i < 5; i++) {

        E4_1.writecoordinate(0, 0, 230, 0, 255.5, 0);

        E4_1.writecoordinate(0, 0, 230, 0, 200, 0);

    }

    my_serial.close();

    system("pause");

    return 0;

}
```

示例(UART)

```
#include "serial/serial.h"

#include "wlkataC.h"

int main() {

    serial::Serial my_serial("com4", 115200);

    E4_UART E4_1;

    E4_1.init(&my_serial, -1, true,true);

    E4_1.homing();

    for (int i = 0; i < 5; i++) {

        E4_1.writecoordinate(0, 0, 230, 0, 255.5, 0);

        E4_1.writecoordinate(0, 0, 230, 0, 200, 0);

    }

    my_serial.close();

    system("pause");

    return 0;

}
```

示例 (UART) 单独控制某一轴 (params.set_a= false **后对用的值将不再起作用)**

```
#include "serial/serial.h"
#include "wlkataC.h"
int main() {
    serial::Serial my_serial("com4", 115200);
    E4_UART E4_1;
    E4_1.init(&my_serial, -1, true,true);
    E4_1.homing();
    for (int i = 0; i < 5; i++) {
        E4_UART::CoordinateParams params;
        params.x = 230;
        params.y = 0;
        params.z = 255.5;
        params.a = 20;

        params.set_x = params.set_y = params.set_z = true;
        params.set_a =false;
        E4_1.writecoordinate(0, 0, params);

        E4_UART::CoordinateParams params2;
        params2.x = 230;
        params2.y = 0;
        params2.z = 200;
        params2.a = 40;
        params2.set_x = params2.set_y = params2.set_z = true;
        params2.set_a = false;
        E4_1.writecoordinate(0, 0, params2);
    }
```

	my_serial.close();
	system("pause");
	return 0;
	}

22. 角度控制机械臂(E4)

函数	void writeangle(int position, float x, float y, float z, float a);
函数说明	笛卡尔坐标控制机械臂。
参数说明	Position：0—绝对值运动； 1—增量值运动。 x：1 轴角度； y：2 轴角度； z：3 轴角度； a：4 轴角度；
返回值	无

示例(RS485)	#include "serial/serial.h"
	#include "wlkataC.h"
	int main()
	{
	serial::Serial my_serial("com4", 38400);
	E4_UART E4_1;
	E4_1.init(&my_serial, 1, true, true);
	E4_1.homing();
	for (int i = 0; i < 5; i++)
	{

	E4_1.writeangle(0, 10, 0, 0, 0);
	E4_1.writeangle(0, -10, 0, 0, 0);
	}
	my_serial.close();
	system("pause");
	return 0;
	}

示例(UART)	#include "serial/serial.h"
	#include "wlkataC.h"
	int main()
	{
	serial::Serial my_serial("com4", 115200);
	E4_UART E4_1;
	E4_1.init(&my_serial, -1, true, true);
	E4_1.homing();
	for (int i = 0; i < 5; i++)
	{
	E4_1.writeangle(0, 10, 0, 0, 0);
	E4_1.writeangle(0, -10, 0, 0, 0);
	}
	my_serial.close();
	system("pause");
	return 0;
	}

示例 (UART) 单独控制某一轴	#include "serial/serial.h" #include "wlkataC.h" int main() {

(params.set_a= false **后对用的值将不再起作用)**	serial::Serial my_serial("com4", 115200);
	E4_UART E4_1;
	E4_1.init(&my_serial, -1, true, true);
	E4_1.homing();
	for (int i = 0; i < 5; i++) {
	E4_UART::AngleParams params;
	params.x = 0;
	params.y = 20;
	params.z = 0;
	params.a = 20;
	params.set_x = params.set_y = params.set_z = true;
	params.set_a = false;
	E4_1.writeangle(0, params);
	E4_UART::AngleParams params2;
	params2.x = 0;
	params2.y = -20;
	params2.z = 0;
	params2.a = 20;
	params2.set_x = params2.set_y = params2.set_z = true;
	params2.set_a = false;
	E4_1.writeangle(0, params2);
	}
	my_serial.close();
	system("pause");
	return 0;
	}

23. **创建机械臂对象**(MT4)

函数	MT4_UART();
函数说明	创建机械臂对象。
参数说明	无

24. **笛卡尔坐标控制机械臂**(MT4)

函数	void writecoordinate(int motion, int position, float x, float y, float z, float a);
函数说明	笛卡尔坐标控制机械臂。
参数说明	Motion:0—快速运动； 1—直线运动； 2—门型运动。 Position：0—绝对值运动； 1—增量值运动。 x：x 点坐标； y：y 点坐标； z：z 点坐标； a：a 点坐标；
返回值	无

示例(RS485)	#include "serial/serial.h"
	#include "wlkataC.h"
	int main() {
	serial::Serial my_serial("com4", 38400);

```
    MT4_UART MT4_1;

    MT4_1.init(&my_serial, 1, true,true);

    MT4_1.homing();

    for (int i = 0; i < 5; i++) {

        MT4_1.writecoordinate(0, 0, 230, 0, 255.5, 0);

        MT4_1.writecoordinate(0, 0, 230, 0, 200, 0);

    }

    my_serial.close();

    system("pause");

    return 0;

}
```

示例(UART)

```
#include "serial/serial.h"

#include "wlkataC.h"

int main() {

    serial::Serial my_serial("com4", 115200);

    MT4_UART MT4_1;

    MT4_1.init(&my_serial, -1, true,true);

    MT4_1.homing();

    for (int i = 0; i < 5; i++) {

        MT4_1.writecoordinate(0, 0, 230, 0, 255.5, 0);

        MT4_1.writecoordinate(0, 0, 230, 0, 200, 0);

    }

    my_serial.close();

    system("pause");

    return 0;

}
```

示例 (UART) 单独控制某一轴 (params.set_a= false **后对用的值将不再起作用)**

```
#include "serial/serial.h"
#include "wlkataC.h"
int main() {
    serial::Serial my_serial("com4", 115200);
    MT4_UART MT4_1;
    MT4_1.init(&my_serial, -1, true,true);
    MT4_1.homing();
    for (int i = 0; i < 5; i++) {
        MT4_UART::CoordinateParams params;
        params.x = 230;
        params.y = 0;
        params.z = 255.5;
        params.a = 20;

        params.set_x = params.set_y = params.set_z = true;
        params.set_a =false;
        MT4_1.writecoordinate(0, 0, params);

        MT4_UART::CoordinateParams params2;
        params2.x = 230;
        params2.y = 0;
        params2.z = 200;
        params2.a = 40;
        params2.set_x = params2.set_y = params2.set_z = true;
        params2.set_a = false;
        MT4_1.writecoordinate(0, 0, params2);
    }
```

	my_serial.close();
	system("pause");
	return 0;
	}

25. 角度控制机械臂(MT4)

函数	void writeangle(int position, float x, float y, float z, float a);
函数说明	笛卡尔坐标控制机械臂。
参数说明	Position：0—绝对值运动； 1—增量值运动。 x：1 轴角度； y：2 轴角度； z：3 轴角度； a：4 轴角度；
返回值	无

示例(RS485)	#include "serial/serial.h"
	#include "wlkataC.h"
	int main()
	{
	serial::Serial my_serial("com4", 38400);
	MT4_UART MT4_1;
	MT4_1.init(&my_serial, 1, true, true);
	MT4_1.homing();
	for (int i = 0; i < 5; i++)
	{

	MT4_1.writeangle(0, 10, 0, 0, 0);
	MT4_1.writeangle(0, -10, 0, 0, 0);
	}
	my_serial.close();
	system("pause");
	return 0;
	}

示例(UART)	#include "serial/serial.h"
	#include "wlkataC.h"
	int main()
	{
	serial::Serial my_serial("com4", 115200);
	MT4_UART MT4_1;
	MT4_1.init(&my_serial, -1, true, true);
	MT4_1.homing();
	for (int i = 0; i < 5; i++)
	{
	MT4_1.writeangle(0, 10, 0, 0, 0);
	MT4_1.writeangle(0, -10, 0, 0, 0);
	}
	my_serial.close();
	system("pause");
	return 0;
	}

示例 (UART) 单独控制某一轴	#include "serial/serial.h" #include "wlkataC.h" int main() {

(params.set_a= false 后对用的值将不再起作用)	serial::Serial my_serial("com4", 115200);
	MT4_UART MT4_1;
	MT4_1.init(&my_serial, -1, true, true);
	MT4_1.homing();
	for (int i = 0; i < 5; i++) {
	MT4_UART::AngleParams params;
	params.x = 0;
	params.y = 20;
	params.z = 0;
	params.a = 20;
	params.set_x = params.set_y = params.set_z = true;
	params.set_a = false;
	MT4_1.writeangle(0, params);
	MT4_UART::AngleParams params2;
	params2.x = 0;
	params2.y = -20;
	params2.z = 0;
	params2.a = 20;
	params2.set_x = params2.set_y = params2.set_z = true;
	params2.set_a = false;
	MT4_1.writeangle(0, params2);
	}
	my_serial.close();
	system("pause");
	return 0;
	}

26. 创建步进电机控制器对象（MS4220）

函数	MS4220_UART();
函数说明	创建步进电机控制器对象。
参数说明	无

27. 设置地址（MS4220）

函数	void init(serial::Serial *pserial, int addr, bool block_flag = false, bool message_flag = false);
函数说明	将串口和步进电机控制器地址进行绑定。
参数说明	*pserial:串口对象； adr:多功能控制器地址，RS485 模式使用实际地址，UART 模式使用-1； block_flag：函数是否为阻塞模式； message_flag：是否终端打印给机械臂发送和接收的数据；

28. 重启（MS4220）

函数	void restart();
函数说明	步进电机控制器重启
参数说明	无
示例（RS485）	#include "serial/serial.h" #include "wlkataC.h" int main() {

	serial::Serial my_serial("com13", 38400);
	MS4220_UART MS4220_1;
	MS4220_1.init(&my_serial, 10， True,True);
	// MS4220_1.message_print(true);
	MS4220_1.restart();
	my_serial.close();
	system("pause");
	return 0;
	}

29. 回零（MS4220）

函数	void homing();
函数说明	机械臂回零操作。需要注意的是回零功能需要安装回零传感器。
参数说明	无

示例(RS485)	#include "serial/serial.h"
	#include "wlkataC.h"
	int main() {
	serial::Serial my_serial("com13", 38400);
	MS4220_UART MS4220_1;
	MS4220_1.init(&my_serial, 10， True,True);
	// MS4220_1.message_print(true);
	MS4220_1.homing();
	my_serial.close();
	system("pause");
	return 0;
	}

30. 移动（位置模式）（MS4220）

函数	void move_position(int position, int num, int speed = 50);
函数说明	控制步进电机转动指定的脉冲数
参数说明	position:0 为绝对位置模式，1 为相对位置模式。 num:转动的脉冲数； speed:可选参数，转动的速度，取值范围是 0——100;

示例(RS485)	#include "serial/serial.h"
	#include "wlkataC.h"
	int main() {
	serial::Serial my_serial("com13", 38400);
	MS4220_UART MS4220_1;
	MS4220_1.init(&my_serial, 10， True,True);
	// MS4220_1.message_print(true);
	MS4220_1.homing();
	MS4220_1.move_position(0,2000);
	my_serial.close();
	system("pause");
	return 0;
	}

31. 移动（速度模式）（MS4220）

函数	void move_speed(int num);
函数说明	控制步进电机以指定的速度转动；
参数说明	Num:转动的速度，取值范围是-100-100，表示正传反转。

示例(RS485)	#include "serial/serial.h"
	#include "wlkataC.h"
	int main() {
	serial::Serial my_serial("com13", 38400);
	MS4220_UART MS4220_1;
	MS4220_1.init(&my_serial, 10， True,True);
	// MS4220_1.message_print(true);
	MS4220_1.homing();
	MS4220_1.move_speed(100);
	std::this_thread::sleep_for(std::chrono::milliseconds(2000));
	MS4220_1.move_speed(-100);
	std::this_thread::sleep_for(std::chrono::milliseconds(2000));
	my_serial.close();
	system("pause");
	return 0;
	}

32. 获取步进电机控制器状态（MS4220）

函数	string get_State()
函数说明	获取步进电机控制器状态
参数说明	无

示例(RS485)	#include "serial/serial.h"
	#include "wlkataC.h"
	int main()
	{
	serial::Serial my_serial("com13", 38400);
	MS4220_UART MS4220_1;

	MS4220_1.init(&my_serial, 10, true, true);
	std::cout << MS4220_1.get_State() << std::endl;
	my_serial.close();
	system("pause");
	return 0;
	}

33. 获取步进电机当前位置（MS4220）

函数	int get_position();
函数说明	获取步进电机当前位置
参数说明	无

示例(RS485)	#include "serial/serial.h"
	#include "wlkataC.h"
	int main()
	{
	serial::Serial my_serial("com5", 38400);
	MS4220_UART MS4220_1;
	MS4220_1.init(&my_serial, 10, true, true);
	std::cout << MS4220_1.get_position() << std::endl;
	my_serial.close();
	system("pause");
	return 0;
	}

34. 获取传感器的状态（MS4220）

函数	int get_sensor(string name);

函数说明	获取传感器的状态
参数说明	name:可取值为 S0,S1,S2 对应外接的 3 个传感器

示例(RS485)	#include "serial/serial.h"
	#include "wlkataC.h"
	int main()
	{
	serial::Serial my_serial("com5", 38400);
	MS4220_UART MS4220_1;
	MS4220_1.init(&my_serial, 10, true, true);
	std::cout << MS4220_1.get_sensor("S0") << std::endl;
	std::cout << MS4220_1.get_sensor("S1") << std::endl;
	std::cout << MS4220_1.get_sensor("S2") << std::endl;
	my_serial.close();
	system("pause");
	return 0;
	}

35. **创建** Mirobot_Serial_GUI **对象**

函数	Mirobot_Serial_GUI();
函数说明	创建 Mirobot_Serial_GUI 对象
参数说明	无
返回值	无

36. **调取** Mirobot_GUI

函数	Mirobot_Serial_GUI();

函数说明	创建 Mirobot_Serial_GUI 对象
参数说明	无
返回值	无
示例	import wlkatapython Mirobot_GUI=wlkatapython.Mirobot_Serial_GUI() Mirobot_GUI.Mirobot_GUI()

版权所有© 勤牛创智科技有限公司 2025。保留一切权利。

非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部， 并不得以任何形式传播。

免责申明

在法律允许的最大范围内，本手册所描述的产品（含其硬件、软件、固件等）均“按照现状”提供，可能存在瑕疵、错误或故障，勤牛创智不提供任何形式的明示或默示保证，亦不对使用本手册或使用本公司产品导致的任何特殊、偶然或间接的损害进行赔偿。在使用本产品前详细阅读本使用手册及网上发布的相关技术文档并了解相关信息，确保在充分了解产品相关知识的前提下使用本产品。

本产品的使用者有责任确保遵循相关国家的切实可行的法律法规，确保在勤牛创智机械臂的使用中不存在任何重大危险。

版本修订说明

时间	版本号	修订记录
2025.10.31	V0.1.0 版	初稿

北京勤牛创智科技有限公司

网址：cn.wlkata.com

官方公众号