AUBO SDK 使用手册

Version 1.0.0

目录

1 AUBO SDK 简介

AUBO SDK 是一款软件开发工具包（Software Development Kit），由遨博（北京）智能科技股份有限公司开发，提供了丰富的 API 接口，用于编写和部署遨博机器人应用程序，旨在帮助开发人员快速灵活地构建自己的应用程序来控制和使用遨博机械臂，以满足在不同应用场景下的需求。

AUBO SDK 支持多种编程语言，包括 C++、Python、Lua 等，兼容 Windows 和 Linux 操作系统，方便开发者根据自身需求进行选择。开发人员可以通过 SDK 开发自己的应用程序，从而实现与遨博机器人的交互及通信。

此外，AUBO SDK 还提供了丰富的开发文档和代码示例，使开发人员能更快地入门和开发他们的机器人应用程序。

1.1 SDK 功能模块

RPC、RTDE 和 SCRIPT 是 AUBO SDK 三大重要的功能模块，它们之间既相互独立，但又可以互相配合使用，通过结合使用可以实现更加灵活和高效地机器人控制方案。

1.1.1 RPC 模块

SDK 中的 RPC 模块，基于 TCP Socket 在网络中进行通信，使用 JSON-RPC2.0 协议进行数据交换，支持客户端/服务器模式和请求/响应模式，具有简单易用、轻量、跨平台、可扩展等特点，支持多种编程语言和操作系统。通信端口号为 30004。

RPC 模块支持多种类型的指令，如运动指令、状态指令和 IO 指令等，可以满足不同应用场景的需求。外部设备（如计算机）可以通过 SDK 中的 RPC 模块向遨博控制器发送指令请求，遨博控制器可以根据指令内容进行相应的动作，如运动、暂停、停止等，并将执行结果返回给外部设备（如计算机）。

同时，RPC 模块还可以与 RTDE 模块或 SCRIPT 模块配合使用，实现更为复杂的机器人应用程序。例如，用户可以通过 RPC 模块来调用运动指令接口（如 moveJoint、moveLine 等），同时通过 RTDE 模块来订阅获得机器人的实时状态，如实际的位置姿态等。

1.1.2 RTDE 模块

RTDE（Real-Time Data Exchange）是一种实时数据交换协议，用以实现外部设备（如计算机）与机器人控制器之间的实时数据传输，比如机器人的关节角度、速度、加速度等状态信息。AUBO SDK 中的 RTDE 模块，就是基于 RTDE 协议来实现控制器与外部设备之间的实时数据交换。另外，RTDE 模块还可以与 RPC 模块或 SCRIPT 模块同时使用，不会破坏系统的实时性。

SDK 中的 RTDE 模块，基于 TCP Socket 在网络中进行通信，可以将遨博控制器中的数据传输到外部设备（如计算机），并且可以将外部设备中的数据传输到遨博控制器中。外部设备可以通过 RTDE 模块获取机器人的关节角度、速度、加速度等实时数据，也可以通过 RTDE 模块向控制器的寄存器中写入数据。通信端口号为 30010。

RTDE 菜单是一种用于配置 RTDE 数据流的文件格式。它定义了数据的名称、类型等信息，例如机器人的位置、关节角度、速度、电流等信息，支持多种数据类型，包括布尔型、整型、浮点型等。如想查看定义，请阅读附录中的 RTDE 菜单。

1.1.3 SCRIPT 模块

SDK 中的 SCRIPT 模块，基于 TCP Socket 在网络中进行通信，支持发送本地的脚本程序到机器人控制器，然后控制器会加载执行脚本程序。通信端口号为 30002。

SCRIPT 模块支持 Lua 脚本，包括基本的语句、控制结构、函数定义等。它提供了丰富的机器人控制 API，如运动控制、算法、获取机器人状态等。另外，SCRIPT 模块提供了脚本程序调试和处理错误的功能，例如设置断点等。

另外，SCRIPT 模块还可以与 RTDE 模块或 RPC 模块配合使用。例如，用户可以在本地编写脚本程序，并通过 SCRIPT 模块将本地的脚本程序发送到机器人控制器进行执行，同时调用 RTDE 模块来实时监测机器人的状态。

1.2 JSON-RPC 协议说明

1.2.1 简介

JSON-RPC（JavaScript Object Notation Remote Procedure Call）是一种远程过程调用协议。它使用 JSON 作为数据格式，可以通过 TCP Socket、HTTP、WebSocket 等传输协议来进行客户端和服务器之间的通信。

JSON-RPC 具有以下特性：

- 简易性：以 JSON 作为数据交换格式，简洁、易理解和易解析。
- 轻量级：使用基于文本的 JSON 格式进行数据交换，数据和协议结构简单，网络传输数据量小、速度快、性能高。
- 跨语言和跨平台：JSON-RPC 支持多种编程语言（如 C++、Python 等）和不同平台之间的通信
- 支持多种传输协议：JSON-RPC 可以使用多种传输协议进行数据传输，如 HTTP、TCP Socket、WebSocket 等，适用于多种网络环境。
- 错误处理：JSON-RPC 定义了错误对象和错误码的规范，客户端可以根据服务器返回的错误信息来调试和处理问题。

JSON-RPC 采用请求-响应的方式来实现客户端和服务器之间的远程过程调用。客户端发送 JSON 格式请求给服务器，例如 {”jsonrpc”:”2.0”,”method”:”getRobotNames”,”params”:[],”id”:1}。服务器接收到请求后，解析 JSON 对象，提取 “method”、“params” 和” id “等字段的值。服务器执行与”method” 对应的方法，并使用提供的参数。根据执行结果，服务器会生成一个响应对象并将响应对象序列化成 JSON 格式，例如 {”id”:1,”jsonrpc”:”2.0”,”result”:[”rob1”]}，然后将 JSON 格式的响应发送给客户端。最后，客户端接收到服务器发送的响应。

1.2.2 支持的传输协议与端口号

AUBO 中的 RPC 支持 TCP、HTTP 和 WebSocket 传输协议，其对应的端口号如下：

协议	端口号
TCP	30004
HTTP	9012
WebSocket	9012

1.2.3 JSON-RPC 2.0 规范

请求对象

RPC 请求对象包含以下成员：

- jsonrpc：表示 JSON-RPC 协议的版本号，必须是 2.0
- method：表示被调用的方法名称，是一个字符串
- params：表示参数值，是一个结构化的值
- id：标识符

响应对象

这里有 RPC 调用成功和 RPC 调用失败两种情况：

1. 当 RPC 调用成功时，RPC 响应对象包含以下成员：

   - jsonrpc：表示 JSON-RPC 协议的版本号，必须是 2.0
   - result：表示函数被执行的返回值
   - id：表示标识符，与请求对象的标识符一定相同

2. 当 RPC 调用出现错误时，RPC 响应对象包含以下成员：

   - jsonrpc：表示 JSON-RPC 协议的版本号，必须是 2.0
   - error：表示错误对象，错误对象包含以下成员

     code 表示错误码

     message 表示简短的错误描述

     data 表示错误的附加信息，可能被忽略

   - id：表示标识符，与请求对象的标识符一定相同

错误对象的错误码如下表所示：

错误码	消息	说明
-32700	Parse error.	服务器接收到无效的 JSON。 在服务器解析 JSON 文本时发生错误。
-32600	Invalid Request.	发送的 JSON 不是有效的请求对象。
-32601	Method not found.	该方法不存在/不可用。
-32602	Invalid params.	无效的方法参数。
-32603	Internal error.	JSON-RPC 内部错误。
-32099..-32000	Server error.	保留用于实现定义的服务器错误。

1.2.4 示例

用 Python 编写的 WebSocket 协议传输示例如下：
localhost 是机器人 IP 地址。

```python
import json
import websocket

# WebSocket 服务器的地址
server_address = "ws://localhost:9012"

# 创建 WebSocket 连接
ws = websocket.create_connection(server_address)

# 构造 JSON-RPC 请求
request = {
    "jsonrpc": "2.0",
    "method": "getRobotNames",
    "params": [],
    "id": 1
}

# 将请求转换为 JSON 字符串
request_json = json.dumps(request)

# 发送请求
ws.send(request_json)

# 接收响应
response_json = ws.recv()

# 解析响应
response = json.loads(response_json)

print("Received Response:", response)

# 关闭 WebSocket 连接
ws.close()
```

程序运行结果：

```
Received Response: {'id': 1, 'jsonrpc': '2.0', 'result': ['rob1']}
```

1.3 RTDE 协议说明

1.3.1 简介

RTDE（Real-Time Data Exchange）协议是机器人控制器与外部设备进行实时数据交换的协议。通过 RTDE 协议，外部设备（如计算机）可以获取机器人的状态信息，例如关节角

大小、末端工具的位置姿态、运动的速度和加速度等。此外，外部设备还可以通过 RTDE 协议向机器人发送控制命令，例如通过写入寄存器来使机器人运动到指定的位置。

RTDE 协议具有以下特性：

- 实时性：机器人控制器与外部设备通过 RTDE 协议进行数据交换的时候，数据传输快，不会破坏机器人控制器的实时性。
- 双向通信：RTDE 协议支持机器人控制器与外部设备进行双向数据交换，即外部设备既可以接收机器人数据，也可以发送指令。
- 跨语言和跨平台：RTDE 协议支持多种编程语言（如 C++、Python 等）和不同平台之间的通信。
- 支持多种传输协议：RTDE 协议可以使用多种传输协议进行数据传输，如 HTTP、TCP、WebSocket 等，适用于多种网络环境。

1.3.2 支持的传输协议与端口号

AUBO 中的 RTDE 支持 TCP、HTTP 和 WebSocket 传输协议，其对应的端口号如下：

协议	端口号
TCP	30010
HTTP	9013
WebSocket	9013

1.3.3 RTDE 规范

RTDE 菜单

RTDE 菜单是一个定义了 RTDE 数据同步包的内容和格式的菜单，它是一个规范。它描述了可同步的输入和输出字段以及它们的数据类型。RTDE 菜单包括了输入菜单和输出菜单，输入菜单为客户端向服务端推送数据的菜单，输出菜单为服务端向客户端推送数据的菜单。如想知道 RTDE 菜单中的字段和数据类型列表，可查看附录中的 RTDE 菜单。

请求-响应

客户端可以向服务器订阅话题、发布话题和取消话题。

订阅话题

当客户端订阅话题的时候，客户端会向服务器发送数据包。数据包的第 1 个元素为索引值，固定为 100，第二个元素为对 `RtdeRecipe` 结构体。`RtdeRecipe` 结构体的定义如下：

```
struct RtdeRecipe
{
    bool to_server;          // 输入/输出，
                             // to_server=false 表示订阅来自服务器的数据，
                             // to_server=true 表示向服务器推送数据
    int chanel;                         // 通道  通道数仅支持 0-99
    double frequency;                   // 更新频率
    int trigger;                        // 触发方式：0 - 周期；1 - 变化
    std::vector<std::string> segments; // 字段列表，表示订阅/发布的话题
};
```

其中，to_server 的值固定为 false，segments 为 RTDE 菜单中的输出菜单的成员。

之后，客户端会接收到服务器响应的数据包。数据包的第一个元素是通道，和订阅时的通道相同。第二个以及往后的元素是订阅话题的数据。

例如，想在 0 号通道订阅话题 R1_actual_q，那么客户端向服务器发送的数据包为：

```
[100,{"chanel":0,"frequency":50.0,"segments":["R1_actual_q"],
"to_server":false,"trigger":0}]
```

客户端接收到的服务器响应如下：

```
[0,[-0.027531569059195515,-0.28143723119618147,1.6961810563264004,
0.4177235477634482,1.5627876064666573,-0.03521353669519756]]
```

上面示例中,数据包中的第一个元素表示 0 号通道,第二个元素是订阅话题 R1_actual_q 的值，即实际六个关节角大小。

取消话题

订阅话题之后，客户端会不断地接收到从服务器发送的数据。如果想取消掉推送，则可以取消话题。取消话题与订阅话题发送的数据包格式相同，但要 frequency 设置为 0，chanel 设置为要取消的话题通道，segments 设置为空。

例如，取消 0 号通道的话题：

```
[100,{"chanel":0,"frequency":0.0,"segments":[],"to_server":false,"trigger":0}]
```

发布话题

当客户端发布话题的时候，客户端会向服务器发送数据包。数据包的第 1 个元素为索引值，固定为 100，第二个元素是 RtdeRecipe 结构体。RtdeRecipe 结构体的定义如下：

```
struct RtdeRecipe
{
    bool to_server;          // 输入/输出,
                             // to_server=false 表示订阅来自服务器的数据,
                             // to_server=true 表示向服务器推送数据
    int chanel;                          // 通道  通道数仅支持 0-99
    double frequency;                    // 更新频率
    int trigger;                         // 触发方式: 0 - 周期; 1 - 变化
    std::vector<std::string> segments; // 字段列表，表示订阅/发布的话题
};
```

其中，to_server 的值固定为 true，segments 为 RTDE 菜单中的输入菜单的成员。

然后，客户端再向服务器推送数据，数据包的第一个元素是通道，和发布话题时的通道相同。第二个以及往后的元素是发布话题的数据。

例如,想在 1 号通道发布话题 input_float_registers_0 和 input_double_registers_1，那么客户端向服务器发送的数据包为：

```
[100,{"chanel":1,"frequency":1.0,
"segments":["input_float_registers_0","input_double_registers_1"],
"to_server":true,"trigger":0}]
```

然后，再从客户端向服务器推送数据：

```
[1,3.1,4.1]
```

上面示例中,数据包中的第一个元素表示 1 号通道,第二个元素是 input_float_registers_0 的值，第三个元素是 input_double_registers_1 的值。

如想判断数据是否被成功写入寄存器,可订阅话题 input_float_registers_r0 和 input_double_registers_r1：

```
[100,{"chanel":1,"frequency":1.0,
"segments":["input_float_registers_r0","input_double_registers_r1"],
"to_server":false,"trigger":0}]
```

如果成功，那么客户端接收到的服务器响应为：

```
[1,3.0999999046325684,4.1]
```

1.3.4 示例

用 Python 编写的 WebSocket 协议传输示例如下：
localhost 是机器人 IP 地址。

```python
import asyncio
import websockets


async def rtde_communication():
    uri = "ws://localhost:9013"
    async with websockets.connect(uri) as websocket:
        # 发布话题
        publish_topic = '[100,{"chanel":1,"frequency":1.0,
                        "segments":["input_float_registers_0",
                        "input_double_registers_1"],"to_server":true,
                        "trigger":0}] '
        await websocket.send(publish_topic)

        # 向服务器推送数据
        publish_data = '[1,3.3,4.4]'
        await websocket.send(publish_data)

        # 订阅话题
        subscribe_topic = '[100,{"chanel":1,"frequency":50.0,
```

```
                        "segments":["input_float_registers_r0",
                        "input_double_registers_r1"],"to_server":false,
                        "trigger":0}] '
        await websocket.send(subscribe_topic)

        # 接收 RTDE 数据包
        while True:
            data_packet = await websocket.recv()
            print("Received data packet:", data_packet)

asyncio.get_event_loop().run_until_complete(rtde_communication())
```

程序运行结果：

```
Received data packet: [1,3.299999952316284,4.4]
Received data packet: [1,3.299999952316284,4.4]
Received data packet: [1,3.299999952316284,4.4]
Received data packet: [1,3.299999952316284,4.4]
Received data packet: [1,3.299999952316284,4.4]
Received data packet: [1,3.299999952316284,4.4]
Received data packet: [1,3.299999952316284,4.4]
```

2 AUBO SDK 开发包

2.1 简介

AUBO SDK 开发包提供了头文件、库文件以及丰富的代码示例等，可以帮助开发者快速地开发应用程序。由遨博提供的 SDK 开发包中有以下几个文件夹：

- include 文件夹：头文件，包含了提供给用户使用的 API 接口、数据类型和宏定义。
- lib 文件夹：编译时使用的库文件。
- share 文件夹：代码示例以及错误码说明。

2.2 如何使用

在下载使用 SDK 开发包之前，需要先确认 SDK 的版本与 Server 的版本是否兼容和匹配。

要使用 SDK 开发包，一般需要以下几个步骤：

1. 配置开发环境
2. 下载并安装 SDK 开发包
3. 在 IDE 中新建一个项目
4. 将 AUBO SDK 开发包中提供的库文件和头文件添加到项目中
5. 编写代码
6. 连接遨博机器人或者Aubo Sim 虚拟机
7. 编译运行代码

具体使用步骤可能会因具体项目而异，请根据实际情况进行调整。

2.3 运行环境

2.3.1 C++ SDK

- 支持的操作系统：Linux 64 位/Windows 64 位
- 编译器要求：
- Linux：G++-7/GCC-7
- Windows：MSVC 2019 及以上

2.3.2 Python SDK

- 支持的操作系统：Linux 64 位/Windows 64 位
- Python 版本要求：
- Linux：Python 3.6 ~ Python 3.11
- Windows：Python 3.6 ~ Python 3.11

2.4 下载与安装

2.4.1 SDK 开发包

SDK 开发包的下载链接：https://download.aubo-robotics.cn/sdk/。

此链接中的 SDK 开发包中有 C++ 头文件、C++ SDK 库文件和 C++/Python demo 示例等，可用于 C++ SDK 二次开发。如需使用 Python SDK，则需要下载和安装 pyaubo_sdk 库文件。

2.4.2 pyaubo_sdk 库

下载和安装

- 方法 1：在 pypi 官网上搜索 pyaubo_sdk
  - 打开 pypi 网址：https://pypi.org/
  - 输入并搜索 pyaubo_sdk

图 1

  - 下载 pyaubo_sdk

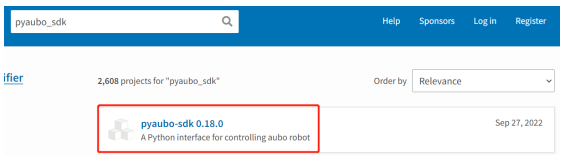

图 2

- 方法 2：通过 pip 命令来下载和安装
  - 安装命令

```
pip3 install pyaubo_sdk
```

  - 安装指定版本

```
pip3 install pyaubo_sdk==x.x.x
```

例如，下载和安装版本 0.19.6

```
pip3 install pyaubo_sdk==0.19.6
```

- 方法 3：通过 PyCharm 来下载和安装
  - File -> Settings

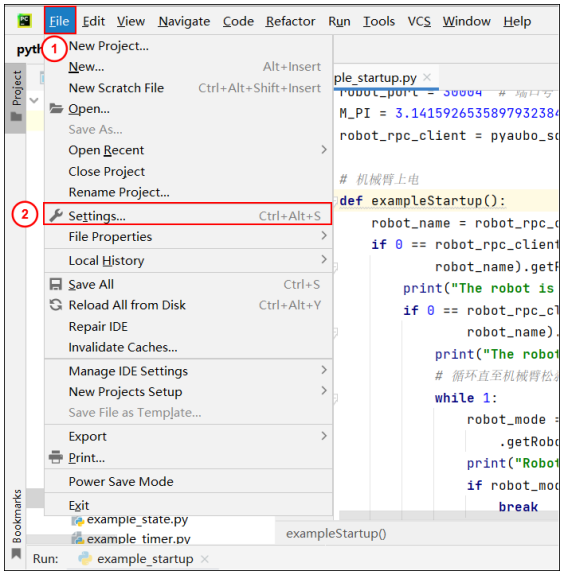

图 3

– 在 Python Interpreter 页面下，点击 + 标志

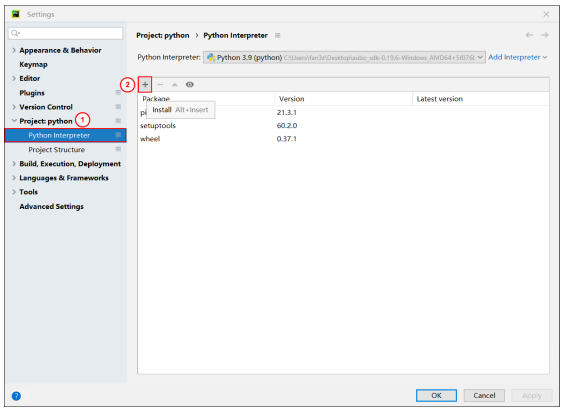

图 4

– 搜索 pyaubo-sdk，选择版本，点击 Install Package

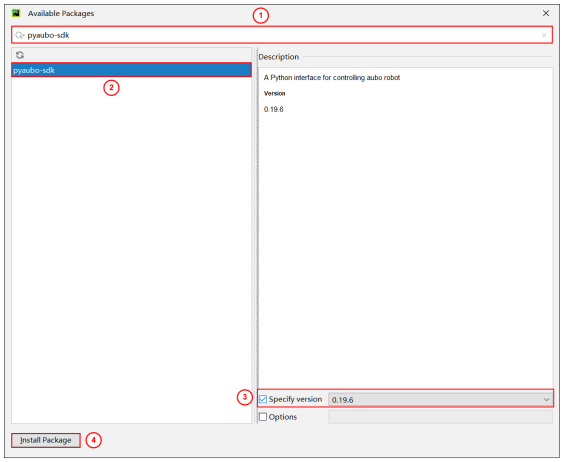

图 5

卸载

- 方法 1：通过 pip 命令来卸载

  `pip3 uninstall pyaubo_sdk`

- 方法 2：通过 PyCharm 来卸载

  – File -> Settings

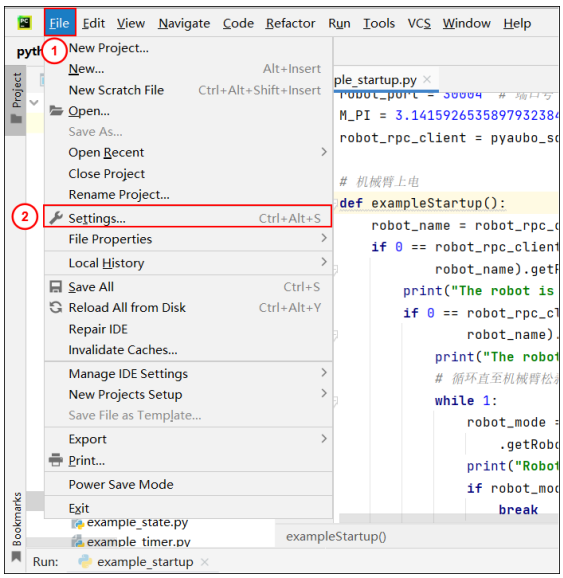

图 6

  – 在 Python Interpreter 界面中，选中 pyaubo-sdk，点击—标志来进行卸载，卸载完成后，点击 OK

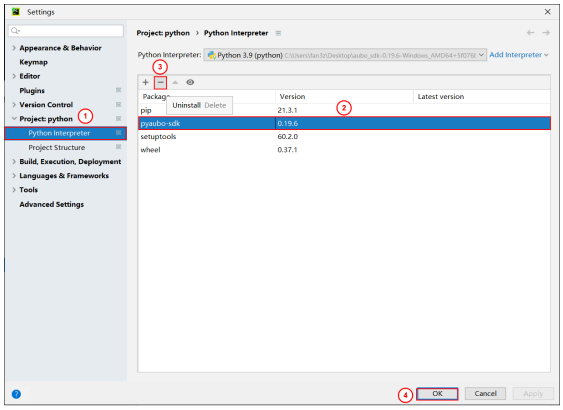

图 7

2.5 版本管理

ARCS SDK 开发包遵循语义版本控制，详细可参考 https://semver.org/lang/zh-CN/。

2.5.1 SDK 版本发布规范

标准版本

标准版本号是由主版本号（MAJOR）、次版本号（MINOR）和修订版本号（PATCH）组成的。例如，标准版本号是 0.18.0，则主版本号为 0，次版本号为 18，修订版本号为 0。

主版本号表示出现不兼容的 API 变化。

次版本号表示新增向下兼容的功能。

修订版本号表示修复向下兼容的 bug。

预发布版本

预发布版本号是在标准版本号的后面，先加上一个“-”再加上一连串以句点分隔的标识符。标识符必须由 ASCII 字母数字和连接号 [0-9A-Za-z-] 组成，且禁止留白。例如，1.0.0-alpha、1.0.0-alpha.1。

预发布版本号的优先级低于相关联的标准版本。被标上预发布版本号则表示这个版本并非稳定而且可能无法满足预期的兼容性需求。

构建元数据

构建元数据是指一些额外的信息，比如构建时间、序列号、校验值等。可以被标注在修订版或预发布版本之后，先加上一个“+”再加上一连串以句点分隔的标识符来修饰。例如，1.0.0-alpha+001、1.0.0+20130313144700、1.0.0-beta+exp.sha.5114f85。

当两个版本的版本号相同且只有版本编译信息有差别时，属于相同的优先层级。

2.5.2 SDK 版本兼容说明

SDK 升级兼容

1. PATCH 修订版本升级：需要替换二进制库，不需要替换头文件，不需要重新编译源码。例如，1.2.1 升级为 1.2.2。
2. MINOR 次版本号升级：需要替换二进制库，不需要替换头文件，不需要重新编译源码。例如，1.2.1 升级为 1.3.0。
3. MAJOR 主版本号升级：不兼容升级，用户需要修改源码，需要替换二进制库，需要替换头文件，需要重新编译源码。例如，1.2.1 升级为 2.0.0。

SDK 兼容示教器软件

1. PATCH 不同：兼容
2. MINOR 不同：

- SDK 接口 MINOR > 软件包接口 MINOR：不兼容
- SDK 接口 MINOR < 软件包接口 MINOR ： 兼容

3. MAJOR 不同：不兼容

3 快速开始

该部分介绍如何直接运行 AUBO SDK 开发包里的代码示例，包含以下内容：

- SDK PC 端与机器人通信设置：介绍了 SDK PC 端与机器人的有线通信和无线通讯这两种方式的设置
- 运行 C++ 示例：在运行 C++ 示例前，需要在本地上配置编译运行环境和安装 aubo sdk 开发包
- 运行 Python 示例：在运行 Python 示例前，需要在本地上配置编译运行环境和安装 pyaubo_sdk 库

3.1 SDK PC 端与机器人通信设置

SDK PC 端可以通过两种方式实现与机器人通信：

无线网络通讯：无线网络通讯是通过无线局域网（Wi-Fi）与机器人进行通信的方法。要使用无线网络通讯，需要配置 PC 和机器人相应的网络设置，让它们处于同一个无线局域网（Wi-Fi）下。PC 端可以通过无线网络向机器人发送指令、接收数据并与其进行交互等。

有线网络通讯：有线网络通讯是通过有线连接（以太网）与机器人进行通信的方式。要使用这种通讯方式，需要用网线连接 PC 与控制柜，并设置 IP 地址、子网掩码和网关，使它们处于同一个网段下。PC 端可以通过有线网络向机器人发送指令、接收数据并与其进行交互等。

3.1.1 无线网络通讯

可参考如下步骤，来实现机器人和 PC 的无线网络通讯：

1. 用网线将控制柜与路由器或网口连接起来，来使控制柜连接上无线网络。

2. 在示教器中将网络设置为 **DHCP**，**网络详细设置**里面的 **IP 地址**就是机器人的 IP。下图里的机器人 IP 地址是 172.30.10.100。

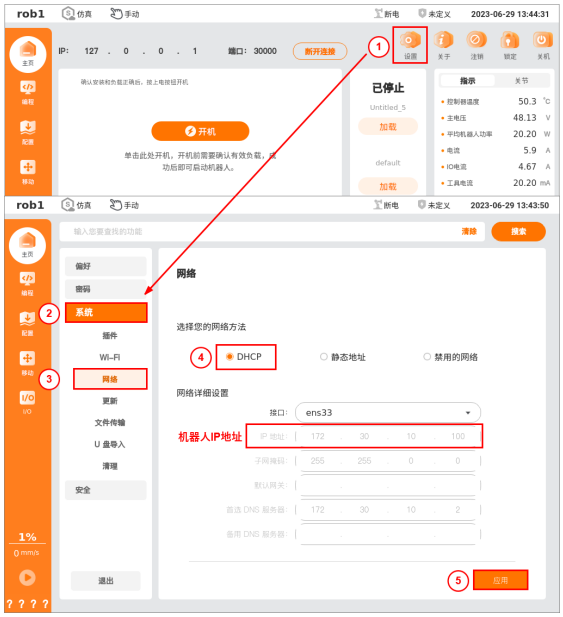

图 8

3. 在 PC 端，连接和机器人相同的 Wi-Fi 。

4. 在 PC 的终端里，测试网络的连通性。

- **Windows 操作系统**:

  – 方法 1：使用 ping 命令。打开 cmd，输入 ping 172.30.10.100。如下图所示，通信成功。

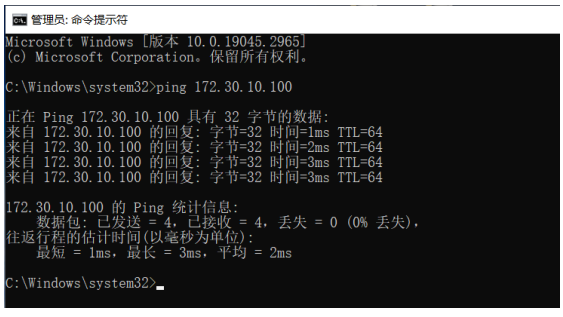

图 9

  – 方法 2：使用 telnet 命令。打开 cmd，输入 telnet 172.30.10.100 30004。如下图所示，通信成功。

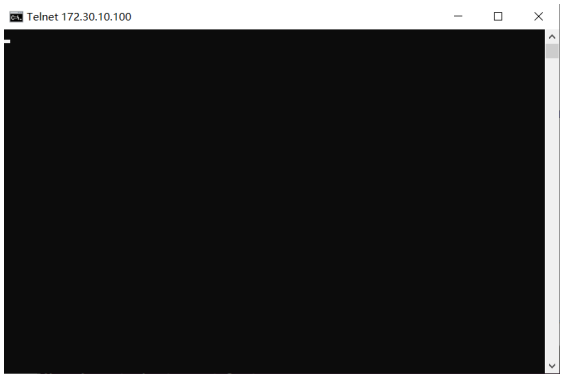

图 10

如果出现下图中的提示，则说明 telnet 没有开启。

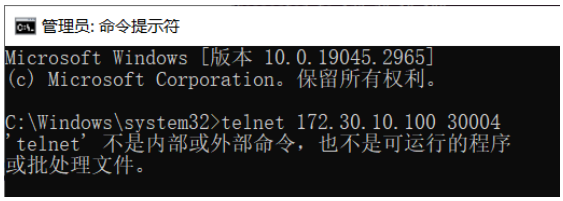

图 11

按照如下图所示操作，开启 telnet 即可。

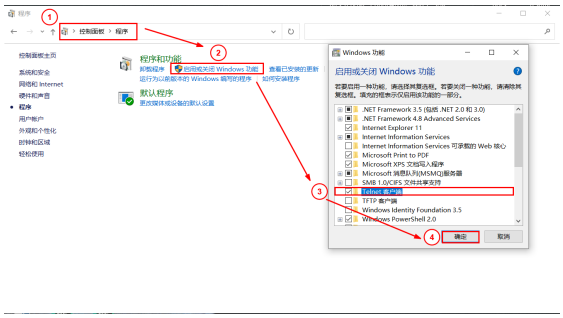

图 12

- **Linux 操作系统**:
  - 方法 1：使用 ping 命令。打开 terminal，输入 ping 172.30.10.100。如下图所示，通信成功。

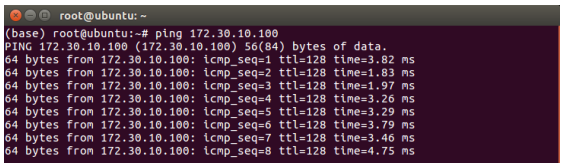

图 13

  - 方法 2：使用 telnet 命令。打开 terminal，输入 telnet 172.30.10.100 30004。如下图所示，通信成功。

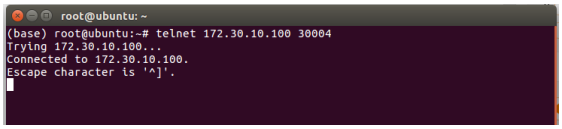

图 14

3.1.2 有线网络通讯

可参考如下步骤，来实现机器人和 PC 的有线网络通讯：

1. 用网线将控制柜与 PC 连接起来。

2. 在示教器中将网络设置为 **静态地址**、**网络详细设置**里面的 **IP 地址**就是机器人的 IP。下图里的机器人 IP 是 192.168.1.100。**子网掩码**必须设置为 255.255.255.0。**网关**的 IP 地址中的主机地址部分通常被设置为 1。例如 IP 地址是 192.168.1.100，那么它的 **默认网关**要设置为 192.168.1.1。

图 15

3. 在 PC 端，设置 **IP 地址**、**子网掩码**和**网关**，使其与机器人在同一个网段下。机器人和 PC 的 **IP 地址前缀**必须是相同的，例如机器人 IP 地址是 192.168.1.100，那么它的 **IP 地址前缀**就是 192.168.1，PC 端的 **IP 地址前缀**必须是 192.168.1。机器人和 PC 的 **子网掩码**都必须是 255.255.255.0。机器人和 PC 的 **网关**必须是相同的，网关的 IP 地址中的主机地址部分通常被设置为 1。例如 IP 地址是 192.168.1.100，那么它的网关要设置为 192.168.1.1。下面分别介绍如何在 **Windows 操作系统**和 **Linux 操作系统**中进行配置。

- **Windows 操作系统**：

  – 在 PC 里找到 **以太网** —> 选中 —> 右键 —> 点击 **属性**。

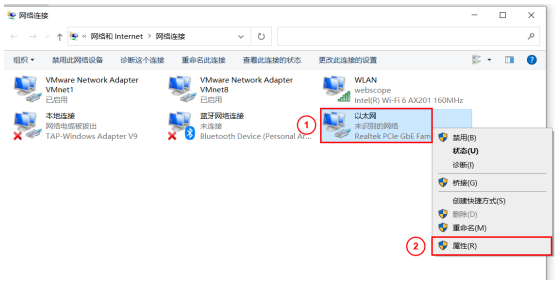

图 16

– 选中 **Internet 协议版本 4（TCP/IPv4）** —> 点击 **属性。**

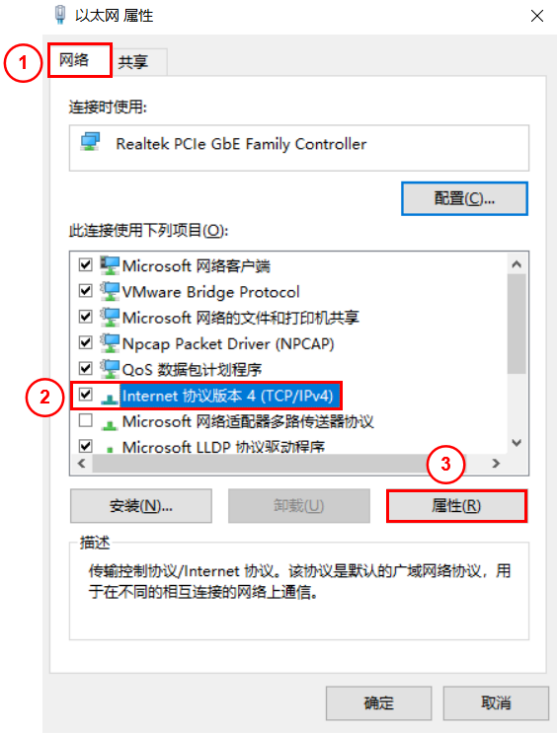

图 17

– 如下图所示，设置 **IP 地址**、**子网掩码**和**网关** —> 点击 **确定。**
图中，主机的 IP 地址为 192.168.1.55，机器人的 IP 地址为 192.168.1.100，它们处在同一个网段。

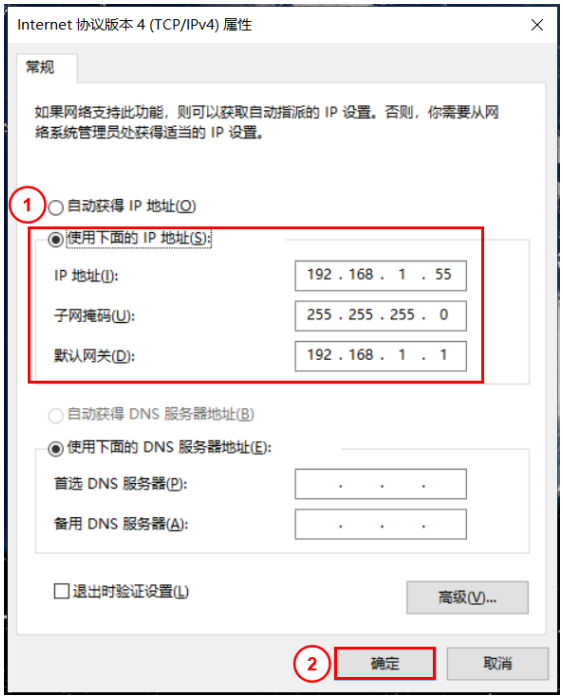

图 18

- **VMware 虚拟机**:

  – 在 **PC 的主机**里面，设置以太网的 **IP 地址**、* 子网掩码 * 和 **网关**，使主机与机器人处于同一网段下。
  – 打开 **虚拟网络编辑器**，选中 **VMnet0**，点击 **更改设置。**

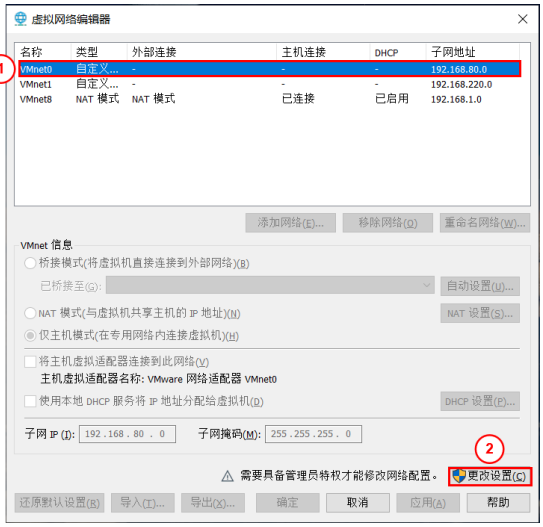

图 19

  – 选中 **VMnet0**，选择桥接至 **Realtek PCIe GbE Family Controller**，点击 **确定。**

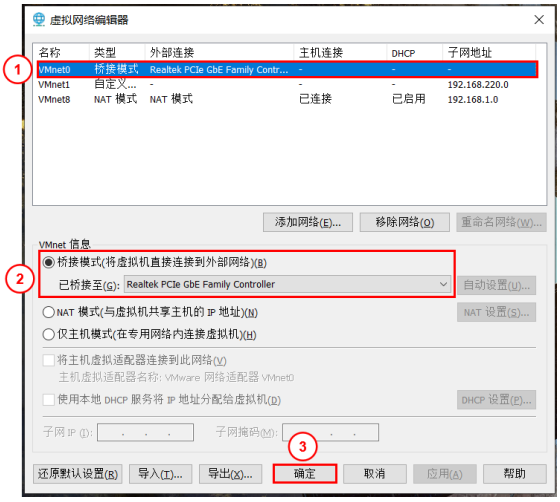

图 20

– 打开 **虚拟机设置**，设置 **网络适配器里**的 **网络连接**为 **桥接模式。**

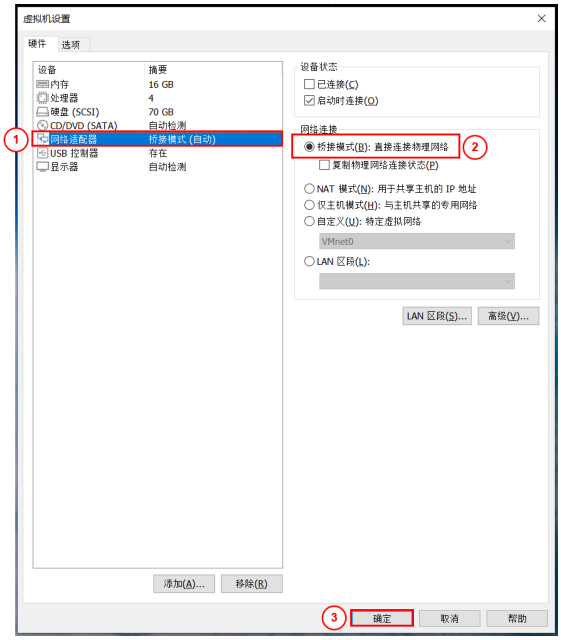

图 21

– 设置虚拟机的 IP 地址，使其与机器人在同一个网段。
下图中，虚拟机的 IP 地址是 192.168.1.33，机器人的 IP 地址是 192.168.1.100，它们处于同一网段中。

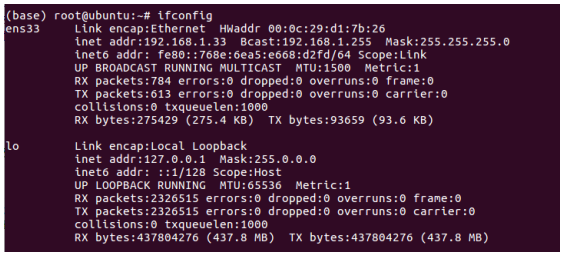

图 22

4. 在 PC 的终端里，测试网络的连通性。

- **Windows 操作系统**：打开 cmd，输入 ping 192.168.1.100。如下图所示，通信成功。

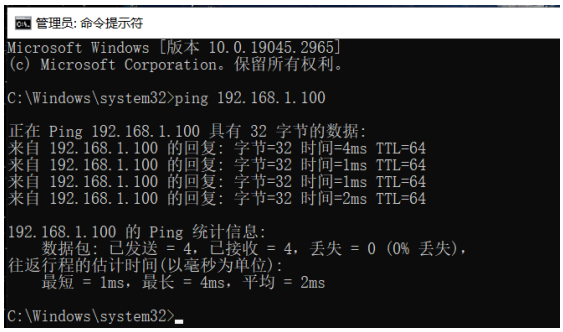

图 23

- **Linux 操作系统**：打开 terminal，输入 ping 192.168.1.100。如下图所示，通信成功。

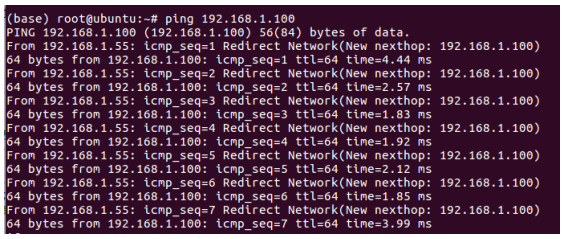

图 24

3.2 C++ SDK

3.2.1 在 Linux 环境下用 QtCreator 编译运行 CMake 工程

下面讲述的是如何用 QtCreator 编译运行 SDK 开发包中的 example_startup.cpp 示例，编译运行成功后，示教器将上电成功。

注意：如果 SDK 客户端连接真实机械臂，必须将代码中的 IP 地址设置成机械臂的 IP 地址。如果在 AuboSim 虚拟机中运行工程并且 SDK 客户端连接的是虚拟机，则 IP 地址为 127.0.0.1。

1. 打开 QtCreator，点击 Open

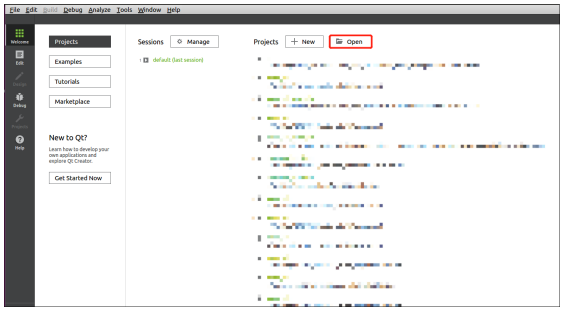

图 25

2. 找到 SDK 包中 C++ 示例所在的路径，选中 CMakeLists.txt，点击 Open

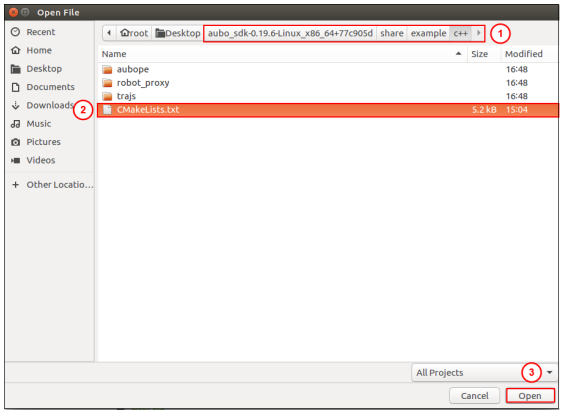

图 26

3. 配置编译器

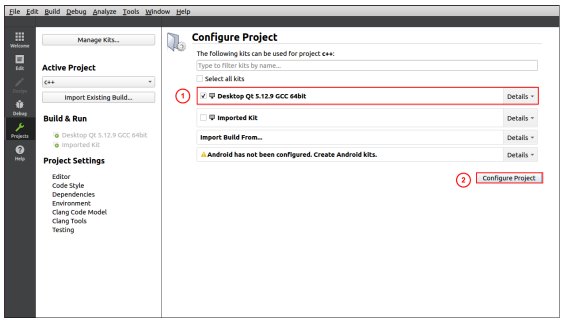

图 27

4. 选中 aubo*sdk*example，右键，点击 Run CMake

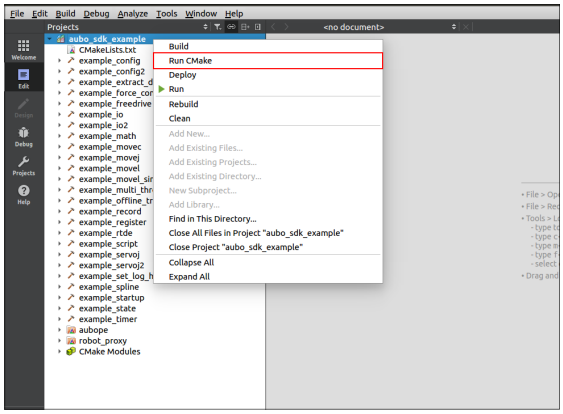

图 28

5. 在 Build 步骤中，在 Tool arguments 栏填写 -j8，选择 example_startup

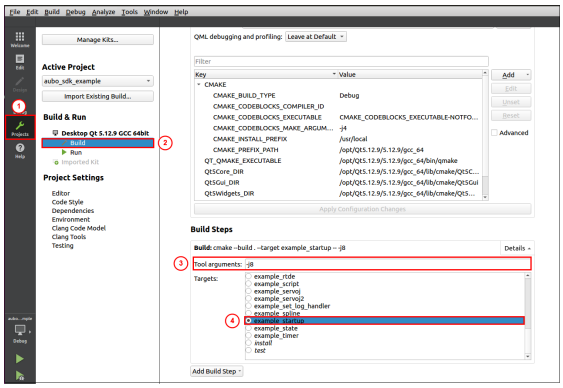

图 29

6. 选择 example_startup

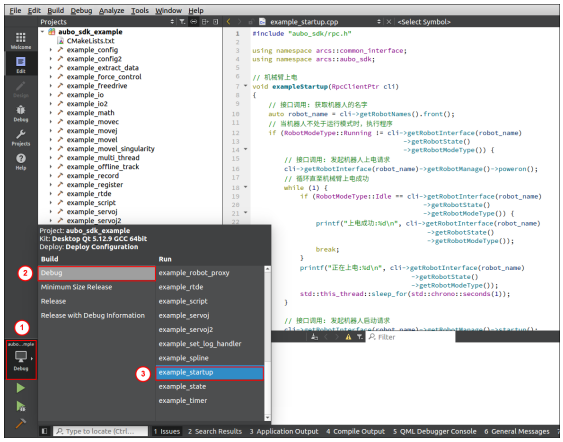

图 30

7. 编译和运行

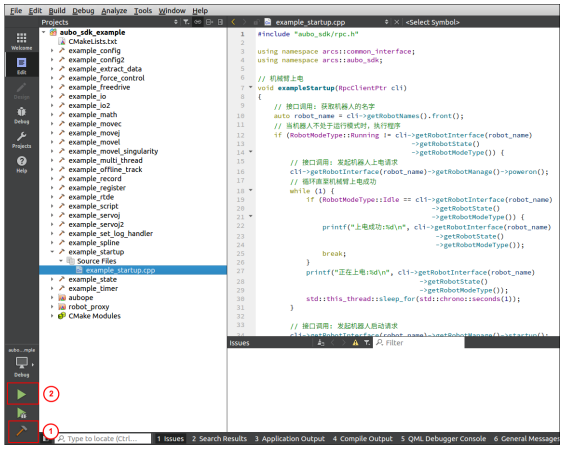

图 31

8. 终端打印信息

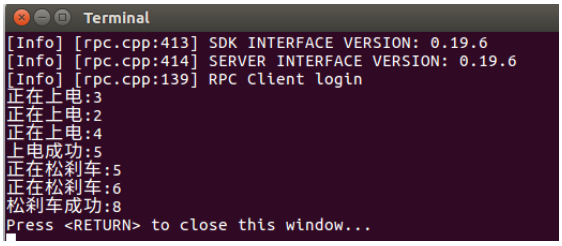

图 32

9. 示教器右上方显示的机器人状态为运行，表明机器人已上电

图 33

3.2.2 在 Windows 环境下用 QtCreator 编译运行 CMake 工程

下面讲述的是如何用 QtCreator 编译运行 SDK 开发包中的 example_startup.cpp 示例，编译运行成功后，示教器将上电成功。

注意：为了确保 SDK 客户端与机械臂能通讯成功，所以在编译运行前，必须将代码中的 IP 地址修改成 AuboSim 虚拟机或者真实机械臂的 IP 地址。

1. 打开 QtCreator，点击 Open

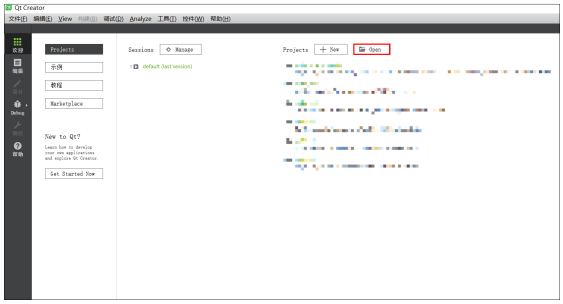

图 34

2. 找到 SDK 包中 C++ 示例所在的路径，选中 CMakeLists.txt，点击 Open

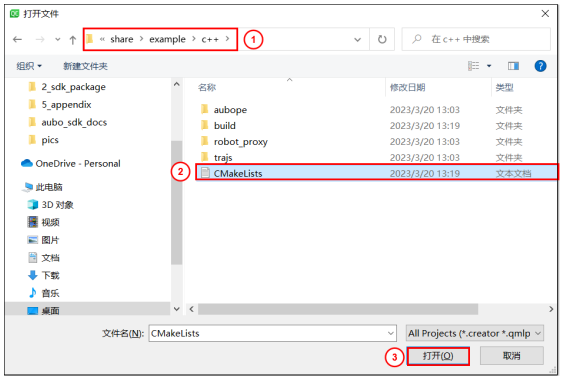

图 35

3. 配置编译器

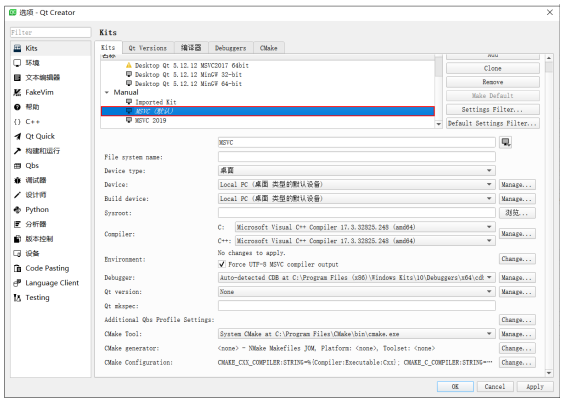

图 36

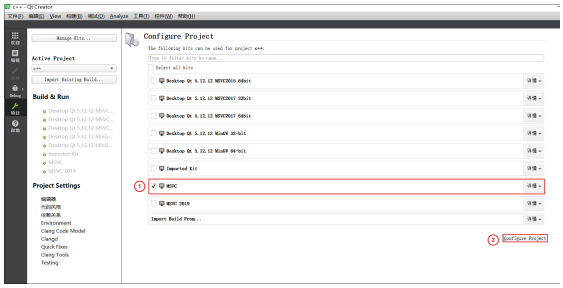

图 37

4. 选中 aubo*sdk*example，右键，点击 Run CMake

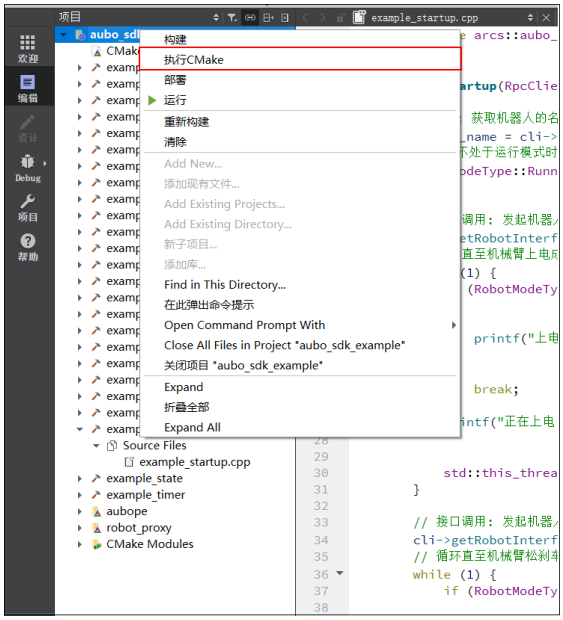

图 38

5. 在 Build 步骤中，在 Tool arguments 栏填写 -j8，选择 example_startup

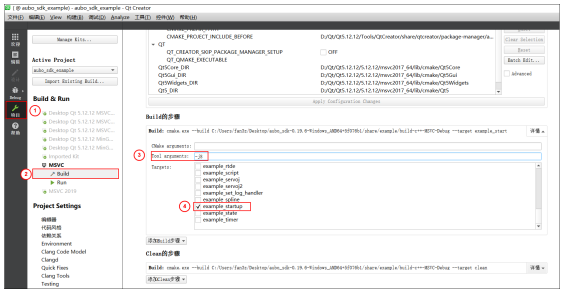

图 39

6. 选择 example_startup

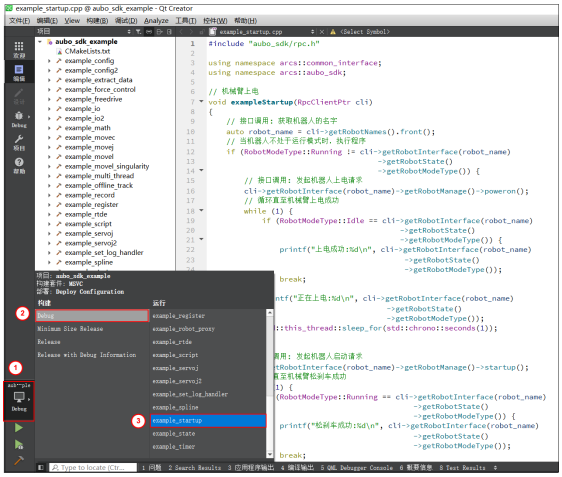

图 40

7. 编译和运行

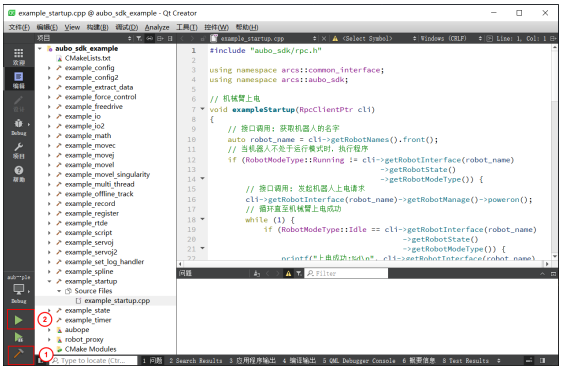

图 41

8. 控制台打印信息

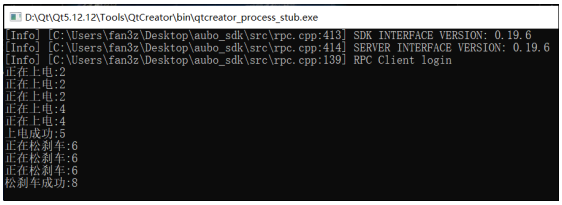

图 42

如果控制台打印为中文乱码，请按下图设置编码，然后重新编译运行

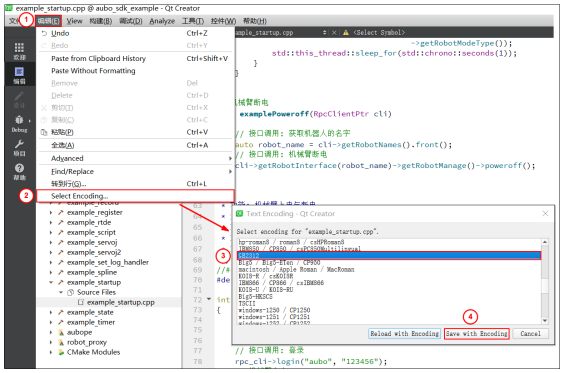

图 43

9. 示教器右上方显示的机器人状态为运行，表明机器人已上电

图 44

3.2.3 在 Windows 环境下用 Visual Studio 编译运行 CMake 工程

下面讲述的是如何用 Visual Studio 编译运行 SDK 开发包中的 example_startup.cpp 示例，编译运行成功后，示教器将上电成功。

注意：为了确保 SDK 客户端与机械臂能通讯成功，所以在编译运行前，必须将代码中的 IP 地址修改成 AuboSim 虚拟机或者真实机械臂的 IP 地址。

1. 打开 Visual Studio，点击”打开本地文件夹“。

图 45

2. 找到 SDK 包中 C++ 示例所在的路径，选中《c++》文件夹，点击“选择文件夹”。

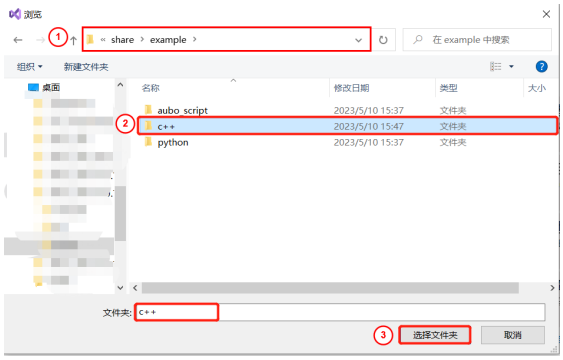

图 46

3. 构建配置，选择 x64-Release。

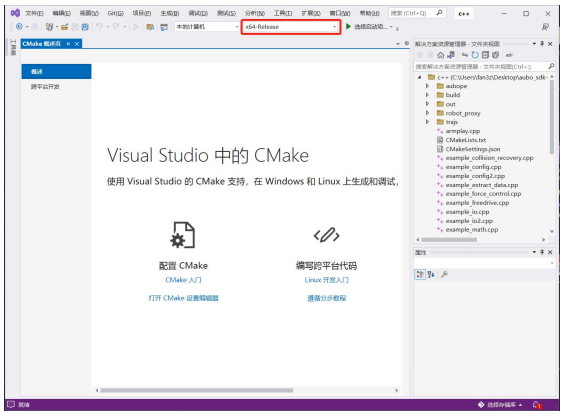

图 47

如果 VS 中没有 Release 模式，可按照如下步骤操作：

1st：点击”管理配置 “。

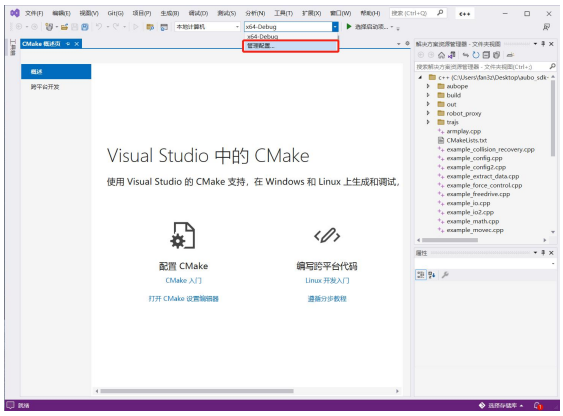

图 48

2nd：在 CMakeSettings.json 文件中，先点击 “+” 图标，再选择 “x64-Release”，点击” 选择 “。

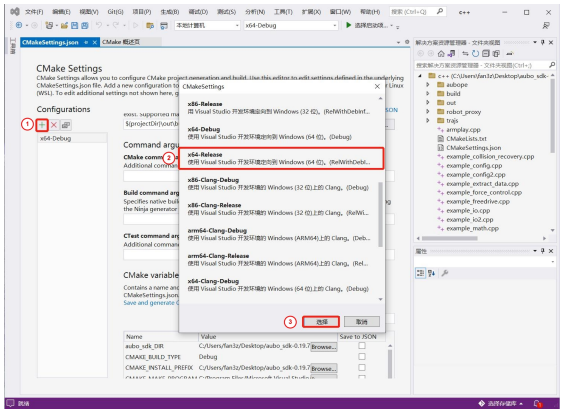

图 49

如下图所示，”x64-Release” 配置成功。

图 50

4. 如果编译报错找不到 aubo*sdk.dll* 文件，则将 *aubo*sdk.dll 移动到生成的可执行程序目录里，即../example/c++/build/bin 路径下。

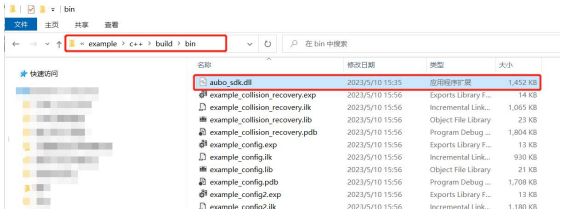

图 51

5. 选择 example_startup.exe，编译运行。

图 52

6. 控制台打印信息。

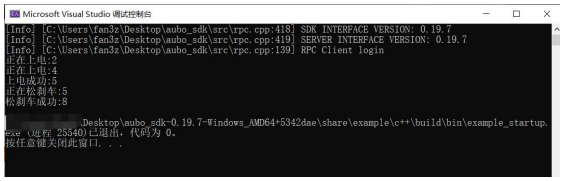

图 53

7. 示教器右上方显示的机器人状态为运行，表明机器人已上电

图 54

3.2.4 在 Windows 环境下用 Visual Studio 编译运行非 CMake 工程

下面讲述的是如何用 Visual Studio 编译运行 SDK 开发包中的 example_startup.cpp 示例，编译运行成功后，示教器将上电成功。

注意：为了确保 SDK 客户端与机械臂能通讯成功，所以在编译运行前，必须将代码中的 IP 地址修改成 AuboSim 虚拟机或者真实机械臂的 IP 地址。

1. 打开 Visual Studio，点击”打开项目或解决方案“。

图 55

2. 找到 SDK 包中 visual*studio.sln* 所在的路径，选中 *visual*studio.sln，点击“打开”。

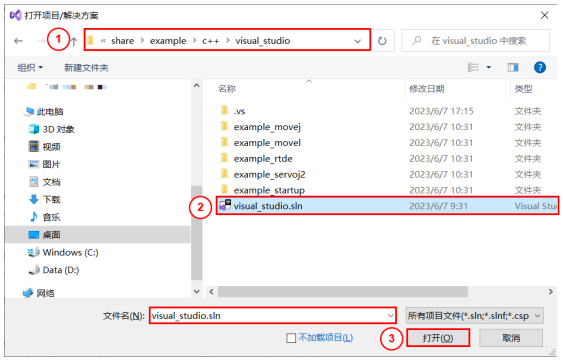

图 56

3. 将想要运行的工程项目设置为启动项目。

   这里以 example*startup* 项目为例，在“解决方案资源管理器”中，选中 *example*startup，右键，点击“设为启动项目”。

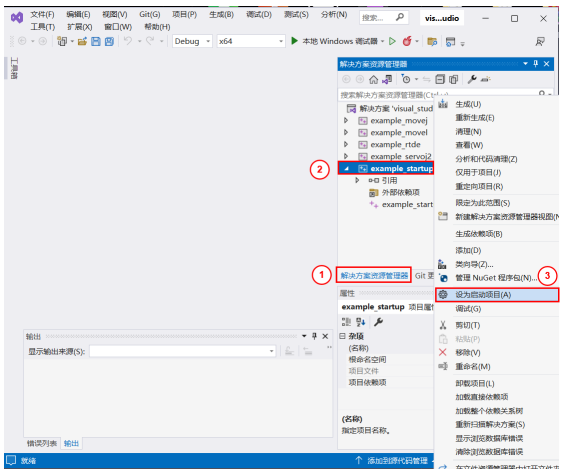

图 57

4. 打开 example_startup.cpp，修改机器人 ip 地址，选择 Release 或者 Debug，调试运行。

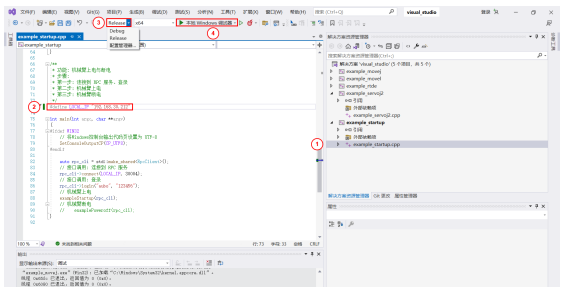

图 58

如果打开 example_startup.cpp 的时候，出现如下错误：

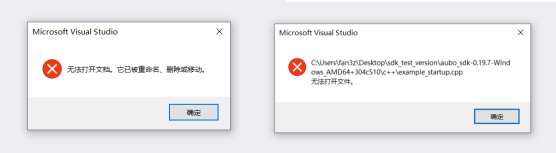

图 59

则可以通过下面的方式重新添加 example_startup.cpp 文件。

选中 example_startup.cpp，右键，点击“从项目中排除”。

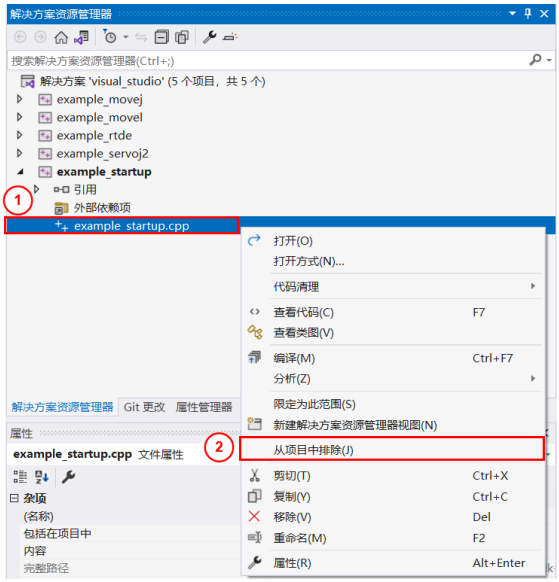

图 60

选中 example_startup，右键，点击“添加”，点击“现有项”。

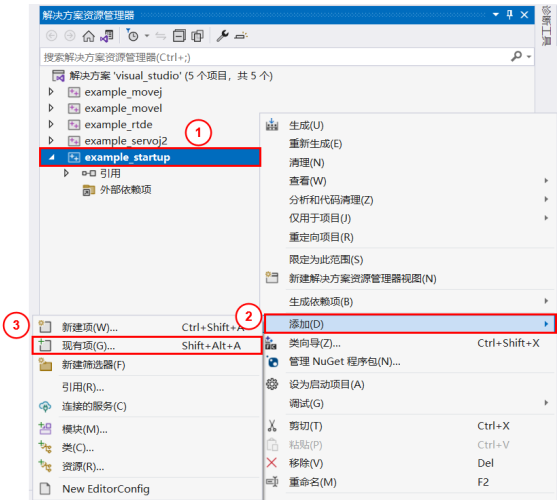

图 61

找到并选中 example_startup.cpp，点击“添加”。

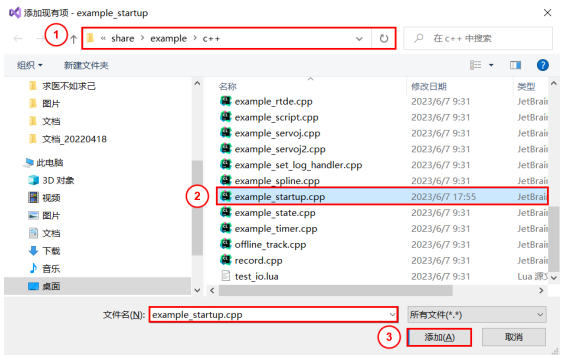

图 62

5. 控制台打印信息如下，显示上电成功。

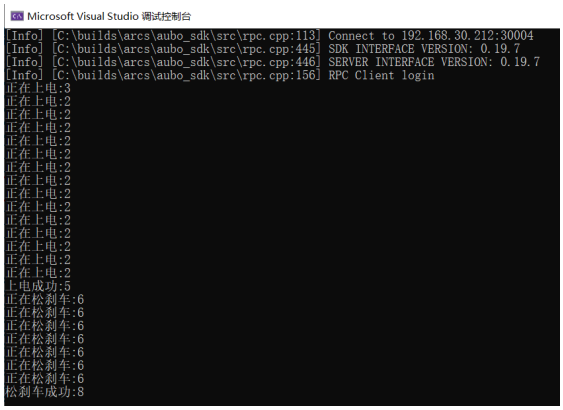

图 63

6. 示教器右上方显示的机器人状态为运行，表明机器人已上电

图 64

3.3 Python SDK

3.3.1 在 Linux 环境下

下面讲述的是如何用 PyCharm 编译运行 SDK 开发包中的 example_startup.py 示例，运行成功后，示教器将先上电后断电。

注意：如果 SDK 客户端连接真实机械臂，必须将代码中的 IP 地址设置成机械臂的 IP 地址。如果在 AuboSim 虚拟机中运行工程并且 SDK 客户端连接的是虚拟机，则 IP 地址为 127.0.0.1。

1. 打开 PyCharm

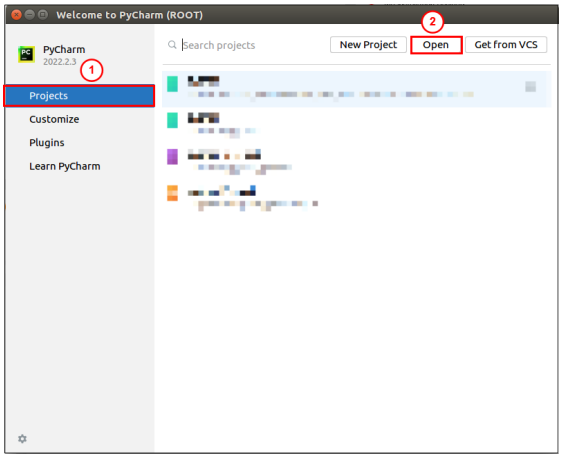

图 65

2. 找到 Python 示例所在的目录，选中 python，点击 OK

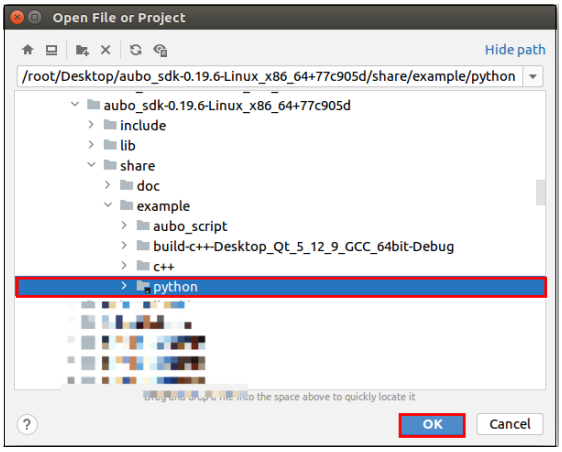

图 66

3. 选中 example_startup.py，点击运行按钮

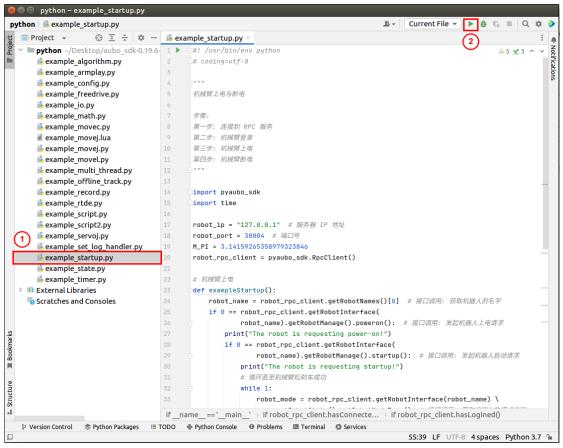

图 67

4. 程序运行结果

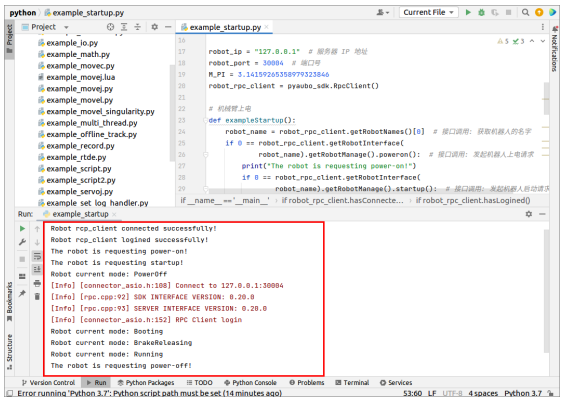

图 68

5. 机器人会先上电然后断电

机器人上电成功，则示教器右上方显示的机器人状态为运行：

图 69

机器人断电成功，则示教器右上方显示的机器人状态为断电：

图 70

3.3.2 在 Windows 环境下

下面讲述的是如何用 PyCharm 编译运行 SDK 开发包中的 example_startup.py 示例，运行成功后，示教器将先上电后断电。

注意：为了确保 SDK 客户端与机械臂能通讯成功，所以在编译运行前，必须将代码中的 IP 地址修改成 AuboSim 虚拟机或者真实机械臂的 IP 地址。

1. 打开 PyCharm

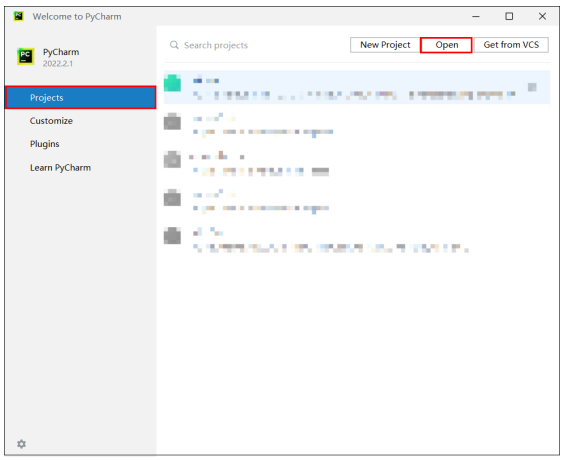

图 71

2. 找到 Python 示例所在的目录，选中 python，点击 OK

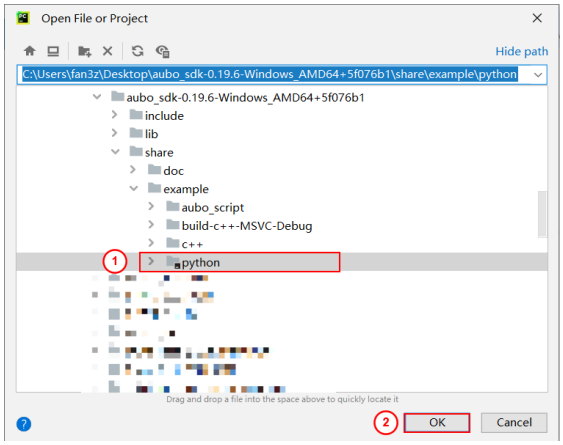

图 72

3. 选中 example_startup.py，点击运行按钮

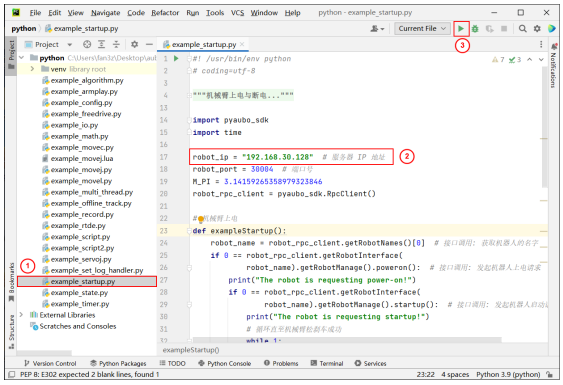

图 73

4. 程序运行结果

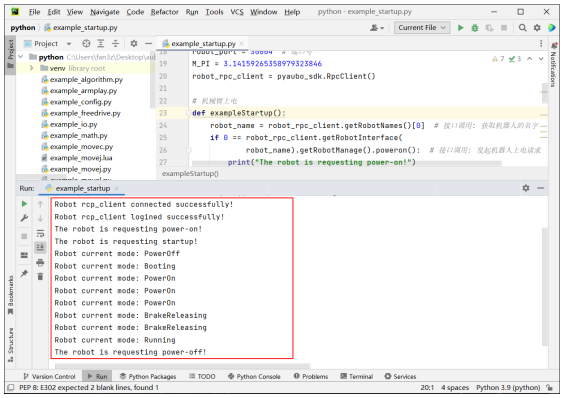

图 74

5. 机器人会先上电然后断电

机器人上电成功，则示教器右上方显示的机器人状态为运行：

图 75

机器人断电成功，则示教器右上方显示的机器人状态为断电：

图 76

4 故障排除

AUBO SDK 提供了错误码、异常和日志，它们是分析、判断和解决 SDK 问题或错误的重要工具。它们可以帮助开发者更好地调试和管理代码，提高代码的可维护性、可靠性和稳定性。日志记录应用程序运行过程中的详细信息，包含程序运行发生错误时的错误码以及说明。异常可用于诊断和处理程序运行过程中出现的非正常情况，比如程序中止或者崩溃。开发者可以根据具体的场景和需求选择合适的方式来调试错误。

4.1 错误码

错误码分为 3 种类型：系统错误码、硬件抽象层错误码和运行时错误码，分别定义在 `system_error.h`、`hal_error.h` 和 `rtm_error.h` 三个头文件中。如想查看错误码及其对应的说明，可查看附录中的*错误码汇总表*。

4.2 日志

AUBO SDK 提供日志来记录程序运行过程中的事件信息、错误信息等详细信息。日志是以文本形式记录，包括时间戳、日志级别、事件描述、错误信息等等，它还提供历史记录，可以帮助开发人员或者技术支持人员在出现问题时快速定位问题，并提供程序运行的详细信息。

4.2.1 日志级别

日志级别表示日志的严重性，范围为从 0 到 5。

LogLevel	值	描述
FATAL	0	严重的错误
ERROR	1	错误
WARNING	2	警告
INFO	3	通知
DEBUG	4	调试
BACKTRACE	5	跟踪

4.2.2 查看日志

用户可以通过以下两种方式查看日志：

1. 日志文件：在 AUBO Sim 虚拟机或者示教器中，日志文件所在路径 `/root/arcs_ws/log`。其中 `aubo_control.log` 表示服务器日志，`aubo_scope.log` 表示示教器日志。

2. 终端实时打印日志：在 AUBO Sim 虚拟机中或者示教器中的打开终端，切换到日志所在的文件夹路径下，输入 `tail -f aubo_control.log` 或者 `tail -f aubo_scope.log`。用户在运行程序的时候，可以在终端中实时看到服务器或者示教器的日志打印信息，监测程序运行情况、复现错误问题或查找错误原因。

4.3 异常

AUBO SDK 支持异常处理，在其头文件 type_def.h 定义了 AuboException 类。程序在运行过程中可能会出现错误而导致程序终止或者程序崩溃等非正常情况。开发人员可以在程序里添加代码来异常捕获，来避免程序崩溃或者无响应等问题，并定位错误发生的位置或原因等。

如果用 C++ 语言来编写程序，则使用 try/catch 来捕获并处理可能出现的异常情况。代码如下：

```
try {
        ...// 出现异常的代码
}
catch (const arcs::aubo_sdk::AuboException& e) {
    // 打印异常信息
    std::cout << "An exception occurs: " << e.what() << std::endl;
    // 打印错误码
    std::cout << "Exception code: " << e.code() << std::endl;
    // 打印异常类型
    std::cout << "Exception type: " << e.type() << std::endl;
}
```

如果用 Python 进行开发，则使用 try/except 来做异常处理。代码如下：

```
try:
    ...# 出现异常的代码
except pyaubo_sdk.AuboException as ex:
    print("An exception occurs: ", ex)
```

5 附录

5.1 RTDE 菜单

5.1.1 RTDE 输入菜单

表 1: RTDE 输入菜单

名称	数据类型	说明
set_recipe	RtdeRecipe	Set
input_bit_registers0_to_31	int	General purpose bits This range of the boolean input registers is reserved forFieldBus/PLC interface usage.
input_bit_registers32_to_63	int	General purpose bits This range of the boolean input registers is reserved forFieldBus/PLC interface usage.
input_bit_registers64_to_127	int64_t	64 general purpose bits X: [64..127] - The upper range of the boolean input registers canbe used by external RTDE clients (i.e AUBOCAPS).
input_int_registers_0	int	48 general purpose integer registers X: [0..23] - The lower range of the integer input registers isreserved for FieldBus/PLC interface usage. X: [24..47] - Theupper range of the integer input registers can be used byexternal RTDE clients (i.e AUBOCAPS).
input_float_registers_0	float	48 general purpose integer registers X: [0..23] - The lower range of the integer input registers isreserved for FieldBus/PLC interface usage. X: [24..47] - Theupper range of the integer input registers can be used byexternal RTDE clients (i.e AUBOCAPS).
input_double_registers_0	double	48 general purpose double registersX: [0..23] - The lower range of the double input registers is reserved for FieldBus/PLC interface usage.X: [24..47] - The upper range of the double input registers can be used by external RTDE clients (i.e AUBOCAPS).

下一页继续

表 1: RTDE 输入菜单

名称	数据类型	说明
R1_speed_slider_mask	double	0 = don't change speed slider with this input = use speed_slider_fraction to set speed slider value
R1_speed_slider_fraction	double	new speed slider value
R1_standard_digital_output_mask	uint32_t	Standard digital outputs
R1_configurable_digital_output_mask	uint32_t	Configurable digital outputs
R1_standard_digital_output	uint32_t	Standard digital outputs
R1_configurable_digital_output	uint32_t	Configurable digital outputs
R1_tool_digital_output	uint32_t	Tool digital outputs its 0-1: output state, remaining bits are reserved for future use
R1_standard_analog_output_type	std::vector<int>	Output domain {0=current[A], 1=voltage[V]} Bits 0-1: standard_analog_output_0
R1_standard_analog_output_mask	uint32_t	Standard analog output 0 (ratio) [0..1]
R1_standard_analog_output	std::vector<double>	Standard analog output 1 (ratio) [0..1]
R1_debug	uint32_t	Debug for internal use
R1_rtde_input_max	int	
R2_speed_slider_mask	double	0 = don't change speed slider with this input = use speed_slider_fraction to set speed slider value
R2_speed_slider_fraction	double	new speed slider value
R2_standard_digital_output_mask	uint32_t	Standard digital outputs
R2_configurable_digital_output_mask	uint32_t	Configurable digital outputs
R2_standard_digital_output	uint32_t	Standard digital outputs
R2_configurable_digital_output	uint32_t	Configurable digital outputs
R2_tool_digital_output	uint32_t	Tool digital outputs its 0-1: output state, remaining bits are reserved for future use
R2_standard_analog_output_type	std::vector<int>	Output domain {0=current[A], 1=voltage[V]} Bits 0-1: standard_analog_output_0
R2_standard_analog_output_mask	uint32_t	Standard analog output 0 (ratio) [0..1]
R2_standard_analog_output	std::vector<double>	Standard analog output 1 (ratio) [0..1]
R2_debug	uint32_t	Debug for internal use
R2_rtde_input_max	int	

下一页继续

表 1: RTDE 输入菜单

名称	数据类型	说明
R3_speed_slider_mask	double	0 = don't change speed slider with this input = use speed_slider_fraction to set speed slider value
R3_speed_slider_fraction	double	new speed slider value
R3_standard_digital_output_mask	uint32_t	Standard digital outputs
R3_configurable_digital_output_mask	uint32_t	Configurable digital outputs
R3_standard_digital_output	uint32_t	Standard digital outputs
R3_configurable_digital_output	uint32_t	Configurable digital outputs
R3_tool_digital_output	uint32_t	Tool digital outputs its 0-1: output state, remaining bits are reserved for future use
R3_standard_analog_output_type	std::vector<int>	Output domain {0=current[A], 1=voltage[V]} Bits 0-1: standard_analog_output_0
R3_standard_analog_output_mask	uint32_t	Standard analog output 0 (ratio) [0..1]
R3_standard_analog_output	std::vector<double>	Standard analog output 1 (ratio) [0..1]
R3_debug	uint32_t	Debug for internal use
R3_rtde_input_max	int	
R4_speed_slider_mask	double	0 = don't change speed slider with this input = use speed_slider_fraction to set speed slider value
R4_speed_slider_fraction	double	new speed slider value
R4_standard_digital_output_mask	uint32_t	Standard digital outputs
R4_configurable_digital_output_mask	uint32_t	Configurable digital outputs
R4_standard_digital_output	uint32_t	Standard digital outputs
R4_configurable_digital_output	uint32_t	Configurable digital outputs
R4_tool_digital_output	uint32_t	Tool digital outputs its 0-1: output state, remaining bits are reserved for future use
R4_standard_analog_output_type	std::vector<int>	Output domain {0=current[A], 1=voltage[V]} Bits 0-1: standard_analog_output_0
R4_standard_analog_output_mask	uint32_t	Standard analog output 0 (ratio) [0..1]
R4_standard_analog_output	std::vector<double>	Standard analog output 1 (ratio) [0..1]
R4_debug	uint32_t	Debug for internal use
R4_rtde_input_max	int	

5.1.2 RTDE 输出菜单

表 2: RTDE 输出菜单

名称	数据类型	说明
timestamp	double	Time elapsed since the controller was started [s]
line_number	int	line number set by setPlanContext
runtime_state	RuntimeState	Program state
output_bit_registers_0_to_63	int64_t	64 [000..063] General purpose bits
output_bit_registers_64_to_127	int64_t	64 [064..127] general purpose bits
output_int_registers_0	int	48 general purpose integer registersX: [0..23] - The lower range of the integer output registers isreserved for FieldBus/PLC interface usage. X: [24..47] - The upper range of the integer output registers can be used by external RTDE clients (i.e AUBOCAPS).
output_float_registers_0	int	48 general purpose integer registersX: [0..23] - The lower range of the integer output registers isreserved for FieldBus/PLC interface usage. X: [24..47] - The upper range of the integer output registers can be used by external RTDE clients (i.e AUBOCAPS).
output_double_registers_0	double	48 general purpose double registersX: [0..23] - The lower range of the double output registers isreserved for FieldBus/PLC interface usage. X: [24..47] - The upper range of the double output registers can be used by external RTDE clients (i.e AUBOCAPS).
input_bit_registers_r0_to_63	int64_t	[0..63] General purpose bits This range of the boolean outputregisters is reserved for FieldBus/PLC interface usage.
input_bit_registers_r64_to_127	int64_t	64 [64..127] general purpose bits

下一页继续

表 2: RTDE 输出菜单

名称	数据类型	说明
input_int_registers_r0	int	([0 .. 48]) 48 general purpose integer registersX: [0..23] - The lower range of the integer input registers isreserved for FieldBus/PLC interface usage. X: [24..47] - Theupper range of the integer input registers can be used byexternal RTDE clients (i.e AUBOCAPS).
input_float_registers_r0	int	([0 .. 48]) 48 general purpose integer registersX: [0..23] - The lower range of the integer input registers isreserved for FieldBus/PLC interface usage. X: [24..47] - Theupper range of the integer input registers can be used byexternal RTDE clients (i.e AUBOCAPS).
input_double_registers_r0	double	([0 .. 48]) 48 general purpose double registersX: [0..23] - The lower range of the double input registers isreserved for FieldBus/PLC interface usage. X: [24..47] - Theupper range of the double input registers can be used byexternal RTDE clients (i.e AUBOCAPS).
modbus_signals	std::vector<int>	Modbus signals from connected modbus slaves
modbus_signals_errors	std::vector<int>	Modbus signals request status from connected modbus slaves
R1_message	RobotMsg	Robot message from controller
R1_target_q	std::vector<double>	Target joint positions
R1_target_qd	std::vector<double>	Target joint velocities
R1_target_qdd	std::vector<double>	Target joint accelerations
R1_target_current	std::vector<double>	Target joint currents
R1_target_moment	std::vector<double>	Target joint moments (torques)
R1_actual_q	std::vector<double>	Actual joint positions
R1_actual_qd	std::vector<double>	Actual joint velocities
R1_actual_current	std::vector<double>	Actual joint currents
R1_joint_control_output	std::vector<double>	Joint control currents
R1_joint_temperatures	std::vector<double>	Temperature of each joint in degrees Celsius
R1_actual_joint_voltage	std::vector<double>	Actual joint voltages

下一页继续

表 2: RTDE 输出菜单

名称	数据类型	说明
R1_joint_mode	std::vector<JointStateType>	Joint control modes Please see Remote Control Via TCP/IP - 16496
R1_actual_execution_time	double	Controller real-time thread execution time
R1_robot_mode	RobotModeType	Robot mode Please see Remote Control Via TCP/IP - 16496
R1_safety_mode	SafetyModeType	Safety mode Please see Remote Control Via TCP/IP - 16496
R1_safety_status	unknown	Safety ststus
R1_robot_status_bits	unknown	Bits 0-3: Is power on
R1_safety_status_bits	unknown	Bits 0-10: Is normal mode
R1_speed_scaling	double	Speed scaling of the trajectory limiter
R1_target_speed_fraction	double	Target speed fraction
R1_actual_TCP_pose	std::vector<double>	Actual Cartesian coordinates of the tool:(x,y,z,rx,ry,rz), where rx, ry and rz is a rotationvector representation of the tool orientation
R1_actual_TCP_speed	std::vector<double>	Actual speed of the tool given in Cartesian coordinates
R1_actual_TCP_force	std::vector<double>	Generalized forces in the TCP
R1_target_TCP_pose	std::vector<double>	Target Cartesian coordinates of the tool:(x,y,z,rx,ry,rz), where rx, ry and rz is a rotationvector representation of the tool orientation
R1_target_TCP_speed	std::vector<double>	Target speed of the tool given in Cartesian coordinates
R1_elbow_position	std::vector<double>	Position of robot elbow in Cartesian Base Coordinates
R1_elbow_velocity	std::vector<double>	Velocity of robot elbow in Cartesian Base Coordinates
R1_actual_momentum	std::vector<double>	Norm of Cartesian linear momentum
R1_tcp_force_scalar	std::vector<double>	TCP force scalar [N]
R1_actual_main_voltage	unknown	Safety Control Board: Main voltage
R1_actual_robot_voltage	unknown	Safety Control Board: Robot voltage (48V)
R1_actual_robot_current	unknown	Safety Control Board: Robot current
R1_operationalModeSelectorInput	unknown	
R1_threePositionEnablingDeviceInput	unknown	
R1_masterboard_temperature	unknown	

下一页继续

表 2: RTDE 输出菜单

名称	数据类型	说明
R1_standard_digital_input_bits	uint64_t	Current state of the digital inputs. 0-7:Standard, 8-15: Configurable, 16-17: Tool
R1_tool_digital_input_bits	uint64_t	Current state of the digital inputs. 0-7:Standard, 8-15: Configurable, 16-17: Tool
R1_configurable_digital_input_bits	uint64_t	Current state of the digital inputs. 0-7:Standard, 8-15: Configurable, 16-17: Tool
R1_link_digital_input_bits	uint64_t	Current state of the digital inputs. 0-7:Standard, 8-15: Configurable, 16-17: Tool
R1_standard_digital_output_bits	uint64_t	Current state of the digital outputs. 0-7: Standard, 8-15:Configurable, 16-17: Tool
R1_tool_digital_output_bits	uint64_t	Current state of the digital outputs. 0-7: Standard, 8-15:Configurable, 16-17: Tool
R1_configurable_digital_output_bits	uint64_t	Current state of the digital outputs. 0-7: Standard, 8-15:Configurable, 16-17: Tool
R1_link_digital_output_bits	uint64_t	Current state of the digital outputs. 0-7: Standard, 8-15:Configurable, 16-17: Tool
R1_standard_analog_input_values	std::vector<double>	
R1_tool_analog_input_values	std::vector<double>	
R1_standard_analog_output_values	std::vector<double>	
R1_tool_analog_output_values	std::vector<double>	
R1_master_io_current	unknown	I/O current [A]
R1_euromap67_input_bits	unknown	Euromap67 input bits
R1_euromap67_output_bits	unknown	Euromap67 output bits
R1_euromap67_24V_voltage	unknown	Euromap 24V voltage [V]
R1_euromap67_24V_current	unknown	Euromap 24V current [A]
R1_tool_mode	unknown	Tool mode Please see Remote Control Via TCP/IP - 16496
R1_tool_output_mode	unknown	The current output mode
R1_tool_output_voltage	unknown	Tool output voltage [V]
R1_tool_output_current	unknown	Tool current [A]
R1_tool_voltage_48V	unknown	
R1_tool_current	unknown	
R1_tool_temperature	unknown	Tool temperature in degrees Celsius
R1_actual_tool_accelerometer	unknown	Tool x, y and z accelerometer values
R1_motion_progress	unknown	Trajectory running progress

下一页继续

表 2: RTDE 输出菜单

名称	数据类型	说明
R1_actual_qdd	unknown	Actual joint accelerations
R1_rtde_output_max	int	
R2_message	RobotMsg	Robot message from controller
R2_target_q	std::vector<double>	Target joint positions
R2_target_qd	std::vector<double>	Target joint velocities
R2_target_qdd	std::vector<double>	Target joint accelerations
R2_target_current	std::vector<double>	Target joint currents
R2_target_moment	std::vector<double>	Target joint moments (torques)
R2_actual_q	std::vector<double>	Actual joint positions
R2_actual_qd	std::vector<double>	Actual joint velocities
R2_actual_current	std::vector<double>	Actual joint currents
R2_joint_control_output	std::vector<double>	Joint control currents
R2_joint_temperatures	std::vector<double>	Temperature of each joint in degrees Celsius
R2_actual_joint_voltage	std::vector<double>	Actual joint voltages
R2_joint_mode	std::vector<JointStateType>	Joint control modes Please see Remote Control Via TCP/IP - 16496
R2_actual_execution_time	double	Controller real-time thread execution time
R2_robot_mode	RobotModeType	Robot mode Please see Remote Control Via TCP/IP - 16496
R2_safety_mode	SafetyModeType	Safety mode Please see Remote Control Via TCP/IP - 16496
R2_safety_status	unknown	Safety ststus
R2_robot_status_bits	unknown	Bits 0-3: Is power on
R2_safety_status_bits	unknown	Bits 0-10: Is normal mode
R2_speed_scaling	double	Speed scaling of the trajectory limiter
R2_target_speed_fraction	double	Target speed fraction
R2_actual_TCP_pose	std::vector<double>	Actual Cartesian coordinates of the tool:(x,y,z,rx,ry,rz), where rx, ry and rz is a rotationvector representation of the tool orientation
R2_actual_TCP_speed	std::vector<double>	Actual speed of the tool given in Cartesian coordinates
R2_actual_TCP_force	std::vector<double>	Generalized forces in the TCP
R2_target_TCP_pose	std::vector<double>	Target Cartesian coordinates of the tool:(x,y,z,rx,ry,rz), where rx, ry and rz is a rotationvector representation of the tool orientation
R2_target_TCP_speed	std::vector<double>	Target speed of the tool given in Cartesian coordinates
R2_elbow_position	std::vector<double>	Position of robot elbow in Cartesian Base Coordinates

下一页继续

表 2: RTDE 输出菜单

名称	数据类型	说明
R2_elbow_velocity	std::vector<double>	Velocity of robot elbow in Cartesian Base Coordinates
R2_actual_momentum	std::vector<double>	Norm of Cartesian linear momentum
R2_tcp_force_scalar	std::vector<double>	TCP force scalar [N]
R2_actual_main_voltage	unknown	Safety Control Board: Main voltage
R2_actual_robot_voltage	unknown	Safety Control Board: Robot voltage (48V)
R2_actual_robot_current	unknown	Safety Control Board: Robot current
R2_operationalModeSelectorInput	unknown	
R2_threePositionEnablingDeviceInput	unknown	
R2_masterboard_temperature	unknown	
R2_standard_digital_input_bits	uint64_t	Current state of the digital inputs. 0-7:Standard, 8-15: Configurable, 16-17: Tool
R2_tool_digital_input_bits	uint64_t	Current state of the digital inputs. 0-7:Standard, 8-15: Configurable, 16-17: Tool
R2_configurable_digital_input_bits	uint64_t	Current state of the digital inputs. 0-7:Standard, 8-15: Configurable, 16-17: Tool
R2_link_digital_input_bits	uint64_t	Current state of the digital inputs. 0-7:Standard, 8-15: Configurable, 16-17: Tool
R2_standard_digital_output_bits	uint64_t	Current state of the digital outputs. 0-7: Standard, 8-15:Configurable, 16-17: Tool
R2_tool_digital_output_bits	uint64_t	Current state of the digital outputs. 0-7: Standard, 8-15:Configurable, 16-17: Tool
R2_configurable_digital_output_bits	uint64_t	Current state of the digital outputs. 0-7: Standard, 8-15:Configurable, 16-17: Tool
R2_link_digital_output_bits	uint64_t	Current state of the digital outputs. 0-7: Standard, 8-15:Configurable, 16-17: Tool
R2_standard_analog_input_values	std::vector<double>	
R2_tool_analog_input_values	std::vector<double>	
R2_standard_analog_output_values	std::vector<double>	
R2_tool_analog_output_values	std::vector<double>	
R2_master_io_current	unknown	I/O current [A]
R2_euromap67_input_bits	unknown	Euromap67 input bits
R2_euromap67_output_bits	unknown	Euromap67 output bits

下一页继续

表 2: RTDE 输出菜单

名称	数据类型	说明
R2_euromap67_24V_voltage	unknown	Euromap 24V voltage [V]
R2_euromap67_24V_current	unknown	Euromap 24V current [A]
R2_tool_mode	unknown	Tool mode Please see Remote Control Via TCP/IP - 16496
R2_tool_output_mode	unknown	The current output mode
R2_tool_output_voltage	unknown	Tool output voltage [V]
R2_tool_output_current	unknown	Tool current [A]
R2_tool_voltage_48V	unknown	
R2_tool_current	unknown	
R2_tool_temperature	unknown	Tool temperature in degrees Celsius
R2_actual_tool_accelerometer	unknown	Tool x, y and z accelerometer values
R2_motion_progress	unknown	Trajectory running progress
R2_actual_qdd	unknown	Actual joint accelerations
R2_rtde_output_max	int	
R3_message	RobotMsg	Robot message from controller
R3_target_q	std::vector<double>	Target joint positions
R3_target_qd	std::vector<double>	Target joint velocities
R3_target_qdd	std::vector<double>	Target joint accelerations
R3_target_current	std::vector<double>	Target joint currents
R3_target_moment	std::vector<double>	Target joint moments (torques)
R3_actual_q	std::vector<double>	Actual joint positions
R3_actual_qd	std::vector<double>	Actual joint velocities
R3_actual_current	std::vector<double>	Actual joint currents
R3_joint_control_output	std::vector<double>	Joint control currents
R3_joint_temperatures	std::vector<double>	Temperature of each joint in degrees Celsius
R3_actual_joint_voltage	std::vector<double>	Actual joint voltages
R3_joint_mode	std::vector<JointStateType>	Joint control modes Please see Remote Control Via TCP/IP - 16496
R3_actual_execution_time	double	Controller real-time thread execution time
R3_robot_mode	RobotModeType	Robot mode Please see Remote Control Via TCP/IP - 16496
R3_safety_mode	SafetyModeType	Safety mode Please see Remote Control Via TCP/IP - 16496
R3_safety_status	unknown	Safety ststus
R3_robot_status_bits	unknown	Bits 0-3: Is power on
R3_safety_status_bits	unknown	Bits 0-10: Is normal mode
R3_speed_scaling	double	Speed scaling of the trajectory limiter
R3_target_speed_fraction	double	Target speed fraction

下一页继续

表 2: RTDE 输出菜单

名称	数据类型	说明
R3_actual_TCP_pose	std::vector<double>	Actual Cartesian coordinates of the tool:(x,y,z,rx,ry,rz), where rx, ry and rz is a rotationvector representation of the tool orientation
R3_actual_TCP_speed	std::vector<double>	Actual speed of the tool given in Cartesian coordinates
R3_actual_TCP_force	std::vector<double>	Generalized forces in the TCP
R3_target_TCP_pose	std::vector<double>	Target Cartesian coordinates of the tool:(x,y,z,rx,ry,rz), where rx, ry and rz is a rotationvector representation of the tool orientation
R3_target_TCP_speed	std::vector<double>	Target speed of the tool given in Cartesian coordinates
R3_elbow_position	std::vector<double>	Position of robot elbow in Cartesian Base Coordinates
R3_elbow_velocity	std::vector<double>	Velocity of robot elbow in Cartesian Base Coordinates
R3_actual_momentum	std::vector<double>	Norm of Cartesian linear momentum
R3_tcp_force_scalar	std::vector<double>	TCP force scalar [N]
R3_actual_main_voltage	unknown	Safety Control Board: Main voltage
R3_actual_robot_voltage	unknown	Safety Control Board: Robot voltage (48V)
R3_actual_robot_current	unknown	Safety Control Board: Robot current
R3_operationalModeSelectorInput	unknown	
R3_threePositionEnablingDeviceInput	unknown	
R3_masterboard_temperature	unknown	
R3_standard_digital_input_bits	uint64_t	Current state of the digital inputs. 0-7:Standard, 8-15: Configurable, 16-17: Tool
R3_tool_digital_input_bits	uint64_t	Current state of the digital inputs. 0-7:Standard, 8-15: Configurable, 16-17: Tool
R3_configurable_digital_input_bits	uint64_t	Current state of the digital inputs. 0-7:Standard, 8-15: Configurable, 16-17: Tool
R3_link_digital_input_bits	uint64_t	Current state of the digital inputs. 0-7:Standard, 8-15: Configurable, 16-17: Tool
R3_standard_digital_output_bits	uint64_t	Current state of the digital outputs. 0-7: Standard, 8-15:Configurable, 16-17: Tool

下一页继续

表 2: RTDE 输出菜单

名称	数据类型	说明
R3_tool_digital_output_bits	uint64_t	Current state of the digital outputs. 0-7: Standard, 8-15:Configurable, 16-17: Tool
R3_configurable_digital_output_bits	uint64_t	Current state of the digital outputs. 0-7: Standard, 8-15:Configurable, 16-17: Tool
R3_link_digital_output_bits	uint64_t	Current state of the digital outputs. 0-7: Standard, 8-15:Configurable, 16-17: Tool
R3_standard_analog_input_values	std::vector<double>	
R3_tool_analog_input_values	std::vector<double>	
R3_standard_analog_output_values	std::vector<double>	
R3_tool_analog_output_values	std::vector<double>	
R3_master_io_current	unknown	I/O current [A]
R3_euromap67_input_bits	unknown	Euromap67 input bits
R3_euromap67_output_bits	unknown	Euromap67 output bits
R3_euromap67_24V_voltage	unknown	Euromap 24V voltage [V]
R3_euromap67_24V_current	unknown	Euromap 24V current [A]
R3_tool_mode	unknown	Tool mode Please see Remote Control Via TCP/IP - 16496
R3_tool_output_mode	unknown	The current output mode
R3_tool_output_voltage	unknown	Tool output voltage [V]
R3_tool_output_current	unknown	Tool current [A]
R3_tool_voltage_48V	unknown	
R3_tool_current	unknown	
R3_tool_temperature	unknown	Tool temperature in degrees Celsius
R3_actual_tool_accelerometer	unknown	Tool x, y and z accelerometer values
R3_motion_progress	unknown	Trajectory running progress
R3_actual_qdd	unknown	Actual joint accelerations
R3_rtde_output_max	int	
R4_message	RobotMsg	Robot message from controller
R4_target_q	std::vector<double>	Target joint positions
R4_target_qd	std::vector<double>	Target joint velocities
R4_target_qdd	std::vector<double>	Target joint accelerations
R4_target_current	std::vector<double>	Target joint currents
R4_target_moment	std::vector<double>	Target joint moments (torques)
R4_actual_q	std::vector<double>	Actual joint positions
R4_actual_qd	std::vector<double>	Actual joint velocities
R4_actual_current	std::vector<double>	Actual joint currents
R4_joint_control_output	std::vector<double>	Joint control currents
R4_joint_temperatures	std::vector<double>	Temperature of each joint in degrees Celsius
R4_actual_joint_voltage	std::vector<double>	Actual joint voltages

下一页继续

表 2: RTDE 输出菜单

名称	数据类型	说明
R4_joint_mode	std::vector<JointStateType>	Joint control modes Please see Remote Control Via TCP/IP - 16496
R4_actual_execution_time	double	Controller real-time thread execution time
R4_robot_mode	RobotModeType	Robot mode Please see Remote Control Via TCP/IP - 16496
R4_safety_mode	SafetyModeType	Safety mode Please see Remote Control Via TCP/IP - 16496
R4_safety_status	unknown	Safety ststus
R4_robot_status_bits	unknown	Bits 0-3: Is power on
R4_safety_status_bits	unknown	Bits 0-10: Is normal mode
R4_speed_scaling	double	Speed scaling of the trajectory limiter
R4_target_speed_fraction	double	Target speed fraction
R4_actual_TCP_pose	std::vector<double>	Actual Cartesian coordinates of the tool:(x,y,z,rx,ry,rz), where rx, ry and rz is a rotationvector representation of the tool orientation
R4_actual_TCP_speed	std::vector<double>	Actual speed of the tool given in Cartesian coordinates
R4_actual_TCP_force	std::vector<double>	Generalized forces in the TCP
R4_target_TCP_pose	std::vector<double>	Target Cartesian coordinates of the tool:(x,y,z,rx,ry,rz), where rx, ry and rz is a rotationvector representation of the tool orientation
R4_target_TCP_speed	std::vector<double>	Target speed of the tool given in Cartesian coordinates
R4_elbow_position	std::vector<double>	Position of robot elbow in Cartesian Base Coordinates
R4_elbow_velocity	std::vector<double>	Velocity of robot elbow in Cartesian Base Coordinates
R4_actual_momentum	std::vector<double>	Norm of Cartesian linear momentum
R4_tcp_force_scalar	std::vector<double>	TCP force scalar [N]
R4_actual_main_voltage	unknown	Safety Control Board: Main voltage
R4_actual_robot_voltage	unknown	Safety Control Board: Robot voltage (48V)
R4_actual_robot_current	unknown	Safety Control Board: Robot current
R4_operationalModeSelectorInput	unknown	
R4_threePositionEnablingDeviceInput	unknown	
R4_masterboard_temperature	unknown	

下一页继续

表 2: RTDE 输出菜单

名称	数据类型	说明
R4_standard_digital_input_bits	uint64_t	Current state of the digital inputs. 0-7:Standard, 8-15: Configurable, 16-17: Tool
R4_tool_digital_input_bits	uint64_t	Current state of the digital inputs. 0-7:Standard, 8-15: Configurable, 16-17: Tool
R4_configurable_digital_input_bits	uint64_t	Current state of the digital inputs. 0-7:Standard, 8-15: Configurable, 16-17: Tool
R4_link_digital_input_bits	uint64_t	Current state of the digital inputs. 0-7:Standard, 8-15: Configurable, 16-17: Tool
R4_standard_digital_output_bits	uint64_t	Current state of the digital outputs. 0-7: Standard, 8-15:Configurable, 16-17: Tool
R4_tool_digital_output_bits	uint64_t	Current state of the digital outputs. 0-7: Standard, 8-15:Configurable, 16-17: Tool
R4_configurable_digital_output_bits	uint64_t	Current state of the digital outputs. 0-7: Standard, 8-15:Configurable, 16-17: Tool
R4_link_digital_output_bits	uint64_t	Current state of the digital outputs. 0-7: Standard, 8-15:Configurable, 16-17: Tool
R4_standard_analog_input_values	std::vector<double>	
R4_tool_analog_input_values	std::vector<double>	
R4_standard_analog_output_values	std::vector<double>	
R4_tool_analog_output_values	std::vector<double>	
R4_master_io_current	unknown	I/O current [A]
R4_euromap67_input_bits	unknown	Euromap67 input bits
R4_euromap67_output_bits	unknown	Euromap67 output bits
R4_euromap67_24V_voltage	unknown	Euromap 24V voltage [V]
R4_euromap67_24V_current	unknown	Euromap 24V current [A]
R4_tool_mode	unknown	Tool mode Please see Remote Control Via TCP/IP - 16496
R4_tool_output_mode	unknown	The current output mode
R4_tool_output_voltage	unknown	Tool output voltage [V]
R4_tool_output_current	unknown	Tool current [A]
R4_tool_voltage_48V	unknown	
R4_tool_current	unknown	
R4_tool_temperature	unknown	Tool temperature in degrees Celsius
R4_actual_tool_accelerometer	unknown	Tool x, y and z accelerometer values
R4_motion_progress	unknown	Trajectory running progress

下一页继续

表 2: RTDE 输出菜单

名称	数据类型	说明
R4_actual_qdd	unknown	Actual joint accelerations
R4_rtde_output_max	int	

5.2 错误码汇总表

表 3: 错误码汇总表

错误名称	错误码	说明
DEBUG	0	Debug message {}
POPUP	1	{}
POPUP_DISMISS	2	{}
SYSTEM_HALT	3	{}
JOINT_ERR_OVER_CURRENET	10001	joint{} error: over current
JOINT_ERR_OVER_VOLTAGE	10002	joint{} error: over voltage
JOINT_ERR_LOW_VOLTAGE	10003	joint{} error: low voltage
JOINT_ERR_OVER_TEMP	10004	joint{} error: over temperature
JOINT_ERR_HALL	10005	joint{} error: hall
JOINT_ERR_ENCODER	10006	joint{} error: encoder
JOINT_ERR_ABS_ENCODER	10007	joint{} error: abs encoder
JOINT_ERR_Q_CURRENT	10008	joint{} error: detect current
JOINT_ERR_ENC_POLL	10009	joint{} error: encoder pollustion
JOINT_ERR_ENC_Z_SIGNAL	10010	joint{} error: enocder z signal
JOINT_ERR_ENC_CAL	10011	joint{} error: encoder calibrate
JOINT_ERR_IMU_SENS	10012	joint{} error: IMU sensor
JOINT_ERR_TEMP_SENS	10013	joint{} error: TEMP sensor
JOINT_ERR_CAN_BUS	10014	joint{} error: can bus error
JOINT_ERR_SYS_CUR	10015	joint{} error: system current error
JOINT_ERR_SYS_POS	10016	joint{} error: system position error
JOINT_ERR_OVER_SP	10017	joint{} error: over speed
JOINT_ERR_OVER_ACC	10018	joint{} error: over accelerate
JOINT_ERR_TRACE	10019	joint{} error: trace accuracy
JOINT_ERR_TAG_POS_OVER	10020	joint{} error: target position out of range

下一页继续

表 3: 错误码汇总表

错误名称	错误码	说明
JOINT_ERR_TAG_SP_OVER	10021	joint{} error: target speed out of range
JOINT_ERR_COLLISION	10022	joint{} error: collision
TOOL_FLASH_VERIFY_FAILED	2010001	Flash write verify failed
TOOL_PROGRAM_CRC_FAILED	2010002	Program flash checksum failed during bootloading
TOOL_PROGRAM_CRC_FAILED2	2010003	Program flash checksum failed at runtime
TOOL_ID_UNDIFINED	2010004	Tool ID is undefined
TOOL_ILLEGAL_BL_CMD	2010005	Illegal bootloader command
TOOL_FW_WRONG	2010006	Wrong firmware at the joint
TOOL_HW_INVALID	2010007	Invalid hardware revision
TOOL_SHORT_CURCUIT_H	2020001	Short circuit detected on Digital Output: {} high side
TOOL_SHORT_CURCUIT_L	2020002	Short circuit detected on Digital Output: {} low side
TOOL_AVERAGE_CURR_HIGH	2020003	10 second Average tool IO Current of {} A is outside of the allowed range.
TOOL_POWER_PIN_OVER_CURR	2020004	Current of {} A on the POWER pin is outside of the allowed range.
TOOL_DOUT_PIN_OVER_CURR	2020005	Current of {} A on the Digital Output pins is outside of the allowed range.
TOOL_GROUND_PIN_OVER_CURR	2020006	Current of {} A on the ground pin is outside of the allowed range.
TOOL_RX_FRAMING	2030001	RX framing error
TOOL_RX_PARITY	2030002	RX Parity error
TOOL_48V_LOW	2040001	48V input is too low
TOOL_48V_HIGH	2040002	48V input is too high

下一页继续

表 3: 错误码汇总表

错误名称	错误码	说明
PEDSTRAL_PKG_LOST	3010001	Lost package from pedstral
IFB_ERR_ROBOTTYPE	20001	robot error type! 机械臂类型错误
IFB_ERR_ADXL_SENS	20002	adxl sensor error! 加速度计芯片错误
IFB_ERR_EN_LINE	20003	encoder line error! 编码器线数错误
IFB_ERR_ENTER_HDG_MODE	20004	robot enter hdg mode! 进入拖动示教模式错误
IFB_ERR_EXIT_HDG_MODE	20005	robot exit hdg mode! 退出拖动示教模式错误
IFB_ERR_MAC_DATA_BREAK	20006	mac data break! MAC 数据中断错误
IFB_ERR_DRV_FIRMWARE_VERSION	20007	driver firmware version error! 驱动器版本错误（关节固件版本不一致）
INIT_ERR_EN_DRV	20008	driver enable failed! 机械臂初始化使能驱动器失败
INIT_ERR_EN_AUTO_BACK	20009	driver enable auto back failed! 机械臂初始化使能自动回应失败
INIT_ERR_EN_CUR_LOOP	20010	driver enable current loop failed! 机械臂初始化使能电流环失败
INIT_ERR_SET_TAG_CUR	20011	driver set target current failed! 机械臂初始化设置目标电流失败
INIT_ERR_RELEASE_BRAKE	20012	driver release brake failed! 机械臂初始化释放刹车失败
INIT_ERR_EN_POS_LOOP	20013	driver enable postion loop failed! 机械臂初始化使能位置环失败

下一页继续

表 3: 错误码汇总表

错误名称	错误码	说明
INIT_ERR_SET_MAX_ACC	20014	set max accelerate failed! 机械臂初始化设置最大加速度失败
SAFETY_ERR_PROTECTION_STOP_TIMEOUT	20015	protection stop time-out! 机械臂保护停止超时
SAFETY_ERR_REDUCED_MODE_TIMEOUT	20016	reduced mode time-out! 机械臂减速模式超时
SYS_ERR_MCU_COM	20017	robot system error:mcu communication error!
SYS_ERR_RS485_COM	20018	robot system error:RS485 communication error!
IFB_ERR_DISCONNECTED	20019	Interface board may be disconnected. Please check connection between IPC and Interface board.
IFB_ERR_PAYLOAD_ERROR	20020	Payload error.
HW_SCB_SETUP_FAILED	5010001	Setup of Safety Control Board failed
HW_PKG_CNT_DISAGEE	5010002	Packet counter disagreements
HW_SCB_DISCONNECT	5010003	Connection to Safety Control Board lost
HW_SCB_PKG_LOST	5010004	Package lost from Safety Control Board
HW_SCB_CONN_INIT_FAILED	5010005	Ethernet connection initialization with Safety Control Board failed
HW_LOST_JOINT_PKG	5010006	Lost package from joint {}
HW_LOST_TOOL_PKG	5010007	Lost package from tool
HW_JOINT_PKG_CNT_DISAGREE	5010008	Packet counter disagreement in packet from joint {}
HW_TOOL_PKG_CNT_DISAGREE	5010009	Packet counter disagreement in packet from tool

下一页继续

表 3: 错误码汇总表

错误名称	错误码	说明
HW_JOINTS_FAULT	5020001	{} joint entered the Fault State
HW_JOINTS_VIOLATION	5020002	{} joint entered the Violation State
HW_TP_FAULT	5020003	Teach Pendant entered the Fault State
HW_TP_VIOLATION	5020004	Teach Pendant entered the Violation State
HW_JOINT_MV_TOO_FAR	5030001	{} joint moved too far before robot entered RUNNING State
HW_JOINT_STOP_NOT_FAST	5030002	Joint Not stopping fast enough
HW_JOINT_MV_LIMIT	5030003	Joint moved more than allowable limit
HW_FT_SENSOR_DATA_INVALID	5040001	Force-Torque Sensor data invalid
HW_NO_FT_SENSOR	5040002	Force-Torque sensor is expected, but it cannot be detected
HW_FT_SENSOR_NOT_CALIB	5040003	Force-Torque sensor is detected but not calibrated
HW_RELEASE_BRAKE_FAILED	5050000	Robot was not able to brake release, see log for details
HW_OVERCURR_SHUTDOWN	5060000	Overcurrent shutdown
HW_ENERGEY_SURPLUS	5070000	Energy surplus shutdown
HW_IDLE_POWER_HIGH	5080000	Idle power consumption to high
ROBOT_BE_PULLING	30001	Something is pulling the robot.
PSTOP_ELBOW_POS	30002	Protective Stop: Elbow position close to safety plane limits.
PSTOP_STOP_TIME	30003	Protective Stop: Exceeding user safety settings for stopping time.

下一页继续

表 3: 错误码汇总表

错误名称	错误码	说明
PSTOP_STOP_DISTANCE	30004	Protective Stop: Exceeding user safety settings for stopping distance.
PSTOP_CLAMP	30005	Protective Stop: Danger of clamping between the Robot′s lower arm and tool.
PSTOP_POS_LIMIT	30006	Protective Stop: Position close to joint limits
PSTOP_ORI_LIMIT	30007	Protective Stop: Tool orientation close to limits
PSTOP_PLANE_LIMIT	30008	Protective Stop: Position close to safety plane limits
PSTOP_POS_DEVIATE	30009	Protective Stop: Position deviates from path
JOINT_CHK_PAYLOAD	30010	Joint {}: Check payload, center of gravity and acceleration settings. Log screen may contain additional information.
PSTOP_SINGULARITY	30011	Protective Stop: Position in singularity.
PSTOP_CANNOT_MAINTAIN	30012	Protective Stop: Robot cannot maintain its position, check if payload is correct
PSTOP_WRONG_PAYLOAD	30013	Protective Stop: Wrong payload or mounting detected, or something is pushing the robot when entering Freedrive mode
PSTOP_JOINT_COLLISION	30014	Protective Stop: Collision detected by joint

下一页继续

表 3: 错误码汇总表

错误名称	错误码	说明
PSTOP_POS_DISAGREE	30015	Protective stop: The robot was powered off last time due to a joint position disagreement.
PSTOP_LARGE_MOVE	30016	Protective stop: Large movement of the robot detected while it was powered off. The joints were moved while it was powered off, or the encoders do not function
TARGET_POS_SUDDEN_CHG	30017	Sudden change in target position
SUDDEN_STOP	30018	Sudden stop.
ROBOT_STOP_ABNORMAL	30019	Robot has not stopped in the allowed reaction and braking time
PROG_INVALID_SETP	30020	Robot program resulted in invalid setpoint.
BLEND_INVALID_SETP	30021	Blending failed and resulted in an invalid setpoint.
APPROACH_SINGULARITY	30022	Robot approaching singularity −Acceleration threshold failed.
TSPEED_UNMATCH_POS	30023	Target speed does not match target position
INCONSIS_TPOS_SPD	30024	Inconsistency between target position and speed
JOINT_TSPD_UNMATCH_POS	30025	Target joint speed does not match target joint position change −Joint {}
FIELDBUS_INPUT_DISCONN	30026	Fieldbus input disconnected.
OPMODE_CHANGED	30027	Operational mode changed: {}

下一页继续

表 3: 错误码汇总表

错误名称	错误码	说明
NO_KIN_CALIB	30028	No Kinematic Calibration found (calibration.conf file is either corrupt or missing).
KIN_CALIB_UNMATCH_JOINT	30029	Kinematic Calibration for the robot does not match the joint(s).
KIN_CALIB_UNMATCH_ROBOT	30030	Kinematic Calibration does not match the robot.
JOINT_OFFSET_CHANGED	30031	Large movement of the robot detected while it was powered off. The joints were moved while it was powered off, or the encoders do not function
OFFSET_CHANGE_HIGH	30032	Change in offset is too high
JOINT_SPEED_LIMIT	30033	Close to joint speed safety limit.
TOOL_SPEED_LIMIT	30034	Close to tool speed safety limit.
MOMENTUM_LIMIT	30035	Close to momentum safety limit.
ROBOT_MV_STOP	30036	Robot is moving when in Stop Mode
HAND_PROTECTION	30037	Hand protection: Tool is too close to the lower arm: {} meter.
WRONG_SAFETYMODE	30038	Wrong safety mode: {}
SAFETYMODE_CHANGED	30039	Safety mode changed: {}
JOINT_ACC_LIMIT	30040	Close to joint acceleration safety limit
TOOL_ACC_LIMIT	30041	Close to tool acceleration safety limit
JOINT_TEMPERATURE_LIMIT	30042	Joint {} temperature too high(>{}℃)

下一页继续

表 3: 错误码汇总表

错误名称	错误码	说明
CONTROL_BOX_TEMPERATURE_LIMIT	30043	Control box temperature too high(>{}℃)
ROBOT_EMERGENCY_STOP	30044	Robot emergency stop
ROBOTMODE_CHANGED	30045	Robot mode changed: {}
ROBOTMODE_ERROR	30046	Wrong robot mode: {}
POSE_OUT_OF_REACH	30047	Target pose [{}] out of reach
TP_PLAN_FAILED	30048	Trajectory plan FAILED.
START_FORCE_FAILED	30049	Start force control failed, because force sensor does not exist.
OVER_SAFE_PLANE_LIMIT	30050	{} axis exceeds the safety plane limit (Move_type:{} id:{}).
POWERON_FAIL_VIOLATION	30051	Failed to power on because the robot safety mode is in violation
POWERON_FAIL_SYSTEMEMERGENCYSTOP	30052	Failed to power on because the robot safety mode is in system emergency stop
POWERON_FAIL_ROBOTEMERGENCYSTOP	30053	Failed to power on because the robot safety mode is in robot emergency stop
POWERON_FAIL_FAULT	30054	Failed to power on because the robot safety mode is in fault
STARTUP_FAIL_VIOLATION	30055	Failed to startup because the robot safety mode is in violation

下一页继续

表 3: 错误码汇总表

错误名称	错误码	说明
STARTUP_FAIL_SYSTEMEMERGENCYSTOP	30056	Failed to startup because the robot safety mode is in system emergency stop
STARTUP_FAIL_ROBOTEMERGENCYSTOP	30057	Failed to startup because the robot safety mode is in robot emergency stop
STARTUP_FAIL_FAULT	30058	Failed to startup because the robot safety mode is in fault
BACKDRIVE_FAIL_VIOLATION	30059	Failed to backdrive because the robot safety mode is in violation
BACKDRIVE_FAIL_SYSTEMEMERGENCYSTOP	30060	Failed to backdrive because the robot safety mode is in system emergency stop
BACKDRIVE_FAIL_ROBOTEMERGENCYSTOP	30061	Failed to backdrive because the robot safety mode is in robot emergency stop
BACKDRIVE_FAIL_FAULT	30062	Failed to backdrive because the robot safety mode is in fault
SETSIM_FAIL_VIOLATION	30063	Switch sim mode failed because the robot safety mode is in violation
SETSIM_FAIL_SYSTEMEMERGENCYSTOP	30064	Switch sim mode failed because the robot safety mode is in system emergency stop
SETSIM_FAIL_ROBOTEMERGENCYSTOP	30065	Switch sim mode failed because the robot safety mode is in robot emergency stop

下一页继续

表 3: 错误码汇总表

错误名称	错误码	说明
SETSIM_FAIL_FAULT	30066	Switch sim mode failed because the robot safety mode is in fault
FREEDRIVE_FAIL_VIOLATION	30067	Enable handguide mode failed because the robot safety mode is in violation
FREEDRIVE_FAIL_SYSTEMEMERGENCYSTOP	30068	Enable handguide mode failed because the robot safety mode is in system emergency stop
FREEDRIVE_FAIL_ROBOTEMERGENCYSTOP	30069	Enable handguide mode failed because the robot safety mode is in robot emergency stop
FREEDRIVE_FAIL_FAULT	30070	Enable handguide mode failed because the robot safety mode is in fault
UPFIRMWARE_FAIL_VIOLATION	30071	Firmware update failed because the robot safety mode is in violation
UPFIRMWARE_FAIL_SYSTEMEMERGENCYSTOP	30072	Firmware update failed because the robot safety mode is in system emergency stop
UPFIRMWARE_FAIL_ROBOTEMERGENCYSTOP	30073	Firmware update failed because the robot safety mode is in robot emergency stop
UPFIRMWARE_FAIL_FAULT	30074	Firmware update failed because the robot safety mode is in fault
SETPERSOSTENT_FAIL_VIOLATION	30075	Set persistent parameter failed because the robot safety mode is in violation

下一页继续

表 3: 错误码汇总表

错误名称	错误码	说明
SETPERSOSTENT_FAIL_SYSTEMEMERGENCYSTOP	30076	Set persistent parameter failed because the robot safety mode is in system emergency stop
SETPERSOSTENT_FAIL_ROBOTEMERGENCYSTOP	30077	Set persistent parameter failed because the robot safety mode is in robot emergency stop
SETPERSOSTENT_FAIL_FAULT	30078	Set persistent parameter failed because the robot safety mode is in fault
SETPERSOSTENT_FAIL_PARAM_ERR	30079	Set persistent parameter failed
ROBOT_CABLE_DISCONN	30080	Robot cable not connected
TP_TOO_SHORT	30081	The generated trajectory is ignored because it is too short
INV_KIN_FAIL	30082	Inverse kinematics solution failed. The target pose may be in a singular position or exceed the joint limits
FREEDRIVE_ENABLED	30083	Freedrive status changed to {}
TP_INV_FAIL_REFERENCE_JOINT_OUT_OF_LIMIT	30084	Inverse kinematics solution failed. Reference angle [{}] exceeds joint limit [{}].
TP_INV_FAIL_NO_SOLUTION	30085	Inverse kinematics solution failed. The reference angle [{}] and the target angle [{}] are used as parameters. there is no solution in the calculation of the inverse solution process.

下一页继续

表 3: 错误码汇总表

错误名称	错误码	说明
SERVO_FAIL_VIOLATION	30086	Switch servo mode failed because the robot safety mode is in violation
SERVO_FAIL_SYSTEMEMERGENCYSTOP	30087	Switch servo mode failed because the robot safety mode is in system emergency stop
SERVO_FAIL_ROBOTEMERGENCYSTOP	30088	Switch servo mode failed because the robot safety mode is in robot emergency stop
SERVO_FAIL_FAULT	30089	Switch servo mode failed because the robot safety mode is in fault
FREEDRIVE_FAIL_NO_RUNNING	30090	Enable handguide mode failed because the robot mode type is {}(not running)
RUNTIME_MACHINE_ERROR	30091	The state of the running machine is {}, not {}. {} function execution failed because the state is wrong.
RESUME_FAR_PAUSE_PT	30092	Cannot resume from joint position [{}].\nToo far away from paused point [{}].
PAYLOAD_LIGHTER_ERROR	30093	The payload setting is too small!
PAYLOAD_OVERLOAD_ERROR	30094	The payload setting is too large!
PAUSE_FAIL_NOT_POSITION_PLAN_MODE	30095	This motion does not support the pause function. The motion is stopping.
TP_PLAN_FAILED_CIRCULAR_WAYPOINTS_COINCIDE	30096	The planning failed because the three waypoints of the arc were determined to coincide.

下一页继续

表 3: 错误码汇总表

错误名称	错误码	说明
SERVO_WRONG_SAFETYMODE	30097	Switch servo mode failed because the robot safety mode is in {}.
SET_PERSTPARAM_WRONG_SAFETYMODE	30098	Set persistent parameter failed because the robot safety mode is in {}
SET_KINPARAM_WRONG_SAFETYMODE	30099	Set Kinematics Compensate parameters failed because the robot safety mode is in {}
SET_ROBOT_ZERO_WRONG_SAFETYMODE	30100	Set current joint angles to zero failed because the robot safety mode is in {}
UPFIRMWARE_WRONG_SAFETYMODE	30101	Firmware update failed because the robot safety mode is in {}
POWERON_WRONG_SAFETYMODE	30102	Failed to power on because the robot safety mode is in {}
STARTUP_WRONG_SAFETYMODE	30103	Failed to startup because the robot safety mode is in {}
BACKDRIVE_WRONG_SAFETYMODE	30104	Failed to backdrive because the robot safety mode is in system emergency stop
SETSIM_WRONG_SAFETYMODE	30105	Switch sim mode failed because the robot safety mode is in violation
FREEDRIVE_WRONG_SAFETYMODE	30106	Enable handguide mode failed because the robot safety mode is in system emergency stop
ARCS_MAX_ERROR_CODE	-1	Max error code