7. 接口函数详细说明

7.1 执行器接口

类名	功能描述	主要方法
AsioExecutor	异步IO执行器，管理后台工作线程处理通信和回调	`create()`，`get_io_context()`

- 异步执行器工厂方法

字段	说明
函数名	`create_asio_executor`
函数原型	`create_asio_executor(worker_count)`
功能概述	创建异步IO执行器，管理后台工作线程处理通信和回调
参数	`worker_count`：工作线程数量
返回值	`AsioExecutor`，异步IO执行器对象
备注	用于管理异步IO操作和后台任务

代码示例：

```
executor = ah.create_asio_executor(8)
```

- 异步执行器IO上下文获取接口

字段	说明
函数名	`get_io_context`
函数原型	`get_io_context()`

字段	说明
功能概述	获取异步执行器管理的IO上下文对象
参数	无
返回值	`IOContext`，IO上下文对象
备注	用于传递给其他组件进行异步IO操作

代码示例：

```
io_context = executor.get_io_context()
```

7.2 机械臂控制接口

类名	功能描述	主要方法
单Play	6轴机械臂控制类，集成电机控制功能	`create()`, `init()`, `uninit()`, `ping()`, `enable()`, `disable()`, `set_zero()`, `reset_error()`, `csv()`, `pvt()`, `mit()`, `get_param()`, `set_param()`, `persist_param()`, `state()`, `params()`
PlayWithEEF	7轴机械臂控制类，集成电机和末端执行器控制	`create()`, `init()`, `uninit()`, `ping()`, `enable()`, `disable()`, `set_zero()`, `reset_error()`, `csv()`, `pvt()`, `mit()`, `get_param()`, `set_param()`, `persist_param()`, `state()`, `params()`

代码示例：

```
arm = ah.Play.create(
    ah.MotorType.OD, ah.MotorType.OD, ah.MotorType.OD,
    ah.MotorType.DM, ah.MotorType.DM, ah.MotorType.DM,
    ah.EEFType.NA, ah.MotorType.NA
)
```

- PlayWithEEF 类工厂方法

字段	说明
函数名	`create`
函数原型	`@staticmethod PlayWithEEF.create(m1, m2, m3, m4, m5, m6, eef_type, eef_motor)`
功能概述	创建7轴机械臂控制实例，集成电机和末端执行器控制功能
参数	`m1-m6`：6个关节的电机类型 `eef_type`：末端执行器类型 (E2/G2) `eef_motor`：末端执行器电机类型
返回值	`PlayWithEEF`，7轴机械臂控制对象实例
备注	适用于带末端执行器的机械臂配置

代码示例：

```
arm = ah.PlayWithEEF.create(
    ah.MotorType.EC, ah.MotorType.EC, ah.MotorType.EC,
    ah.MotorType.EC, ah.MotorType.EC, ah.MotorType.EC,
    ah.EEFType.E2, ah.MotorType.EC
)
```

- 机械臂初始化接口

字段	说明
函数名	`init`
函数原型	`init(io_context, interface, spin_freq)`
功能概述	初始化机械臂，建立与硬件的通信连接
参数	`io_context`：异步IO上下文对象 `interface`：通信接口名称 ("can0") `spin_freq`：控制频率 (Hz)
返回值	`bool`，初始化成功返回 True，失败返回 False

字段	说明
备注	必须在其他操作前调用，控制频率影响控制精度和响应速度

代码示例：

```
executor = ah.create_asio_executor(8)
io_context = executor.get_io_context()
success = arm.init(io_context, "can0", 250)
```

- 机械臂反初始化接口

字段	说明
函数名	`uninit`
函数原型	`uninit()`
功能概述	反初始化机械臂，释放通信资源
参数	无
返回值	`bool`，反初始化成功返回 True，失败返回 False
备注	程序结束前应调用此方法释放资源

代码示例：

```
arm.disable()
arm.uninit()
```

- 机械臂使能接口

字段	说明
函数名	`enable`
函数原型	`enable()`
功能概述	使能机械臂，依次使能所有电机和末端执行器

字段	说明
参数	无
返回值	`bool`，所有组件使能成功返回 True，否则返回 False
备注	使能后机械臂可以接收控制命令

代码示例：

```
if arm.enable():
    print("Arm enabled successfully")
```

- 机械臂失能接口

字段	说明
函数名	`disable`
函数原型	`disable()`
功能概述	失能机械臂，依次失能所有电机和末端执行器
参数	无
返回值	`bool`，所有组件失能成功返回 True，否则返回 False
备注	失能后机械臂停止运动，进入安全状态

代码示例：

```
arm.disable()
```

- 速度控制接口

字段	说明
函数名	`csv`
函数原型	`csv(vel)`

字段	说明
功能概述	发送循环同步速度控制帧，控制各关节目标速度
参数	`vel`：各关节目标速度列表 (rad/s)
返回值	`bool`，控制命令发送成功返回 True，否则返回 False
备注	需要持续调用以维持连续控制，适用于速度控制场景

代码示例：

```
velocities = [0.1, 0.0, 0.0, 0.0, 0.0, 0.0]
arm.csv(velocities)
```

- 位置控制接口

字段	说明
函数名	`pvt`
函数原型	`pvt(pos, max_vel=None, max_eff=None)`
功能概述	发送位置-速度-时间控制帧，控制各关节目标位置，带速度和力矩限制
参数	`pos`：各关节目标位置列表 (rad) `max_vel`：最大速度限制列表 (rad/s) `max_eff`：最大力矩限制列表
返回值	`bool`，控制命令发送成功返回 True，否则返回 False
备注	适用于轨迹跟踪和路径规划，支持速度和力矩限制

代码示例：

```
positions = [0.5, 0.0, 0.0, 0.0, 0.0, 0.0]
arm.pvt(positions)
```

- MIT控制接口

字段	说明
函数名	`mit`
函数原型	`mit(pos, vel, eff, mit_kp, mit_kd)`
功能概述	发送MIT控制帧，实现位置-速度-力矩复合控制，支持前馈力矩和PID参数调节
参数	`pos`：目标位置列表 (rad) `vel`：目标速度列表 (rad/s) `eff`：前馈力矩列表 `mit_kp`：比例增益列表 `mit_kd`：微分增益列表
返回值	`bool`，控制命令发送成功返回 True，否则返回 False
备注	适用于精确力控制和阻抗控制，支持PID参数调节

代码示例：

```
pos = [0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
vel = [0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
eff = [0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
kp = [100.0, 100.0, 100.0, 100.0, 100.0, 100.0]
kd = [10.0, 10.0, 10.0, 10.0, 10.0, 10.0]
arm.mit(pos, vel, eff, kp, kd)
```

- 参数设置接口

字段	说明
函数名	`set_param`
函数原型	`set_param(name, value)`
功能概述	发送参数设置请求，设置机械臂参数值
参数	`name`：参数名称字符串 `value`：参数值
返回值	`bool`，请求发送成功返回 True，否则返回 False

字段	说明
备注	设置后响应帧包含已设置的参数，无需再次调用 get_param

代码示例：

```
arm.set_param("arm.control_mode", ah.MotorControlMode.MIT)
```

- 状态获取接口

字段	说明
函数名	`state`
函数原型	`state()`
功能概述	获取机械臂当前状态，包括位置、速度、力矩等信息
参数	无
返回值	`ArmState`，包含 is_valid、pos、vel、eff 字段的状态对象
备注	线程安全，可多线程调用，返回状态快照

代码示例：

```
state = arm.state()
if state.is_valid:
    print(f"Position: {state.pos[0]}")
```

- ping 接口

字段	说明
函数名	`ping`
函数原型	`ping()`
功能概述	向机械臂发送ping请求，获取最新状态信息

字段	说明
参数	无
返回值	`bool`，ping成功返回 True，失败返回 False
备注	用于主动查询机械臂当前状态，更新内部状态缓存

代码示例：

```
if arm.ping():
    state = arm.state()
    print("Ping successful, state updated")
```

- set_zero 接口

字段	说明
函数名	`set_zero`
函数原型	`set_zero()`
功能概述	设置机械臂当前位置为零点位置
参数	无
返回值	`bool`，设置成功返回 True，失败返回 False
备注	用于重新校准机械臂零点位置，影响后续位置控制

- reset_error 接口

字段	说明
函数名	`reset_error`
函数原型	`reset_error()`
功能概述	重置机械臂错误状态，清除故障标志

字段	说明
参数	无
返回值	`bool`，错误重置成功返回 True，失败返回 False
备注	在机械臂出现故障后调用，恢复正常工作状态

- get_param 接口

字段	说明
函数名	`get_param`
函数原型	`get_param(name)`
功能概述	获取机械臂参数值
参数	`name`：参数名称字符串
返回值	`bool`，请求发送成功返回 True，否则返回 False
备注	异步获取参数，实际值通过响应帧返回

- persist_param 接口

字段	说明
函数名	`persist_param`
函数原型	`persist_param(name, value)`
功能概述	持久化保存参数到非易失存储器
参数	`name`：参数名称字符串 `value`：参数值
返回值	`bool`，持久化请求发送成功返回 True，否则返回 False

字段	说明
备注	参数将保存在Flash中，断电不丢失

- params 接口

字段	说明
函数名	`params`
函数原型	`params()`
功能概述	获取机械臂所有参数信息
参数	无
返回值	`dict`，包含所有参数名称和值的字典
备注	返回当前已缓存的参数信息，可能不是最新值

- set_param (MotorControlMode重载) 接口

字段	说明
函数名	`set_param`
函数原型	`set_param(name, value)`
功能概述	设置电机控制模式参数的重载版本
参数	`name`: 参数名称字符串 `value`: MotorControlMode枚举值
返回值	`bool`，请求发送成功返回 True，否则返回 False
备注	专用于设置电机控制模式，支持POSITION、VELOCITY、TORQUE等模式

- PlayWithEEF create 接口 (详细配置)

字段	说明
函数名	`create`
函数原型	`@staticmethod PlayWithEEF.create(m1, m2, m3, m4, m5, m6, eef_type, motor_type)`
功能概述	创建7轴机械臂控制实例，支持多种电机和末端执行器配置组合
参数	`m1-m6`：6个关节的电机类型 (OD/DM/ODM/EC) `eef_type`：末端执行器类型 (G2/E2) `motor_type`：末端执行器电机类型 (DM/ODM/OD)
返回值	`PlayWithEEF`，7轴机械臂控制对象实例
备注	支持多种已验证配置组合， (OD,OD,OD,DM,DM,DM,G2,DM)等

7.3 末端执行器接口

类名	功能描述	主要方法
EEF0	零自由度末端执行器控制类，支持空配置	`create()`，`init()`，`uninit()`，`ping()`，`enable()`，`disable()`，`set_zero()`，`reset_error()`，`pvt()`，`mit()`，`get_param()`，`set_param()`，`persist_param()`，`state()`，`params()`
EEF1	单自由度末端执行器控制类，支持G2夹爪	`create()`，`init()`，`uninit()`，`ping()`，`enable()`，`disable()`，`set_zero()`，`reset_error()`，`pvt()`，`mit()`，`get_param()`，`set_param()`，`persist_param()`，`state()`，`params()`

- 末端执行器工厂方法

字段	说明
函数名	`create`
函数原型	`@staticmethod EEF.create(eef_type, motor_type)`

字段	说明
功能概述	创建末端执行器控制实例，通过指定末端执行器类型和电机类型来创建控制对象
参数	`eef_type`：末端执行器类型 (ah.EEFType.G2/E2/NA) `motor_type`：驱动电机类型 (ah.MotorType.DM/ODM/OD)
返回值	`EEF`，末端执行器控制对象实例
备注	支持G2+DM、G2+ODM等组合

代码示例：

```
eef = ah.EEF.create(ah.EEFType.G2, ah.MotorType.DM)
```

- 末端执行器初始化接口

字段	说明
函数名	`init`
函数原型	`init(io_context, interface, spin_freq)`
功能概述	初始化末端执行器，建立与硬件的通信连接
参数	`io_context`：异步IO上下文对象 `interface`：通信接口名称 ("can0") `spin_freq`：控制频率 (Hz)
返回值	`bool`，初始化成功返回 True，失败返回 False
备注	必须在其他操作前调用

代码示例：

```
success = eef.init(io_context, "can0", 250)
```

- 末端执行器释放资源接口

字段	说明
函数名	`uninit`
函数原型	`init(io_context, interface, spin_freq)`
功能概述	初始化末端执行器，建立与硬件的通信连接
参数	`io_context`：异步IO上下文对象 `interface`：通信接口名称 (　"can0") `spin_freq`：控制频率 (Hz)
返回值	`bool`，初始化成功返回 True，失败返回 False
备注	必须在其他操作前调用

代码示例：

```
success = eef.init(io_context, "can0", 250)
```

- 末端执行器ping接口

字段	说明
函数名	`ping`
函数原型	`ping()`
功能概述	向末端执行器发送ping请求，获取最新状态信息
参数	无
返回值	`bool`，ping成功返回 True，失败返回 False
备注	用于主动查询末端执行器当前状态，更新内部状态缓存

代码示例：

```
if eef.ping():
    state = eef.state()
    print("Ping successful, state updated")
```

- 末端执行器enable接口

字段	说明
函数名	`enable`
函数原型	`enable()`
功能概述	使能末端执行器
参数	无
返回值	`bool`，使能成功返回 True，否则返回 False
备注	使能后末端执行器可以接收控制命令

代码示例：

```
if eef.enable():
    print("EEF enabled successfully")
```

- 末端执行器disable 接口

字段	说明
函数名	`disable`
函数原型	`disable()`
功能概述	失能末端执行器
参数	无
返回值	`bool`，失能成功返回 True，否则返回 False
备注	失能后末端执行器停止运动，进入安全状态

代码示例：

```
eef.disable()
```

- 末端执行器位置置零接口

字段	说明
函数名	`set_zero`
函数原型	`set_zero()`
功能概述	将末端执行器的位置参考点设置为零
参数	无
返回值	无
备注	调用后末端执行器当前位置将被视为位置参考零点，用于校准校准位置基准

代码示例：

```
eef.set_zero()  # 将末端执行器当前位置设为零点
```

- 末端执行器reset_error接口

字段	说明
函数名	`reset_error`
函数原型	`reset_error()`
功能概述	重置末端执行器错误状态，清除故障标志
参数	无
返回值	`bool`，错误重置成功返回 True，失败返回 False
备注	在末端执行器出现故障后调用，恢复正常工作状态

代码示例：

```
if eef.state().has_error:
    eef.reset_error()
```

- 末端执行器PVT控制接口

字段	说明
函数名	`pvt`
函数原型	`pvt(cmd)`
功能概述	发送PVT控制命令到末端执行器
参数	`cmd`：末端执行器控制命令对象，包含位置、速度、力矩等信息
返回值	`bool`，控制命令发送成功返回 True，否则返回 False
备注	适用于轨迹跟踪和路径规划

代码示例：

```
cmd = ah.EEFCommand()
cmd.pos[0] = 0.05
eef.pvt(cmd)
```

- 末端执行器MIT接口

字段	说明
函数名	`mit`
函数原型	`mit(cmd)`
功能概述	发送MIT命令到末端执行器
参数	`cmd`：EEFState命令对象
返回值	`bool`，命令发送成功返回 True，否则返回 False

字段	说明
备注	用于MIT控制模式，支持阻抗控制

代码示例：

```
cmd = ah.EEFState()
cmd.pos = [0.0]
cmd.vel = [0.0]
cmd.eff = [0.0]
cmd.kp = [0.0]
cmd.kd = [0.0]
eef.mit(cmd)
```

- get_param 接口

字段	说明
函数名	`get_param`
函数原型	`get_param(name)`
功能概述	获取末端执行器参数值
参数	`name`：参数名称字符串
返回值	`bool`，请求发送成功返回 True，否则返回 False
备注	异步获取参数，实际值通过响应帧返回

代码示例：

```
eef.get_param("eef.control_mode")
```

- 末端执行器set_param接口

字段	说明
函数名	`set_param`
函数原型	`set_param(name, value)`

字段	说明
功能概述	设置末端执行器参数值
参数	`name`: 参数名称字符串 `value`: 参数值
返回值	`bool`，请求发送成功返回 True，否则返回 False
备注	设置后响应帧包含已设置的参数，无需再次调用 get_param

代码示例：

```
eef.set_param("eef.control_mode", ah.MotorControlMode.POSITION)
```

- 末端执行器set_param(MotorControlMode)接口

字段	说明
函数名	`set_param`
函数原型	`set_param(name, value)`
功能概述	设置电机控制模式参数
参数	`name`: 参数名称字符串 `value`: MotorControlMode枚举值
返回值	`bool`，请求发送成功返回 True，否则返回 False
备注	专用于设置电机控制模式，支持POSITION、VELOCITY、TORQUE等模式

代码示例：

```
eef.set_param("motor.control_mode", ah.MotorControlMode.POSITION)
```

- 末端执行器状态获取接口

字段	说明

字段	说明
函数名	`state`
函数原型	`state()`
功能概述	获取末端执行器当前状态，包括位置、速度、力矩等信息
参数	无
返回值	`EEFState`，包含 is_valid、pos、vel、eff 字段的状态对象
备注	线程安全，可多线程调用，返回状态快照

代码示例：

```
state = eef.state()
if state.is_valid:
    print(f"Position: {state.pos[0]}")
```

- 末端执行器params接口

字段	说明
函数名	`params`
函数原型	`params()`
功能概述	获取末端执行器所有参数信息
参数	无
返回值	`MotorParams`，包含所有参数名称和值的MotorParams对象
备注	返回当前已缓存的参数信息，可能不是最新值

代码示例：

```
parameters = eef.params()
print(f"Control mode: {parameters.control_mode}")
```

7.4 灵巧手

7.4.1 灵巧手接口

- **DexterousHand灵巧手工厂方法**

字段	说明
函数名	`create_dexterous_hand`
函数原型	`def create_dexterous_hand(id: typing.SupportsInt, type: DexterousHandTypes) -> DexterousHand:`
功能概述	工厂方法创建灵巧手实例，通过指定设备ID和设备类型来创建控制对象
参数	`id`：设备ID `type`：类型
返回值	`DexterousHand`，灵巧手实例
备注	创建失败时返回nullptr

代码示例：

```
hand : DexterousHand = ah.create_dexterous_hand(hand_id, hand_type)
```

- **DexterousHand灵巧手初始化接口**

字段	说明
函数名	`init`
函数原型	`def init(self, io_context: Stub, interface: str, spin_freq: typing.SupportsInt) -> bool:`
功能概述	初始化灵巧手，建立与硬件的通信连接
参数	`io_context`：异步IO执行器指针 `interface`：通信接口名称 ("can0") `spin_freq`：控制循环频率

字段	说明
返回值	`bool`，初始化成功返回 true，失败返回 false
备注	必须在其他操作前调用

代码示例：

```
success : bool = hand.init(io_context, can_interface, 250)
```

- **DexterousHand灵巧手状态更新接口**

字段	说明
函数名	`update_state`
函数原型	`def update_state(self) -> bool:`
功能概述	更新灵巧手当前状态，包括位置、速度、力矩等信息。**状态的更新可能会有延迟。**
参数	无
返回值	`bool`，初始化成功返回 true，失败返回 false
备注	需要在获取状态前调用

代码示例：

```
hand.update_state()
```

- **DexterousHand灵巧手状态获取接口**

字段	说明
函数名	`state`
函数原型	`def state(self) -> HandState:`
功能概述	获取灵巧手当前状态，包括位置、速度、力矩等信息

字段	说明
参数	无
返回值	`HandState`，包含灵巧手状态信息的结构体
备注	线程不安全，避免多线程调用，返回状态快照

代码示例：

```
state : HandState= hand.state()
print(list(state.forces))
print(list(state.positions))
print(list(state.velocities))
```

- **DexterousHand灵巧手位置控制接口**

字段	说明
函数名	`set_pos`
函数原型	`def set_pos(self, cmd: HandState) -> bool:`
功能概述	控制灵巧手以cmd中的参数状况移动到指定位置
参数	`cmd`：`HandState`参数结构体
返回值	`bool`，初始化成功返回 true，失败返回 false
备注	线程不安全，避免多线程调用，返回状态快照

代码示例：

```
hand_cmd = ah.HandState()
hand_cmd.positions = [1000] * 6
hand_cmd.velocities = [1000] * 6
hand.set_pos(hand_cmd)
```

- **DexterousHand灵巧手参数设置接口**

字段	说明

字段	说明
函数名	`set_param`
函数原型	`def set_param(self, name: str, value: ParamValue) -> bool:`
功能概述	设置灵巧手参数
参数	`name`：参数名称 `value`：参数值
返回值	`bool`，初始化成功返回 true，失败返回 false
备注	线程不安全，避免多线程调用，返回状态快照

7.4.2 各型号灵巧手默认出厂id（仅供参考）

** `Hand Type`	`Enum Value (Hex)`	`Default Device ID (Hex)`
INS_RH56DFX	0x00	0x01
INS_RH56BFX	0x01	0x01
INS_RH56E2	0x02	0x01
INS_RH56F1	0x03	0x01
BRAINCO_REVO2	0x21	0x7E (Left) / 0x7F (Right)
ROH_LITES001	0x40	0x02
ROH_A002	0x41	0x02
LINKERBOT_O6	0x51	0x28 (Left) / 0x27 (Right)

7.4.3 参数映射表

不同型号的机械手支持的配置参数各不相同。下表总结了当前支持的参数及其访问方式：

手型	参数名	访问方式	类型	描述
BrainCo Revo2	`hand_side`	R	`uint16`	手的方向（0=左手，1=右手）
	`unit_mode`	R/W	`0 / 1`	工作模式
	`id_485`	R/W	`uint16`	RS-485 节点 ID
	`baudrate`	R/W	`uint8`	通信波特率（115200 / 57600 / 19200 / 460800 / 1M / 2M / 5M）
INS RH56BFX	`hand_id`	R	`uint8`	设备 ID
	`baudrate`	R/W	`uint8`	CAN 波特率
	`current`	R	`uint16[6]`	各通道电流
	`pos_act`	R	`uint16[6]`	实际关节位置反馈
	`clear_error`	W	`0 / 1`	清除错误标志
	`force_calibration`	W	`0 / 1`	触发力校准过程
INS RH56DFX	`hand_id`	R	`uint16`	设备 ID
	`baudrate`	R/W	`uint8`	CAN 波特率
	`current`	R	`uint16[6]`	各通道电流
	`pos_act`	R	`uint16[6]`	实际关节位置反馈
	`clear_error`	W	`0 / 1`	清除错误标志

手型	参数名	访问方式	类型	描述
	`force_calibration`	W	`0 / 1`	触发力校准过程
INS RH56F1	`hand_id`	R	`uint16`	设备 ID
	`baudrate`	R/W	`uint32`	CAN 波特率
	`current`	R	`uint16[6]`	各通道电流
	`pos_act`	R	`uint16[6]`	实际关节位置反馈
	`clear_error`	W	`0 / 1`	清除错误标志
	`force_calibration`	W	`0 / 1`	触发力校准过程
INS RH56E2	`hand_id`	R	`uint16`	设备 ID
	`baudrate`	R/W	`uint8`	CAN 波特率
	`current`	R	`uint16[6]`	各通道电流
	`pos_act`	R	`uint16[6]`	实际关节位置反馈
	`clear_error`	W	`0 / 1`	清除错误标志
	`force_calibration`	W	`0 / 1`	触发力校准过程
ROH A002	`hand_id`	R/W	`uint8`	CAN 节点 ID

> ⚠ 对于不支持的参数，接口会忽略请求或返回 `false`。