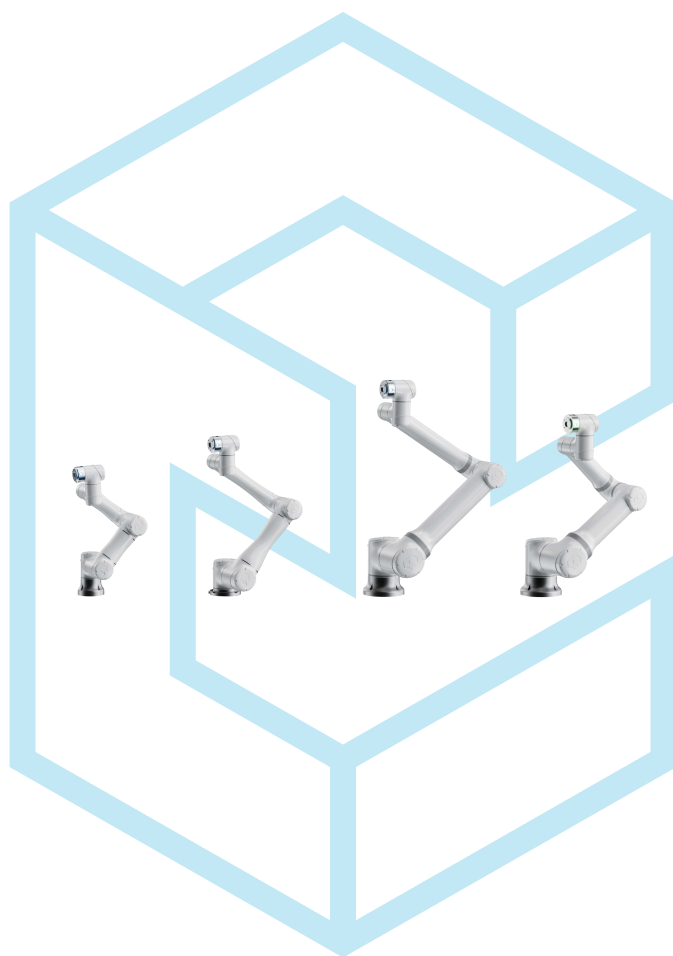


ELITE ROBOTS EC系列

编程手册



通信协议手册

苏州艾利特机器人有限公司

2025-07-07

版本: Ver3.20.2

目录

1	安全注意事项	1
1.1	安全警示标志	1
2	Modbus	4
2.1	MODBUS 协议简介	4
2.1.1	Modbus RTU	5
2.1.2	Modbus TCP	6
2.1.3	Modbus 标准功能码	7
2.2	Modbus 功能的系统配置方法	7
2.2.1	Modbus 配置文件	7
2.3	Modbus 功能的程序应用	9
2.3.1	软元件地址	9
2.3.2	程序范例	10
2.3.2.1	Modbus RTU 通信报文范例	10
2.3.2.2	Modbus TCP 通讯报文范例	11
2.3.3	机器人端 M 变量的操作	13
2.4	获取机器人信息	13
2.4.1	简介	13
2.4.2	获取机器人当前关节角度	14
2.4.3	获取机器人当前关节角速度	14
2.4.4	获取机器人当前 TCP 位置	14
2.4.5	保存安全参数使能状态	14
2.4.6	保存碰撞检测使能状态	15
2.4.7	获取机器人当前 TCP 速度	15
2.4.8	获取系统数字输入口状态	15

2.4.9	获取系统数字输出口状态	16
2.4.10	获取系统模拟量输入口状态	16
2.4.11	获取系统模拟量输出口状态	16
2.4.12	获取系统时间戳	17
2.4.13	获取机器人 tcp 加速度	17
2.4.14	获取机器人关节加速度	17
2.4.15	获取安全限制功率参数	18
2.4.16	获取安全限制动量参数	18
2.4.17	获取安全限制工具力参数	18
2.4.18	获取安全限制肘部力参数	18
2.4.19	获取安全限制速度百分比参数	19
2.4.20	获取碰撞检测灵敏度参数	19
2.4.21	其他通用寄存器	19
2.5	实例	19
3	Profinet	23
3.1	寄存器访问方式	23
3.1.1	Profinet 主站	23
3.1.2	SDK 接口	24
3.1.3	Lua 函数	24
3.2	Profinet 配置	25
3.2.1	启用 Profinet 功能	25
3.2.2	设置 Profinet	25
3.3	获取机器人信息	30
3.3.1	简介	30
3.3.1.1	获取机器人当前位姿	31
3.3.1.2	获取机器人当前 TCP 速度	31
3.3.1.3	获取机器人 TCP 加速度	31
3.3.1.4	获取机器人当前关节角度	31

3.3.1.5	获取机器人当前关节角速度	32
3.3.1.6	获取机器人关节加速度	32
3.3.1.7	获取系统数字输入口状态	32
3.3.1.8	获取系统数字输出口状态	32
3.3.1.9	获取系统模拟量输入值	33
3.3.1.10	获取系统模拟量输出值	33
3.3.1.11	获取碰撞检测使能状态	33
3.3.1.12	获取碰撞检测灵敏度参数	34
3.3.1.13	获取安全参数使能状态	34
3.3.1.14	获取安全限制功率参数	34
3.3.1.15	获取安全限制动量参数	34
3.3.1.16	获取安全限制工具力参数	35
3.3.1.17	获取安全限制肘部力参数	35
3.3.1.18	获取安全限制速度百分比参数	35
3.3.1.19	获取系统时间戳	35
4	Ethernet/IP接口与协议	37
4.1	Ethernet/IP简介	37
4.1.1	相关术语	37
4.1.2	Ethernet/IP寄存器访问方式	37
4.2	Ethernet/IP寄存器访问方式	50
4.2.1	Ethernet/IP主站设备	50
4.2.1.1	Ethernet/IP主站设备	50
4.2.2	SDK接口	50
4.2.3	Lua函数	50

第 1 章 安全注意事项

使用本系统前，请务必熟读并全部掌握本说明书和其他附属资料，在熟知全部设备知识、安全知识及注意事项后再开始使用。

1.1 安全警示标志

本说明书中的安全注意事项分为“危险”、“注意”、“强制”、“禁止”四类分别记载。

危险



此安全信息表示危险情况，如果不避免，将导致死亡或严重伤害。

提醒



此安全信息表示危险情况，如果不避免，可能导致轻微或中度伤害。

强制



此安全信息表示必须遵守的事项。

禁止



此安全信息表示禁止的事项。

需要说明的，即使是“注意”所记载的内容，也会因情况不同而产生严重的后果，因此任何一条注意事项都极为重要，请务必严格遵守。甚至在有些地方就连“注意”或“危险”等内容都未标记，也是用户必须严格遵守的事项。

危险

1. 操作机器人前，按下示教器上的急停键，并确认伺服主电源被切断，电机处于失电并抱闸状态。

紧急情况下，若不能及时制动机器人，则可能引发人身伤害或设备损坏事故。

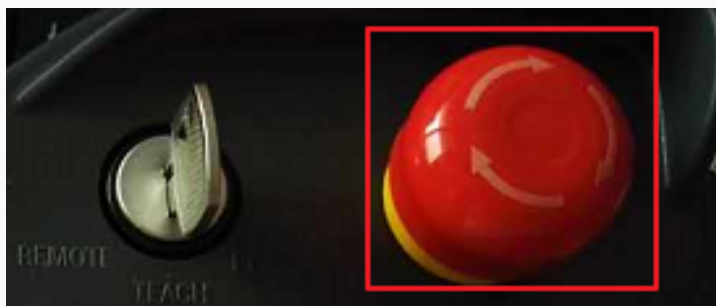


图 1-1：紧急停机按钮

2. 解除急停后再接通伺服电源时，要解除造成急停的事故后再接通伺服电源。

由于误操作造成的机器人动作，可能引发人身伤害事故。

3. 在机器人动作范围内示教时，请遵守以下原则：

保持从正面观看机器人。

严格遵守操作步骤。

考虑机器人突然向自己所处方位运动时的应变方案。

确保设置躲避场所，以防万一。

由于误操作造成的机器人动作，可能引发人身伤害事故。

4. 进行以下作业时，请确认机器人的动作范围内没人，并且操作者处于安全位置操作：

机器人控制柜接通电源时。

用示教器操作机器人时。

试运行。

自动运行时。

不慎进入机器人动作范围内或与机器人发生接触，都有可能引发人身伤害事故。另外，发生异常时，请立即按下急停键。

提醒

1. 操作机器人必须确认：
操作人员是否接受过机器人操作的相关培训。
对机器人的运动特性有足够的认识。
对机器人的危险性有足够的了解。
未酒后上岗。
未服用影响神经系统、反应迟钝的药物。
2. 进行机器人示教作业前要检查以下事项，有异常则应及时修理或采取其它必要措施。
机器人动作有无异常。
原点是否校准正确。
与机器人相关联的外部辅助设备是否正常。
3. 示教器用完后须放回原处，并确保旋转牢固。
如不慎将示教器放在机器人、夹具或地上，当机器人运动时，示教器可能与机器人或夹具发生碰撞，从而引发人身伤害或设备损坏事故。
防止示教器意外跌落造成机器人误动作，从而引发人身伤害或设备损坏事故。

强制

安全操作规程：

1. 所有机器人系统的操作者，都应该参加本系统的培训，学习安全防护措施和使用机器人的功能。
2. 在开始运行机器人之前，确认机器人和外围设备周围没有异常或者危险状况。
3. 在进入操作区域内工作前，即使机器人没有运行，也要关掉电源，或者按下急停按钮。
4. 当在机器人工作区编程时，设置相应看守，保证机器人能在紧急情况，迅速停车。
5. 示教和点动机器人时不要带手套操作，点动机器人时要尽量采用低速操作，遇异常情况时可有效控制机器人停止。
6. 必须知道机器人控制器和外围控制设备上的紧急停止按钮的位置，以便在紧急情况下能准确的按下这些按钮。
7. 永远不要以为机器人处于停止状态时其程序就已经完成。因为此时机器人很有可能是在等待让他继续运动的输入信号。

第 2 章 Modbus

2.1 MODBUS 协议简介

Modbus 是一种串行通信协议，是 Modicon（现在的施耐德电气 Schneider Electric）于 1979 年为使用可编程控制器（PLC）通信而发表。Modbus 已经成为工业领域通信协议的业界标准（De facto），并且现在是工业电子设备之间常用的连接方式。

ERC-G200 控制系统支持 Modbus RTU 和 Modbus TCP 两种通信方式。Modbus 协议的帧格式如图 2-1 所示。

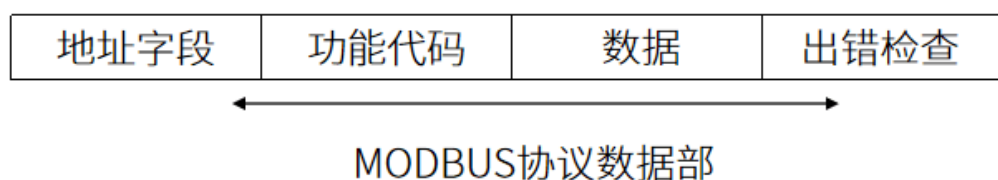


图 2-1：帧格式

Modbus 协议的帧规格的详细内容如下所示。

表 2-1. 帧规格的细节内容

区域名	内容
地址字段	<p>[主站向从站发送请求报文时]</p> <p>0: 向全部从站发送请求报文。(广播)</p> <p>1-247: 向指定的从站发送请求报文。注: 247 是 MODBUS 最大的地址编号。</p> <p>[从站向主站发送相应报文时]</p> <p>发送相应报文时, 从站本站的站号会被存储。</p>
功能代码	<p>[主站向从站发送请求报文时]</p> <p>主站对从站指定功能代码。</p> <p>[从站向主站发送响应报文时]</p> <p>正常结束时, 被请求的功能代码会被存储。异常结束时, 最高位的位会 ON。</p>
数据	<p>[主站向从站发送请求报文时]</p> <p>存储用于执行通过功能代码所指定功能的信息。</p> <p>[从站向主站发送响应报文时]</p> <p>通过功能代码所指定功能的执行结果会被存储。</p> <p>异常结束时, 异常响应代码会被存储。</p>
出错检查	主站及从站会给全部发送报文自动添加检查代码, 并重新计算接收报文的检查代码。报文异常时, 删除报文。

由于 Modbus 协议为通用标准协议, 本文档只做简单的介绍, 具体的协议规范用户可自行查找资料。

2.1.1 Modbus RTU

Modbus RTU 模式是使用二进制代码收发帧的模式, 帧规格依据 Modbus 协议的规格如图 2-2 所示。

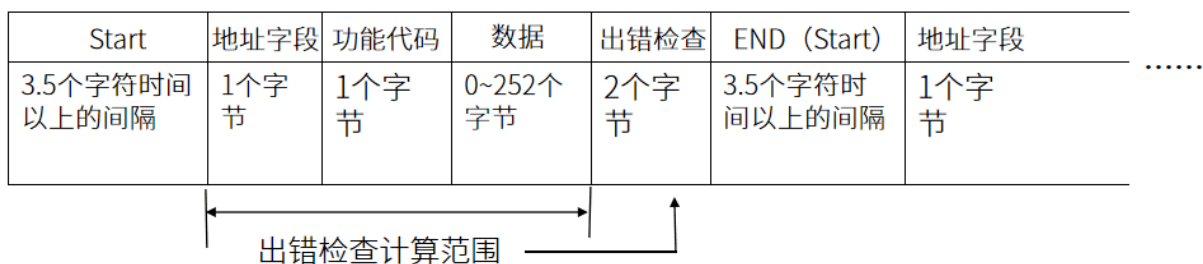


图 2-2: RTU 模式

RTU 模式的出错检查通过 CRC 进行。CRC 是 16 位 (2 个字节) 的二进制值。CRC 值是

由发送设备计算，并添加到报文中。接收设备在报文接收过程中重新计算 CRC，并和接收的实际值进行比较。进行比较的值如果不同则为出错。

2.1.2 Modbus TCP

ERC-G200 控制器作为从站并通过以太网连接各种 Modbus TCP 主站设备进行通信，协议的帧规格如图 2-3 所示。

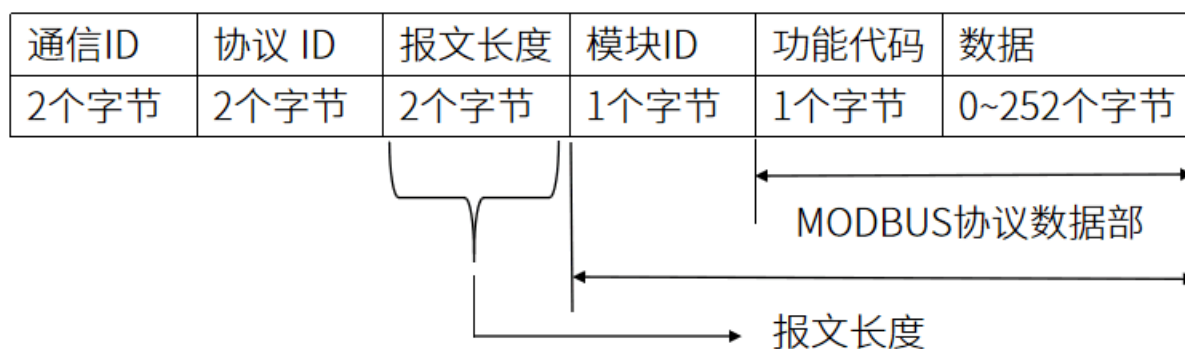


图 2-3：TCP

Modbus TCP 协议的帧规格的详细内容如下所示。

区域名	内容
通信 ID	主站将其用于对照从站发出的相应报文。
协议 ID	显示 PDU（协议数据部）的协议。 Modbus TCP 通信时，存储为 0。
报文长度	报文大小将以字节单位被存储。 所存储的报文长度即为该区域后的报文长度。
模块 ID	对应 RTU 中的地址字段，此处一般设定为 00。
功能代码	主站对从站指定处理内容。
数据	[主站向从站发送请求报文时] 存储处理的要求内容。 [从站向主站发送响应报文时] 存储处理的执行结果。

2.1.3 Modbus 标准功能码

功能代码	功能名	详细	1 个报文可访问的软元件数
01H	线圈读取	线圈读取（可以多点）	1-2000 点
02H	输入读取	输入读取（可以多点）	1-2000 点
03H	保持寄存器读取	保持寄存器读取（可以多点）	1-125 点
04H	输入寄存器读取	输入寄存器读取（可以多点）	1-125 点
05H	1 线圈写入	线圈写入（仅 1 点）	1 点
06H	1 寄存器写入	保持寄存器写入（仅 1 点）	1 点
0FH	多线圈写入	多点的线圈写入	1-1968 点
10H	多寄存器写入	多点的保持寄存器写入	1-123 点
16H	保持寄存器掩码写入	保持寄存器的 AND/OR 掩码写入（仅 1 点）	1 点
17H	批量寄存器读出/写入	保持寄存器的多点读出和多点写入	读出：1-125 点 写入：1-121 点

2.2 Modbus 功能的系统配置方法

2.2.1 Modbus 配置文件

ERC-G200 控制系统支持 Modbus RTU 和 Modbus TCP 两种通信模式。启用 Modbus 通信功能后可在“用户工艺—Modbus”界面中进行配置。

当通信模式为 Modbus TCP 时，在 Modbus 界面选择模式为 TCP，IP 地址为本机 IP，端口号为 502。点击“设置”，系统重启之后生效。



图 2-4 : 选择模式为 TCP

当通信模式为 Modbus RTU 时，在 modbus 界面选择模式为 RTU，根据实际情况配置各参数。参数包括：从站节点号，波特率，数据位长度，校验方式，停止位长度。点击“设置”，系统重启之后生效。



图 2-5：选择模式为 RTU

ELITE 控制器 485 端口为特殊的接插件，针脚定义为：2 为 GND，4 为 485+，3 为 485-。

提醒



在示教器界面中配置 modbus 仅支持 v2.10.0 及以上版本。

2.3 Modbus 功能的程序应用

2.3.1 软元件地址

目前本系统只支持对虚拟变量 M 的操作，M 为 BIT 型变量。单个 M 变量为线圈，多个连续的 M 变量可为寄存器。

本系统 M 变量支持的功能码如表 2-2 所示。

表 2-2. 功能码

变量	线圈地址	寄存器地址	支持的功能码	读写权限
M0-M527	0x0000-0x020F	0-0x41(8bit) 0-0x20(16bit)	0x01,0x03,0x04	只读
M528-M1471	0x0210-0x05BF	0x42-0xB7(8bit) 0x21-0x5B(16bit)	0x01,0x03,0x05, 0x06,0x0F,0x10	可读/可写
M1472-M1535	0x05C0-0x05FF	0xB8-0xBF(8bit) 0x5C-0x5F(16bit)	0x01,0x03,0x04	只读

提示

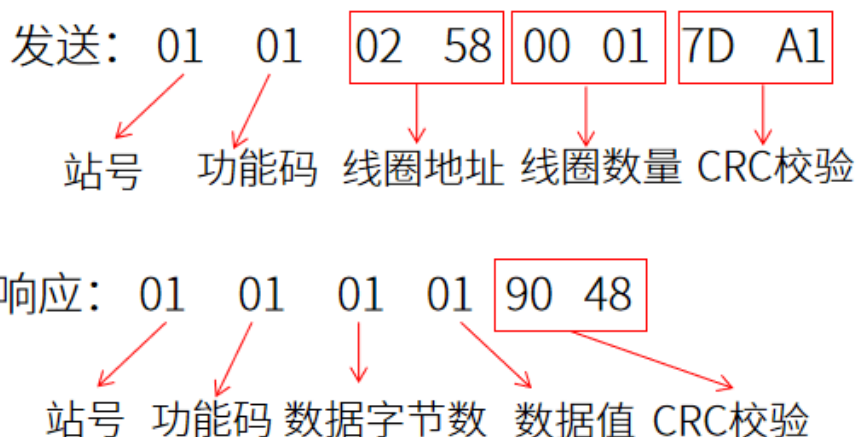


1. M0-M527, M1472-M1535为系统使用, 只能读取状态, 通过示教器可输入的 M 变量范围为 528-799。
2. M 变量的线圈地址即为变量的序号地址。
3. 兼容模式中勾选modbus: M 变量的寄存器地址从 M0 开始, 每连续的16个 M 变量形成一个寄存器地址, 例如 M0-M15, 寄存器地址为 0x0000; M16-M31, 寄存器地址为 0x0001, 以此类推;
兼容模式中不勾选modbus: M 变量的寄存器地址从 M0 开始, 每连续的 8个 M 变量形成一个寄存器地址, 例如 M0-M7, 寄存器地址为 0x0000; M8-M15, 寄存器地址为 0x0001, 以此类推。
4. Modbus 协议中一个寄存器为 16 位变量, 因此对 M 变量的一个寄存器地址进行操作, 实际操作的是连续的 16 个 M 变量。

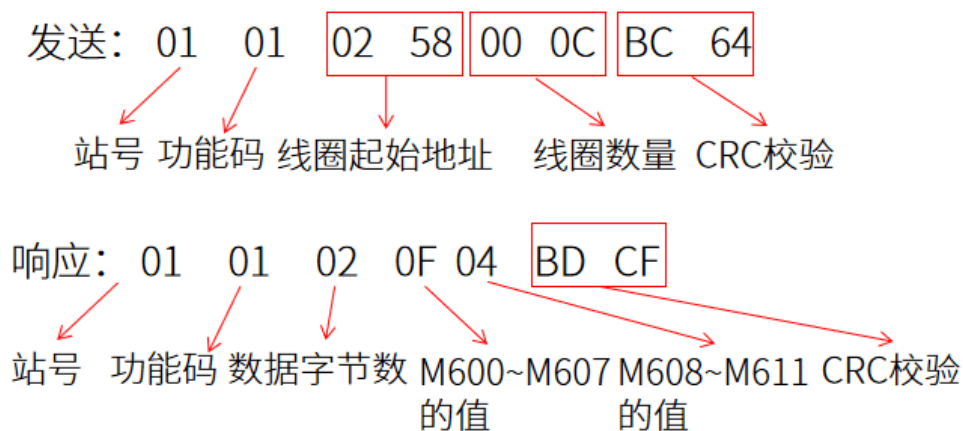
2.3.2 程序范例

2.3.2.1 Modbus RTU 通信报文范例

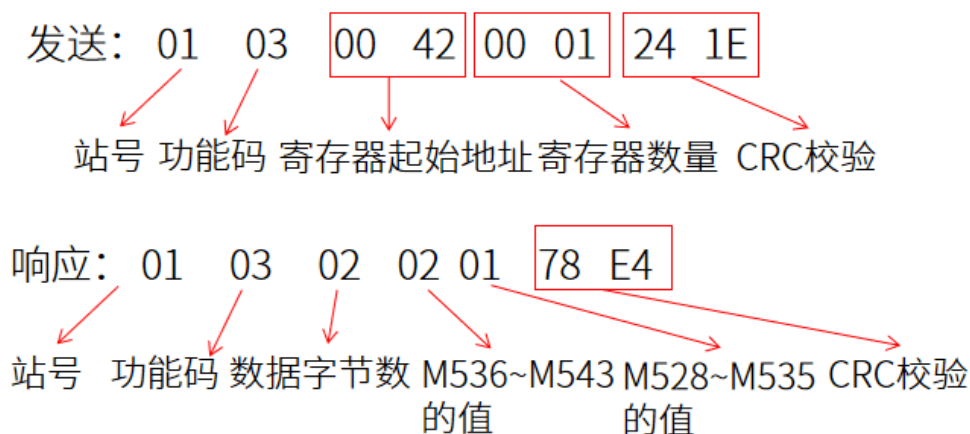
1. 主站读取机器人 M600 的值, 假设此时 M600 为 ON。



2. 主站读取机器人 M600-M611 的值，其中 M600-M603 全部为 ON，M610 为 ON，其他均为 OFF。



3. 主站读取机器人 M528-M543 的值，其中 M528 为 ON，M537 为 ON，其他均为 OFF。

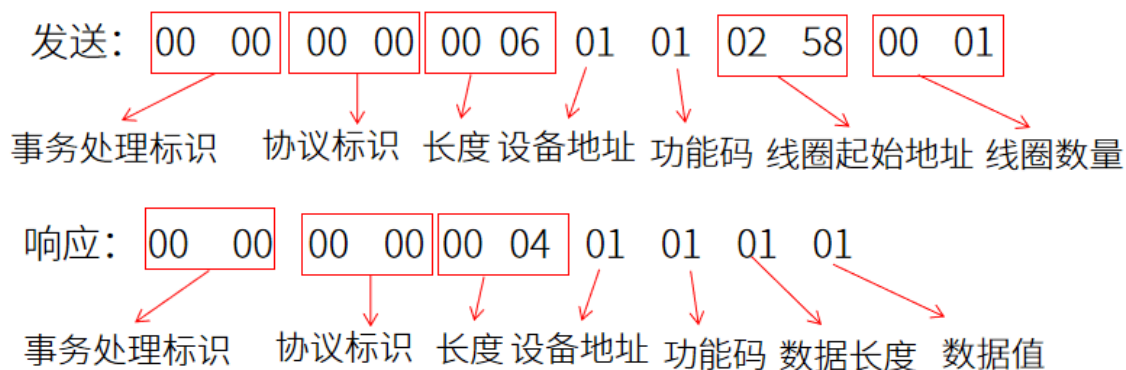


4. 主站将机器人的 M528-M535 这八个变量全部置为 ON。

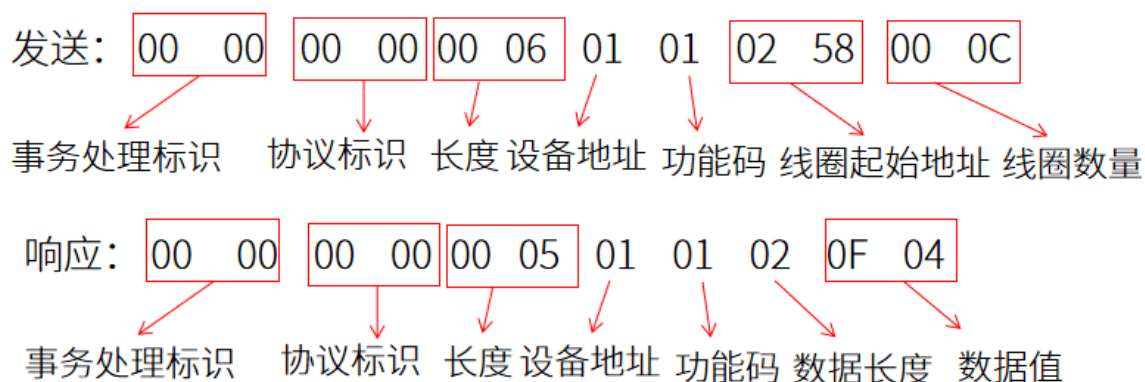


2.3.2.2 Modbus TCP 通讯报文范例

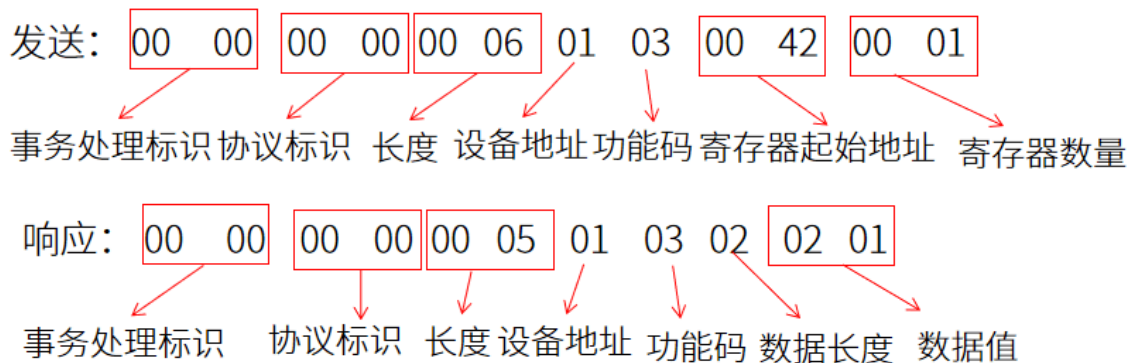
1. 主站读取机器人 M600 的值，M600 为 ON 时。



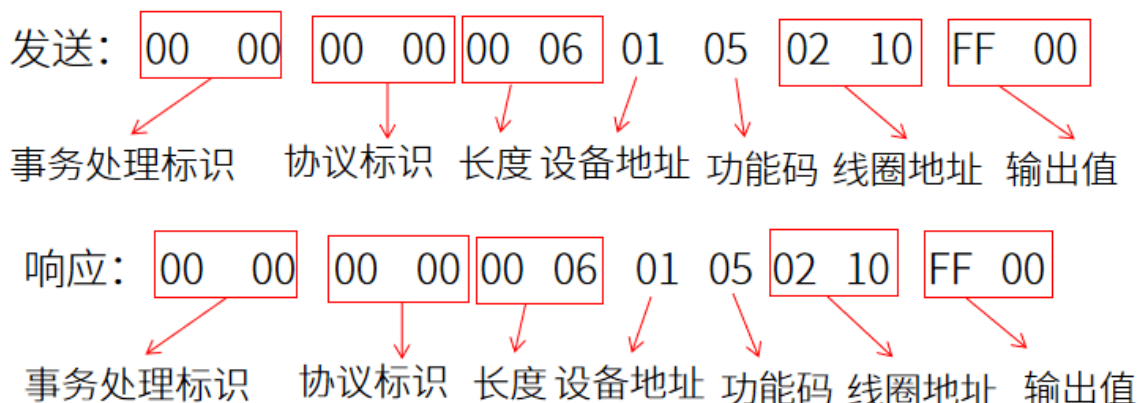
2. 主站读取机器人 M600-M611 的值, 其中 M600-M603 为 ON, M610 为 ON。其它均为 OFF。



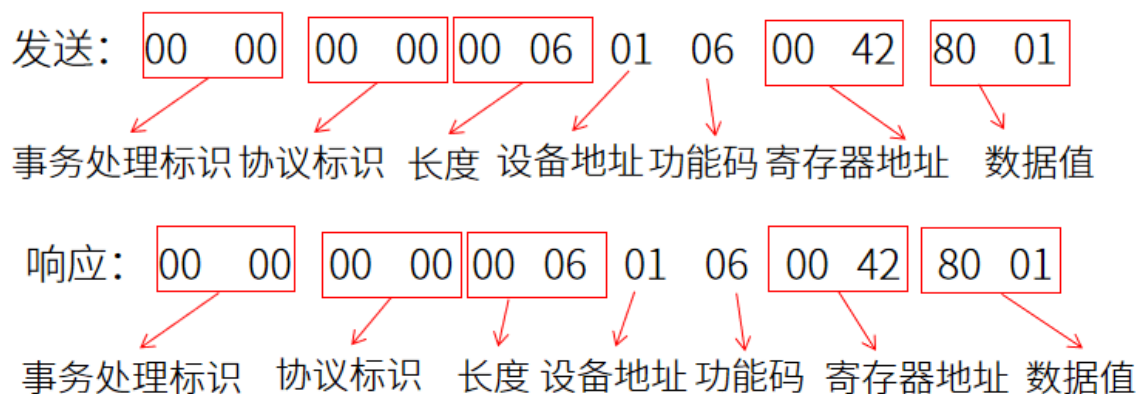
3. 主站读取机器人 M528-M543 的值, 其中 M528 为 ON, M537 为 ON, 其他均为 OFF。



4. 主站将机器人的 M528 置 ON。



5. 主站修改机器人的 M528~M543 的值，其中 M528 置为 ON，M543 置为 ON，其他均置为 OFF。



2.3.3 机器人端 M 变量的操作

在高级指令模式下，可以通过 MOUT 指令对单个或多个 M 变量操作；可以通过 MIN 指令将单个或多个 M 变量以二进制数值格式读入 B/I/D 变量中；可以通过 WAIT 指令来等待单个或多个 M 变量达到设定的条件。

在机器人程序中通过这些指令对 M 变量编程操作，形成需要的逻辑，同时通过 Modbus 协议与主站进行变量通信来实现工作站所需的功能。

2.4 获取机器人信息

2.4.1 简介

系统通过 Modbus，可将机器人当前关节角度、关节速度、当前 TCP 位姿、当前 TCP 速度、系统数字输入口、系统数字输出口、系统模拟量输入口、系统模拟量输出口发送到外部。

modbus 通讯周期为 10ms，在控制器报警状态下，通讯周期会达到 1010ms。

末端 I/O 相关数据：在整体数据基础上增加 1000ms 的周期。

2.4.2 获取机器人当前关节角度

寄存器起始地址：200，寄存器数据长度 16 位。

范围：从 200 开始连续 8 个寄存器分别保存 1~8 轴当前关节角度值。

单位：0.0002 弧度。

说明：获取数据后，需要进行除以 5000.0 操作，得到的单位才是弧度。

2.4.3 获取机器人当前关节角速度

寄存器起始地址：210，寄存器数据长度 16 位。

范围：从 210 开始连续 8 个寄存器，分别保存当前关节角度运行速度。

单位：0.0002 弧度/秒。

说明：获取数据后，需要进行除以 5000.0 操作，得到的单位才是弧度/秒；机器人在进行关节运动时，才能得到实际关节角速度，直线或圆弧运行时，该数据为 0。

2.4.4 获取机器人当前 TCP 位置

寄存器起始地址：220，寄存器数据长度 16 位。

范围：从 220 开始连续 6 个寄存器，分别保存当前直角坐标，X，Y，Z，Rx，Ry，Rz 值。
X，Y，Z 单位为：0.1 毫米，Rx，Ry，Rz 单位为：0.0002 弧度。

说明：获取 X，Y，Z 后，需进行除以 10 操作，得到的单位才是毫米，获取 Rx，Ry，Rz 后，需进行除以 5000 操作，得到的单位才是弧度。

2.4.5 保存安全参数使能状态

寄存器起始地址：228，寄存器数据长度 16 位。

范围：从 228 开始连续 1 个寄存器，保存安全参数使能状态。

说明：保存使能状态，1 使能，0 未使能。

提醒

本功能适用于 v2.17.0 及以上版本。

2.4.6 保存碰撞检测使能状态

寄存器起始地址：229, 寄存器数据长度 16 位。

范围：从 229 开始连续 1 个寄存器，保存碰撞检测使能状态。

说明：保存使能状态，1 使能，0 未使能。

提醒

本功能适用于 v2.17.0 及以上版本。

2.4.7 获取机器人当前 TCP 速度

寄存器起始地址：230, 寄存器数据长度 16 位。

范围：从 230 开始连续 1 个寄存器，保存当前 TCP 运行线速度。

单位：0.1 毫米/秒。

说明：获取数据后，需要进行除以 10.0 操作，得到的单位才是毫米/秒。

2.4.8 获取系统数字输入口状态

寄存器起始地址：240, 寄存器数据长度 16 位。

范围：从 240 开始连续 4 个寄存器，保存从 X0 到 X63 的数字输入口状态。

说明：寄存器中保存数据为 16 位输入口组成的二进制数据的十进制显示，例如若寄存器 240 返回的值是 38490，则对应的 X15——X0 输入口状态为：1001 0110 0101 1010；寄存器 1 保存 X0——X15 数据，寄存器 2 保存 X16——X31 数据，寄存器 3 保存 X32——X47 数据，寄存器 4 保存 X48——X63 数据。

2.4.9 获取系统数字输出口状态

寄存器起始地址：250，寄存器数据长度 16 位。

范围：从 250 开始连续 4 个寄存器，保存从 Y0 到 Y63 的数字输出口状态。

说明：寄存器中保存数据为 16 位输出口组成的二进制数据的十进制显示，例如若寄存器 250 返回的值是 38490，则对应的 Y15——Y0 输出口状态为：1001 0110 0101 1010；寄存器 1 保存 Y0——Y15 数据，寄存器 2 保存 Y16——Y31 数据，寄存器 3 保存 Y32——Y47 数据，寄存器 4 保存 Y48——Y63 数据。

2.4.10 获取系统模拟量输入口状态

寄存器起始地址：260，寄存器数据长度 16 位。

范围：从 260 开始连续 3 个寄存器，保存控制箱内两个模拟量输入口数据，末端 I/O 上一个模拟量输入口数据。

单位：mV。

说明：寄存器 260 对应控制箱模拟量输入口 1 数据，寄存器 261 对应控制箱模拟量口 2 数据，寄存器 262 对应末端 I/O 上模拟量输入口数据。

2.4.11 获取系统模拟量输出口状态

寄存器起始地址：270，寄存器数据长度 16 位。

范围：从 270 开始连续 5 个寄存器，保存控制箱内四个模拟量输出口数据，末端 I/O 上一个模拟量输出口数据。

单位：mV。

说明：寄存器 270 对应控制箱模拟量输出口 1 数据，寄存器 271 对应控制箱模拟量输出口 2 数据，寄存器 272 对应控制箱模拟量输出口 3 数据，寄存器 273 对应控制箱模拟量输出口 4 数据，寄存器 274 对应末端 IO 上模拟量输出口数据。

提醒



上述功能适用于 v2.9.3 及以上版本。

2.4.12 获取系统时间戳

寄存器起始地址：280，寄存器数据长度 16 位。

范围：从 280 开始连续 4 个寄存器，保存控制器从 1970 年 1 月 1 日 0 时 0 分 0 秒至今的毫秒数。

单位：ms。

说明：寄存器 280 保存最低位数据，寄存器 283 保存最高位数据。例如：寄存器 280 得到数据为 39529 (1001 1010 0110 1001)，寄存器 281 得到数据为 41785 (1010 0011 0011 1001)，寄存器 282 得到数据为 369 (0000 0001 0111 0001)，寄存器 283 得到数据为 0 (0000 0000 0000 0000)，则系统时间戳为 $1587581393513 (39529 + (41785 \ll 16) + (369 \ll 32) + (0 \ll 48))$

提醒



本功能适用于 v2.9.4 及以上版本。

2.4.13 获取机器人 tcp 加速度

寄存器起始地址：450

范围：从 450 开始的连续两个寄存器，保存 tcp 加速度

单位：0.1mm/s²

说明：450 为低位,451 为高位，高位需要将 unsigned int 类型转化为有符号的 short 类型，例如 450、451 的值分别为 56154、65535，65535 转化后的值为-1，则 tcp 加速度为： $(56154|((-1) \ll 16))/10 = -938.2 \text{ mm/s}^2$

2.4.14 获取机器人关节加速度

寄存器起始地址：460

范围：从 460 开始的连续 16 个寄存器，每两个寄存器保存一个关节加速度的值

单位：0.001 度/s²

说明：每两个寄存器中，低位在前，高位在后。高位需要将 unsigned int 类型转化为有符号的 short 类型，例如：寄存器 460 463 的值分别为 65024、65533、58885、65534，则：一关节加速度为 $(65024|((-3) \ll 16))/1000 = 131.584 \text{ 度/s}^2$ 二关节的加速度为 $(58885|((-2) \ll 16))/1000 = -72.187 \text{ 度/s}^2$

2.4.15 获取安全限制功率参数

寄存器起始地址：480

范围：从 480 开始的连续 4 个寄存器

单位：0.001w

说明：前两个寄存器保存正常模式的功率参数，后两个寄存器保存缩减模式的功率参数。例如数组 `value` 存储的是寄存器 480 483 的值。则正常模式功率参数为 $(\text{value}[0] | (\text{value}[1] \ll 16))/1000$ ，缩减模式功率参数为 $(\text{value}[2] | (\text{value}[3] \ll 16))/1000$ 。

2.4.16 获取安全限制动量参数

寄存器起始地址：484

范围：从 484 开始的连续 4 个寄存器

单位：0.001kg*m/s

说明：前两个寄存器保存正常模式的动量参数，后两个寄存器保存缩减模式的动量参数。例如数组 `value` 存储的是寄存器 484 487 的值。则正常模式功率参数为 $(\text{value}[0] | (\text{value}[1] \ll 16))/1000$ ，缩减模式功率参数为 $(\text{value}[2] | (\text{value}[3] \ll 16))/1000$ 。

2.4.17 获取安全限制工具力参数

寄存器起始地址：488

范围：从 488 开始的连续 4 个寄存器

单位：0.001N

说明：前两个寄存器保存正常模式的工具力参数，后两个寄存器保存缩减模式的工具力参数。例如数组 `value` 存储的是寄存器 488 491 的值。则正常模式功率参数为 $(\text{value}[0] | (\text{value}[1] \ll 16))/1000$ ，缩减模式功率参数为 $(\text{value}[2] | (\text{value}[3] \ll 16))/1000$ 。

2.4.18 获取安全限制肘部力参数

寄存器起始地址：492

范围：从 492 开始的连续 4 个寄存器

单位：0.001N

说明：前两个寄存器保存正常模式的肘部力参数，后两个寄存器保存缩减模式的肘部力参

数。例如数组 value 存储的是寄存器 492 495 的值。则正常模式功率参数为 $(\text{value}[0] | (\text{value}[1] \ll 16))/1000$ ，缩减模式功率参数为 $(\text{value}[2] | (\text{value}[3] \ll 16))/1000$ 。

2.4.19 获取安全限制速度百分比参数

寄存器起始地址：496

范围：从 496 开始的连续 4 个寄存器

单位：0.001%

说明：前两个寄存器保存正常模式下的速度百分比参数，后两个参数保存缩减模式下的百分比参数，例如数组 value 存储的是寄存器 496 499 的值。则正常模式百分比参数为 $(\text{value}[0] | (\text{value}[1] \ll 16))/1000$ ，缩减模式百分比参数为 $(\text{value}[2] | (\text{value}[3] \ll 16))/1000$ 。

2.4.20 获取碰撞检测灵敏度参数

寄存器起始地址：500

范围：从 500 开始的寄存器 1 个寄存器

单位：%

2.4.21 其他通用寄存器

寄存器数据长度 16 位。

范围：192 到 199，284 到 447。

说明：传送和暂存数据

提醒



本功能适用于 v2.16.2 及以上版本。

2.5 实例



```
1  #!/usr/bin/python
2  # -*- coding:UTF-8 -*-
3
4  import modbus_tk.modbus_tcp as mt
5  import modbus_tk.defines as md
6  from time import sleep
7
8  def getShortValue(value):
9      '''
10     状换value值为short类型
11     param value: unsigned int 类型
12     return: 16位带符号short类型
13     '''
14     if value >= pow(2, 15) :
15         return (~(value -1 )&(pow(2, 16) -1)) * -1
16     else:
17         return value
18
19     #创建ModbusTCP连接
20     master = mt.TcpMaster("192.168.1.203", 502)
21     master.set_timeout(5.0)
22
23     joint_value = []      # 用于保存机器人关节角度信息
24     joint_speed_value = [] # 用于保存机器人关节速度信息
25     tcp_value = [] # 用于保存机器人TCP位姿信息
26     tcp_speed_value = [] # 用于保存机器人TCP速度信息
27     input_value = [] # 用于保存机器人数字输入口信息
28     output_value = [] # 用于保存机器人数字输出口信息
29     ainput_value = [] # 用于保存机器人模拟量输入口信息
30     aoutput_value = [] # 用于保存机器人模拟量输出口信息
31     safety_params_enable_value = [] # 用于保存机器人安全参数使能信息
32     collision_enable_value = [] # 用于保存机器人碰撞检测使能信息
33     timestamp_value = [] # 用于保存机器人系统时间戳信息
34
35     while True:
36         # 分别读取R200,R210,R220,R228,R229,R230,R240,R250,R260,R270,R280
37         # 寄存器数据
38         hold_joint = master.execute(1, md.READ_HOLDING_REGISTERS, 200, 8)
39         hold_joint_speed = master.execute(1, md.READ_HOLDING_REGISTERS,
40             210, 8)
```

```
39     hold_tcp = master.execute(1, md.READ_HOLDING_REGISTERS, 220, 6)
40     hold_safety_params_enable_status = master.execute(1, md.
41         READ_HOLDING_REGISTERS, 228, 1)
42     hold_collision_enable_status = master.execute(1, md.
43         READ_HOLDING_REGISTERS, 229, 1)
44     hold_tcp_speed = master.execute(1, md.READ_HOLDING_REGISTERS, 230,
45         1)
46     hold_input = master.execute(1, md.READ_HOLDING_REGISTERS, 240, 3)
47     hold_output = master.execute(1, md.READ_HOLDING_REGISTERS, 250, 2)
48     hold_ainput = master.execute(1, md.READ_HOLDING_REGISTERS, 260, 3)
49     hold_aoutput = master.execute(1, md.READ_HOLDING_REGISTERS, 270, 5)
50     hold_timestamp = master.execute(1, md.READ_HOLDING_REGISTERS, 280,
51         4)
52
53     # 变换数据单位, 并保存到列表中
54     for joint in hold_joint:
55         joint_value.append(getShortValue(joint) / 5000.0 * 180 / 3.14)
56     for speed in hold_joint_speed:
57         joint_speed_value.append(getShortValue(speed) * 180 / 3.14 /
58             1000.0);
59     for tcp in hold_tcp:
60         if hold_tcp.index(tcp) < 3 :
61             tcp_value.append(getShortValue(tcp) / 10.0)
62         else:
63             tcp_value.append(getShortValue(tcp) / 5000.0 * 180 / 3.14)
64     for status in hold_safety_params_enable_status:
65         safety_params_enable_value.append(getShortValue(status))
66     for status in hold_collision_enable_status:
67         collision_enable_value.append(getShortValue(status))
68     for speed in hold_tcp_speed:
69         tcp_speed_value.append(getShortValue(speed) / 10.0)
70     for input in hold_input:
71         input_value.append(input)
72     for output in hold_output:
73         output_value.append(output)
74     for ainput in hold_ainput:
75         ainput_value.append(getShortValue(ainput) / 1000.0)
76     for aoutput in hold_aoutput:
77         aoutput_value.append(getShortValue(aoutput) / 1000.0)
```

```
73     for timestamp in hold_timestamp:
74         timestamp_value.append(timestamp)
75     # 打印机器人相关数据列表
76     print("joint_value = ", joint_value)
77     print("joint_speed_value = ", joint_speed_value)
78     print("tcp_value = ", tcp_value)
79     print("safety_params_enable_status = ", safety_params_enable_value)
80     print("collision_enable_status = ", collision_enable_value)
81     print("tcp_speed_value = ", tcp_speed_value)
82     print("input_value = ", input_value)
83     print("output_value = ", output_value)
84     print("ainput_value = ", ainput_value)
85     print("aoutput_value = ", aoutput_value)
86     #print("timestamp_value = ", timestamp_value[0] + (timestamp_value
87         [1] << 16) + (timestamp_value[2] << 32) + (timestamp_value[3] <<
88         48))
89     #print(timestamp_value[0], ", ", (timestamp_value[1] << 16) , ", ",
90         (timestamp_value[2] << 32) , ", ", (timestamp_value[3] << 48))
91     print(timestamp_value[0], timestamp_value[1], timestamp_value[2],
92         timestamp_value[3])
93     Time_stamp = (timestamp_value[0] + (timestamp_value[1] << 16) + (
94         timestamp_value[2] << 32) + (timestamp_value[3] << 48))
95     print(Time_stamp)
96     timeValue = time.gmtime((timestamp_value[0] + (timestamp_value[1]
97         << 16) + (timestamp_value[2] << 32) + (timestamp_value[3] << 48)
98         )/1000)
99     print(timeValue)
100     print(time.strftime("%Y-%m-%d %H:%M:%S",timeValue))
101     joint_value.clear()
102     safety_params_enable_value.clear()
103     collision_enable_value.clear()
104     joint_speed_value.clear()
105     tcp_value.clear()
106     tcp_speed_value.clear()
107     input_value.clear()
108     output_value.clear()
109     ainput_value.clear()
110     aoutput_value.clear()
111     timestamp_value.clear()
112     # 1秒获取一次
113     sleep(1)
```

第 3 章 Profinet

3.1 寄存器访问方式

3.1.1 Profinet 主站

模块功能及对应的控制器的 M 变量或 profinet 寄存器的地址范围具体如下：

设置虚拟量输出的值				
模块名称	MIN_64Byte	MIN_32Byte	MIN_16Byte	MIN_8Byte
地址范围	[912,1423]	[912,1167]	[912,1039]	[912,975]
获取虚拟量输出的值				
模块名称	MOUT_64Byte	MOUT_32Byte	MOUT_16Byte	MOUT_8Byte
地址范围	[400,911]	[400,655]	[400,527]	[400,463]
设置 Int 输入寄存器的值				
模块名称	RIN_32Int	RIN_16Int	RIN_8Int	
地址范围	[0,31]	[0,15]	[0,7]	
获取 Int 输出寄存器的值				
模块名称	ROUT_32Int	ROUT_16Int	ROUT_8Int	
地址范围	[0,31]	[0,15]	[0,7]	
设置 Float 输入寄存器的值				
模块名称	RIN_32Float	RIN_16Float	RIN_8Float	
地址范围	[0,31]	[0,15]	[0,7]	

获取 Float 输出寄存器的值

模块名称	ROUT_32Float	ROUT_16Float	ROUT_8Float
地址范围	[0,31]	[0,15]	[0,7]

获取运动相关状态寄存器的值

模块名称	status_motion
地址范围	[0,51]

获取数字量输入输出的值

模块名称	status_DIO
地址范围	[52,55]

获取模拟量输入输出的值

模块名称	status_AIO
地址范围	[56,63]

获取碰撞检测、安全参数相关寄存器的值

模块名称	status_safetyParams
地址范围	[64,81]

获取其他状态寄存器的值

模块名称	status_other
地址范围	[82,127]

3.1.2 SDK 接口

详细信息请参见 SDK 手册。

3.1.3 Lua 函数

详细信息请参见 Lua 手册。

3.2 Profinet 配置

3.2.1 启用 Profinet 功能

用户可选择“系统 > 系统配置 > 机器人配置”，并勾选“Profinet”，启用 Profinet 功能，如图 3-1 所示。



图 3-1：启用 Profinet

3.2.2 设置 Profinet

1. 将艾利特机器人的 GSD 文件安装到 TIA PORTAL 中管理通用栈，如图 3-2 所示。

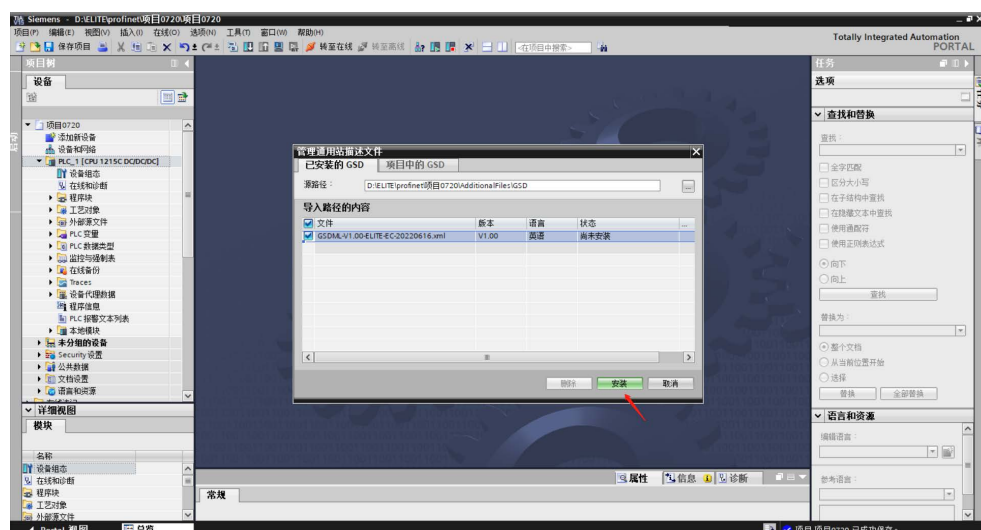


图 3-2：安装 GSD 文件

2. 添加 Elite Robot EC Device 到设备和网络，如图 3-3 所示。

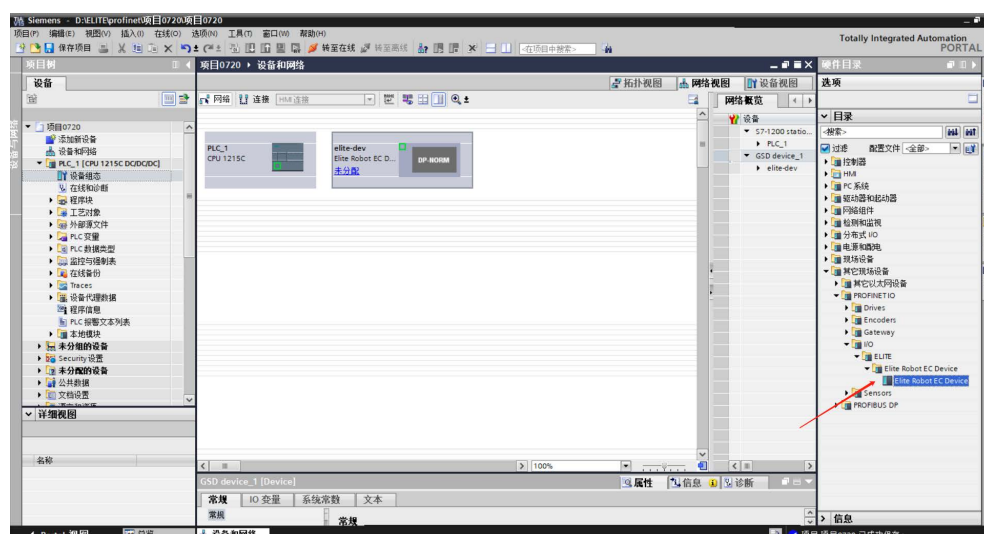


图 3-3：添加到设备和网络

3. 连接 Elite Robot EC Device 到 PLC，如图 3-4 所示。

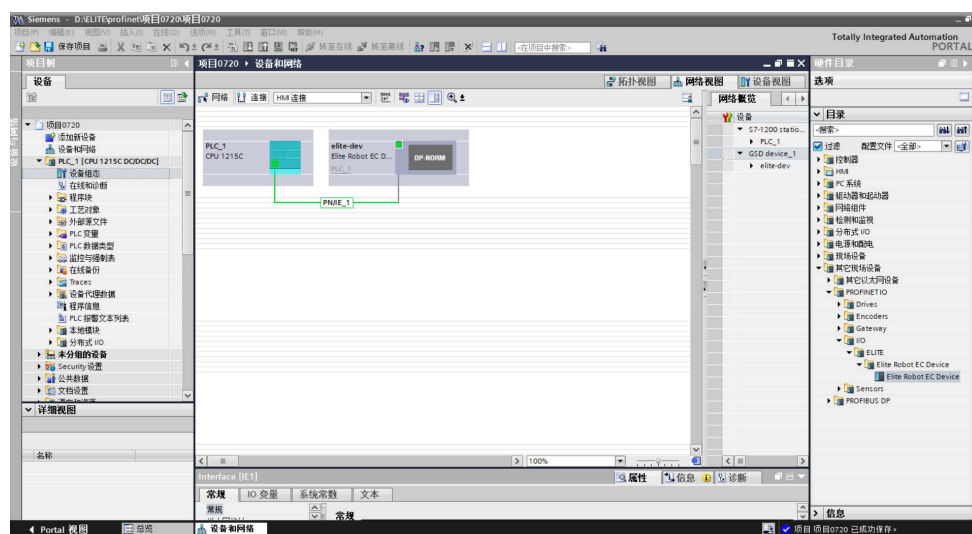


图 3-4：连接到 PLC

4. 在 Elite Robot EC Device 设备的属性中设置 IP 地址和 I/O 周期，如下图所示。

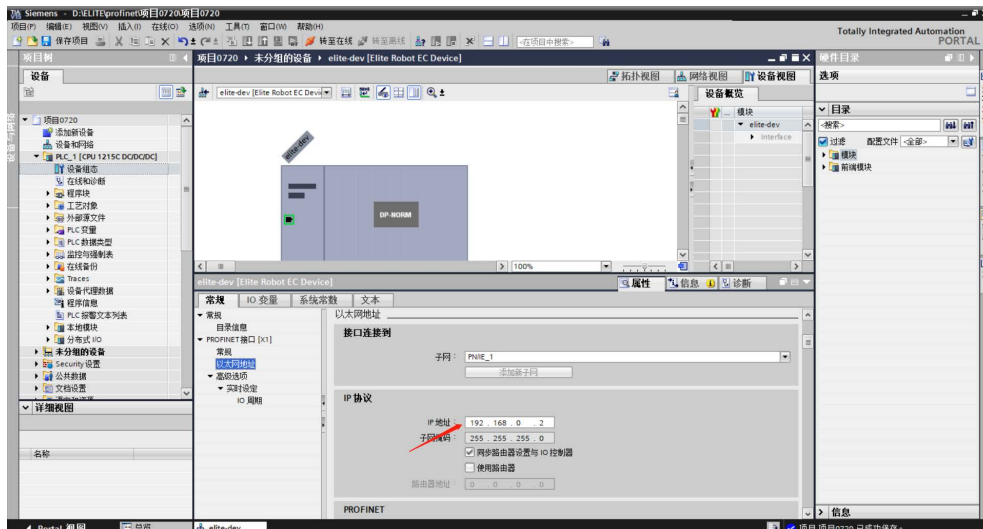


图 3-5：设置 IP 地址

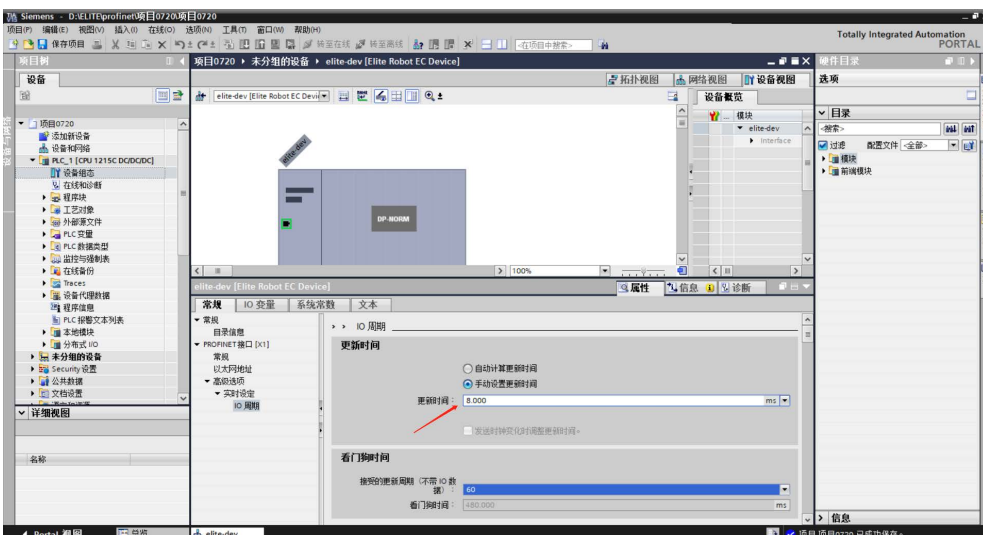


图 3-6：设置 I/O 周期

5. 将硬件目录中所需的模块添加到 elite-dev 设备接口，如图 3-7 所示。

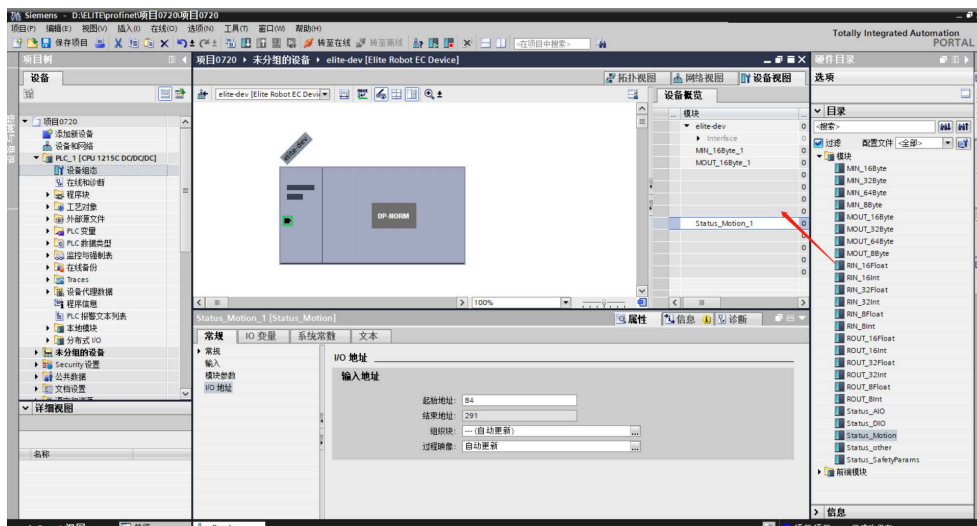
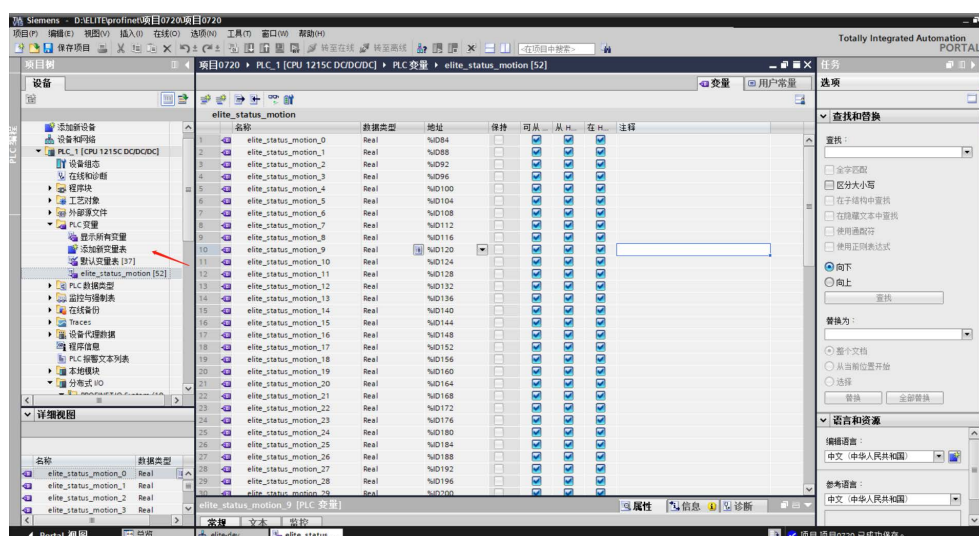


图 3-7：添加硬件模块

6. 添加所需的 PLC 变量表。



7. 根据所需添加监控表，如图 3-8 所示。

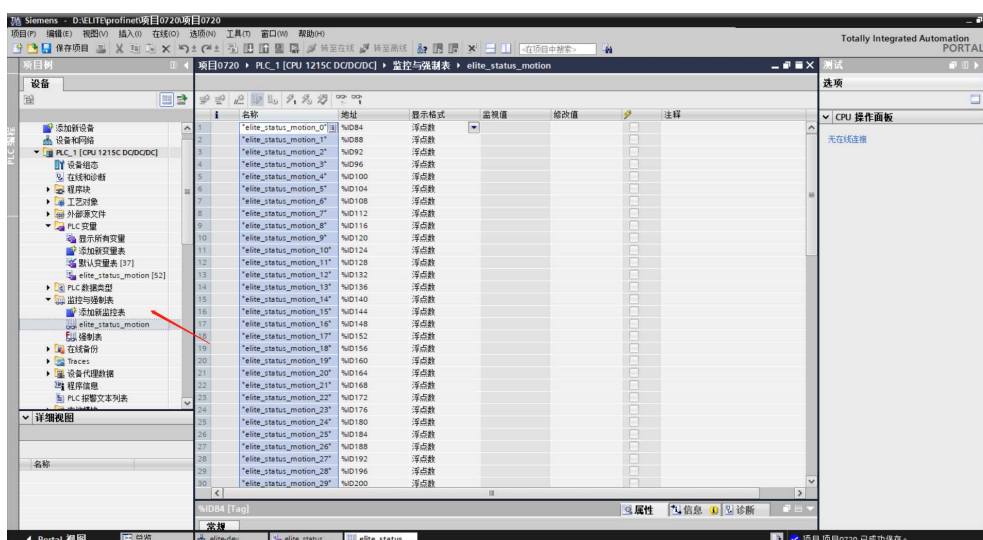


图 3-8：添加监控表

8. 根据所需设置强制表，如图 3-9 所示。

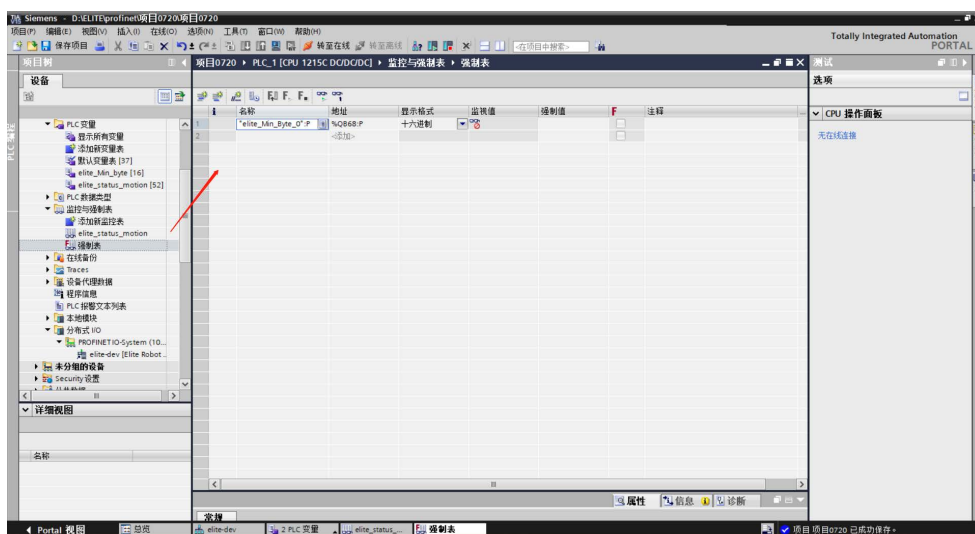
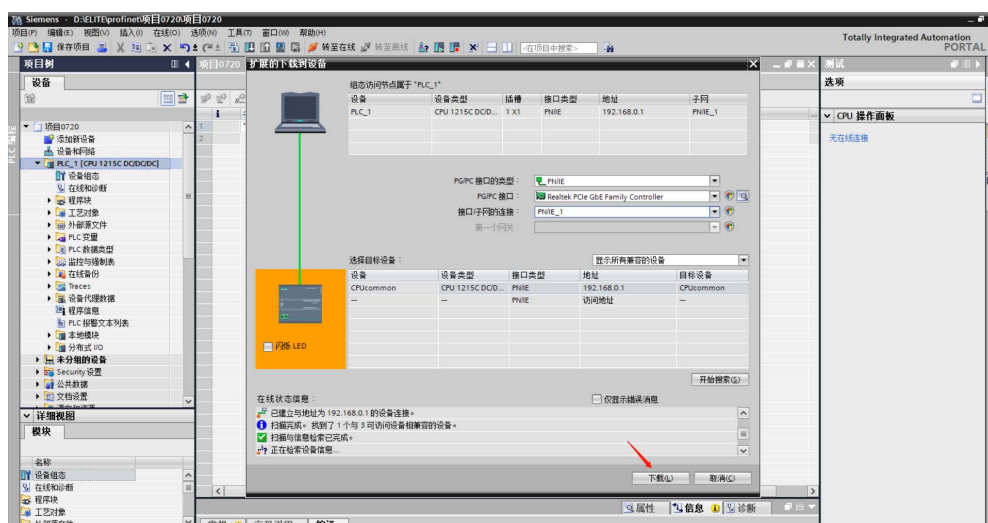
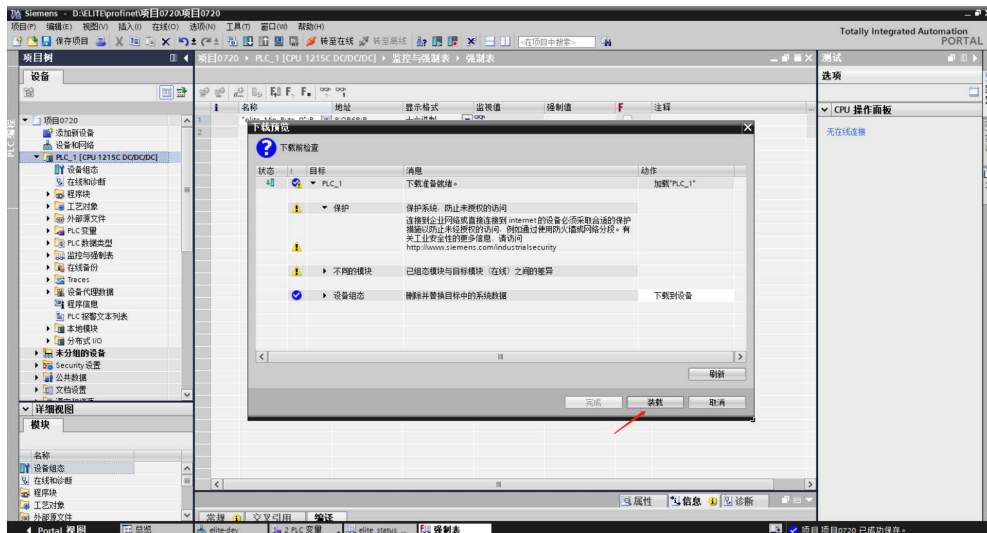


图 3-9：设置强制表

9. 下载硬件配置到 PLC，如下图所示。





10. 重新启动 PLC 后，PLC 状态指示灯无异常，即代表 profinet 通讯正常，如图 3-10 所示。

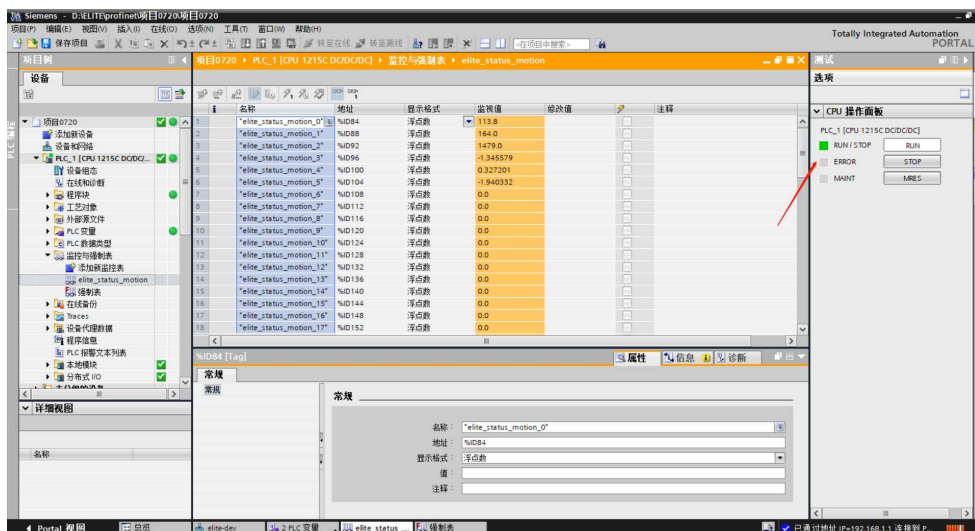


图 3-10：验证结果

如需下载，请登录官网：<https://www.elibot.com/service/technical>，即可找到GSD文件。

3.3 获取机器人信息

3.3.1 简介

机器人控制器作为 profinet 从站，目前包含以下模块：

MIN_64Byte, MIN_32Byte, MIN_16Byte, MIN_8Byte, MOUT_64Byte, MOUT_32Byte, MOUT_16Byte, MOUT_8Byte, RIN_32Int, RIN_16Int, RIN_8Int, ROUT_32Int, ROUT_16Int, ROUT_8Int, RIN_32Float, RIN_16Float, RIN_8Float, ROUT_32Float, ROUT_16Float,

文档编号：EC

ROUT_8Float, status_motion, status_DIO, status_AIO, status_safetyParams, status_other。

3.3.1.1 获取机器人当前位姿

寄存器起始地址：0，寄存器数据长度 32 位

范围：从 0 开始的 6 个寄存器分别保存当前直角坐标，X，Y，Z，Rx，Ry，Rz 值

数据类型：float

单位：X，Y，Z 单位为：毫米，Rx，Ry，Rz 单位为：弧度。

3.3.1.2 获取机器人当前 TCP 速度

寄存器起始地址：6，寄存器数据长度 32 位

范围：从 6 开始连续 1 个寄存器，保存当前 TCP 速度

数据类型：float

单位：毫米/秒

3.3.1.3 获取机器人 TCP 加速度

寄存器起始地址：7，寄存器数据长度 32 位

范围：从 7 开始连续 1 个寄存器，保存 TCP 加速度

数据类型：float

单位：单位 mm/s^2

3.3.1.4 获取机器人当前关节角度

寄存器起始地址：28，寄存器数据长度 32 位

范围：从 28 开始的 6 个寄存器分别保存 1~6 轴关节角度值

数据类型：float

单位：弧度

3.3.1.5 获取机器人当前关节角速度

寄存器起始地址：34，寄存器数据长度 32 位

范围：从 34 开始的 6 个寄存器分别保存当前关节角速度

数据类型：float

单位：弧度/秒

3.3.1.6 获取机器人关节加速度

寄存器起始地址：40，寄存器数据长度 32 位

范围：从 40 开始连续 6 个寄存器，依次保存 6 个关节加速度的值。

数据类型：float

单位：度/ s^2

3.3.1.7 获取系统数字输入口状态

寄存器起始地址：52，寄存器数据长度 32 位

范围：从 52 开始连续 2 个寄存器，保存 X0 到 X63 的数字输入口状态

数据类型：unsigned int

单位：无

说明：寄存器中保存数据为 32 位输入口组成的二进制数据的十进制显示，例如若寄存器 52 返回的值是 153210，则对应的 X31——X0 输入口状态为：0000 0000 0000 0010 0101 0110 0111 1010；寄存器 53 保存 X0——X31 数据，寄存器 53 保存 X32——X63 数据

3.3.1.8 获取系统数字输出口状态

寄存器起始地址：54，寄存器数据长度 32 位

范围：从 54 开始连续 2 个寄存器，保存 Y0 到 Y63 的数字输出口状态

数据类型：unsigned int

单位：无

说明：寄存器中保存的数据位 16 位输入口组成的二进制数据的十进制显示，例如若寄存器 54 返回的值是 153210，则对应的 Y31——Y0 输入口状态为：0000 0000 0000 0010 0101 0110

0111 1010; 寄存器 54 保存 Y0——Y31 数据, 寄存器 55 保存 Y32——Y63 数据

3.3.1.9 获取系统模拟量输入值

寄存器起始地址: 56, 寄存器数据长度 32 位

范围: 从 56 开始连续 3 个寄存器, 保存控制箱内两个模拟量输入口数据, 末端 I/O 上一个模拟量输入口数据.

数据类型: float

单位: mv

说明: 寄存器 56 对应控制器模拟量输入口 1 数据, 寄存器 57 对应控制器模拟量输入口 2 数据, 寄存器 58 对应末端 I/O 模拟量输入口 3 数据.

3.3.1.10 获取系统模拟量输出值

寄存器起始地址: 59, 寄存器数据长度 32 位

范围: 从 59 开始连续 5 个寄存器, 保存控制箱内四个模拟量输出口数据, 末端 I/O 上一个模拟量输出口数据。

数据类型: float

单位: mv

说明: 寄存器 59 对应控制箱模拟量输出口 1 数据, 寄存器 60 对应控制箱模拟量输出口 2 数据, 寄存器 61 对应控制箱模拟量输出口 3 数据, 寄存器 62 对应控制箱模拟量输出口 4 数据, 寄存器 63 对应末端 I/O 模拟量输出口数据。

3.3.1.11 获取碰撞检测使能状态

寄存器起始地址: 64, 寄存器数据长度 32 位

范围: 从 64 开始连续 1 个寄存器, 保存碰撞检测使能状态

数据类型: unsigned int

单位: 无

说明: 1 为使能, 0 为未使能

3.3.1.12 获取碰撞检测灵敏度参数

寄存器起始地址：65, 寄存器的数据长度 32 位

范围：从 65 开始连续 1 个寄存器，保存碰撞检测灵敏度参数

数据类型：float

单位：%

3.3.1.13 获取安全参数使能状态

寄存器起始地址：66, 寄存器数据长度 32 位

范围：从 66 开始连续 1 个寄存器，保存安全参数使能状态.

数据类型：unsigned int

单位：无

说明：1 为使能，0 为未使能

3.3.1.14 获取安全限制功率参数

寄存器起始地址：67, 寄存器的数据长度 32 位

范围：从 67 开始连续 2 个寄存器，分别保存正常模式和缩减模式安全限制功率参数

数据类型：float

单位：w

说明：寄存器 67 对应正常模式安全限制功率参数，寄存器 68 对应缩减模式安全限制功率参数

3.3.1.15 获取安全限制动量参数

寄存器起始地址：69, 寄存器的数据长度 32 位

范围：从 69 开始连续 2 个寄存器，分别保存正常模式和缩减模式安全限制动量参数

数据类型：float

单位：kg*m/s

说明：寄存器 69 对应正常模式安全限制动量参数，寄存器 70 对应缩减模式安全限制动量参数

3.3.1.16 获取安全限制工具力参数

寄存器起始地址：71，寄存器的数据长度 32 位

范围：从 71 开始连续 2 个寄存器，分别保存正常模式和缩减模式安全限制工具力参数

数据类型：float

单位：N

说明：寄存器 71 对应正常模式安全限制工具力参数，寄存器 72 对应缩减模式安全限制工具力参数。

3.3.1.17 获取安全限制肘部力参数

寄存器起始地址：73，寄存器的数据长度 32 位

范围：从 73 开始连续 2 个寄存器，分别保存正常模式和缩减模式安全限制肘部力参数

数据类型：float

单位：N

说明：寄存器 73 对应正常模式安全限制肘部力参数，寄存器 74 对应缩减模式安全限制肘部力参数。

3.3.1.18 获取安全限制速度百分比参数

寄存器起始地址：75，寄存器的数据长度 32 位

范围：从 75 开始连续 2 个寄存器，分别保存正常模式和缩减模式速度百分比参数

数据类型：float

单位：%

说明：寄存器 75 对应正常模式安全限制百分比参数，寄存器 76 对应缩减模式安全限制速度百分比参数

3.3.1.19 获取系统时间戳

寄存器起始地址：82，寄存器数据长度 32 位

范围：从 82 开始连续 2 个寄存器，保存控制器从 1970 年 1 月 1 日 0 时 0 分至今的毫秒数。

数据类型：unsigned int

单位：ms

说明：寄存器 82 保存最低位数据，寄存器 83 保存最高位数据。例如：寄存器 82 得到数据为 2738461289，寄存器 83 得到数据为 369，则系统时间戳为 1587581393513 $(2738461289 + (369 \ll 32))$

第 4 章 Ethernet/IP接口与协议

4.1 Ethernet/IP简介

Ethernet/IP(Industrial Protocol)是将标准的TCP/IP以太网延伸到工业实时控制并和通用工业协议（CIP）结合，将很好地帮助用户获得更加开放集成的工业自动化和信息化的整体解决方案。

EC系列产品支持Ethernet/IP从站应用。控制器作为从站工作时，无须额外配置。主从站间通过RJ-45双绞线通信，由主站主动发起连接建立请求，连接状态都将由主站反馈。

4.1.1 相关术语

1. Originator: 向从站发起连接请求的一方，一般是主站或客户端设备。
2. Target: 终端节点或者是从站设备，被请求建立连接的一方。
3. O->T: 指示发起者到目标节点的方向，用于描述数据报由主站发送到从站。
4. T->O: 指示从目标到发起者的方向，用于描述数据报由从站发送到主站。
5. RPI: Requested Packet Interval, 数据报的请求周期，单位为ms。
6. Connection Timeout Multiplier: 其值表示的乘数与RPI相乘得到数据报传输的超时时间，一般由主站设置。
7. point to point: 一种主从站的连接类型，点对点连接。
8. Multicast: 一种主从站的连接类型, 由一个从站对多个主站传播。（目前暂不支持这种链接）。
9. EDS: Electronic Data Sheet, 一种包含从站所有配置信息的文件，主站必须读取这个配置文件才能生成对从站的正确请求报文。

4.1.2 Ethernet/IP寄存器排布规则

Ethernet/IP 部分寄存器被划分为11个组（也称slot），每个slot实质是一个Ethernet/IP连接，且都包含有 O->T与 T->O两个方向的数据，其中只有一个方向上承载的数据为有效数据，而寄存器数据最终会在哪一个方向传输取决于寄存器的实际意义。如从控制器的视角，所有类型的输出寄存器会在T->O方向传输，此时另一个方向O->T存的意义可以认为是对T->O方向数据的确认；相对的，所有类型的输入寄存器会在O->T方向传输。下面分别对每组寄存器进行说明：

表 4-1 . 寄存器的详细说明

Adapter Slot 0	
寄存器类型	32 个 32 位 int 输出寄存器, 地址范围 [0,31]
传输方向	T->O
Adapter Slot 1	
寄存器类型	32 个 32 位 int 输入寄存器, 地址范围 [0,31]
传输方向	O->T
Adapter Slot 2	
寄存器类型	32 个 float 输出寄存器, 地址范围 [0,31]
传输方向	T->O
Adapter Slot 3	
寄存器类型	32 个 float 输入寄存器, 地址范围 [0,31]
传输方向	O->T

Adapter Slot 4		
寄存器类型	52 个 float 输入寄存器, 地址范围 [0,51]	
传输方向	T->O	
各数据域含义	i. 机器人当前位置姿	寄存器起始地址: 0, 寄存器数据长度 32 位 范围: 从 0 开始的 6 个寄存器分别保存当前直角坐标, X、Y、Z、Rx、Ry、Rz值 数据类型: float 单位: X/Y/Z: 毫米/秒, Rx/Ry/Rz: 弧度
	ii. 机器人当前 TCP 速度	寄存器起始地址: 6, 寄存器数据长度 32 位 范围: 从 6 开始连续 1 个寄存器, 保存当前 TCP 速度 数据类型: float 单位: 毫米/秒
	iii. 机器人当前 TCP 加速度	寄存器起始地址: 7, 寄存器数据长度 32 位 范围: 从 7 开始连续 1 个寄存器, 保存当前 TCP 加速度 数据类型: float 单位: mm/s ²

Adapter Slot 4		
各数据域含义	iv. 机器人当前 关节角度	寄存器起始地址：28，寄存器数据长度 32 位 范围：从 28 开始的 6 个寄存器分别保存 1~6 轴关节角度值 数据类型：float 单位：弧度
	v. 机器人当前 关节角速度	寄存器起始地址：34，寄存器数据长度 32 位 范围：从 34 开始的 6 个寄存器分别保存当 前关节角速度 数据类型：float 单位：弧度/秒
	vi. 机器人关 节加速度	寄存器起始地址：40，寄存器数据长度 32 位 范围：从 40 开始连续 6 个寄存器，依次保 存 6 个关节加速度的值 数据类型：float 单位：度/s ²

Adapter Slot 5		
寄存器类型	4 个 32 位状态	
	寄存器,地址范围 [0,3]	
传输方向	T->O	
各数据域含义	i. 系统数字输入状态	寄存器起始地址：0，寄存器数据长度 32 位 范围：从 0 开始连续 2 个寄存器，保存 X0 到 X63 的数字输入状态 数据类型：unsigned int 单位：无 说明：寄存器中保存数据为 32 位输入组 成的二进制数据的十进制显示，例如若寄存器 0 返回的值是 153210，则对应的 X31——X0 输入状态为：0000 0000 0000 0010 0101 0110 0111 1010；寄存器 0 保存 X0——X31 数据，寄存器 1 保存 X32—— X63 数据

Adapter Slot 5

各数据域含义	ii. 系统数字输	寄存器起始地址：2，寄存器数据长度 32 位
	出口状态	范围：从 2 开始连续 2 个寄存器，保存 Y0 到 Y63 的数字输出口状态
		数据类型：unsigned int
		单位：无
		说明：寄存器中保存的数据位 16 位输入口组成的二进制数据的十进制显示，例如若寄存器 2 返回的值是 153210，则对应的 Y31——Y0 输入口状态为：0000 0000 0000 0010 0101 0110 0111 1010；寄存器 2 保存 Y0——Y31 数据，寄存器 3 保存 Y32——Y63 数据

Adapter Slot 6		
寄存器类型	8 个 32 位状态	
	寄存器,地址范围 [0,7]	
传输方向	T->O	
各数据域含义	i. 系统模拟量输入值	<p>寄存器起始地址：0，寄存器数据长度 32 位</p> <p>范围：从 0 开始连续 3 个寄存器，保存控制箱内两个模拟量输入口数据，末端 I/O 上一个模拟量输入口数据</p> <p>数据类型：float</p> <p>单位：mv</p> <p>说明：寄存器 0 对应控制器模拟量输入口 1 数据，寄存器 1 对应控制器模拟量输入口 2 数据，寄存器 2 对应末端 I/O 模拟量输入口 3 数据</p>

Adapter Slot 6		
各数据域含义	ii. 系统模拟量	寄存器起始地址：3，寄存器数据长度 32 位
	输出值	范围：从 3 开始连续 5 个寄存器，保存控制箱内四个模拟量输出口数据，末端 I/O 上一个模拟量输出口数据
		数据类型：float
		单位：mv
		说明：寄存器 3 对应控制箱模拟量输出口 1 数据，寄存器 4 对应控制箱模拟量输出口 2 数据，寄存器 5 对应控制箱模拟量输出口 3 数据，寄存器 6 对应控制箱模拟量输出口 4 数据，寄存器 7 对应末端 I/O 模拟量输出口数据
Adapter Slot 7		
寄存器类型	18 个 32 位状态寄存器,地址	
	范围 [0,17]	
传输方向	T->O	

Adapter Slot 7		
各数据域含义	i. 碰撞检测使能状态	寄存器起始地址：0，寄存器数据长度 32 位 范围：从 0 开始连续 1 个寄存器，保存碰撞检测使能状态 数据类型：unsigned int 单位：无 说明：1 为使能，0 为未使能
	ii. 碰撞检测灵敏度参数	寄存器起始地址：1，寄存器数据长度 32 位 范围：从 1 开始连续 1 个寄存器，保存碰撞检测灵敏度参数 数据类型：float 单位：%
	iii. 安全参数使能状态	寄存器起始地址：2，寄存器数据长度 32 位 范围：从 2 开始连续 1 个寄存器，保存安全参数使能状态 数据类型：unsigned int 单位：无 说明：1 为使能，0 为未使能

Adapter Slot 7		
各数据域含义	iv. 安全限制功率参数	<p>寄存器起始地址：3，寄存器数据长度 32 位</p> <p>范围：从 3 开始连续 2 个寄存器，分别保存正常模式和缩减模式安全限制功率参数</p> <p>数据类型：float</p> <p>单位：w</p> <p>说明：寄存器 3 对应正常模式安全限制功率参数，寄存器 4 对应缩减模式安全限制功率参数</p>
	v. 安全限制动量参数	<p>寄存器起始地址：5，寄存器数据长度 32 位</p> <p>范围：从 5 开始连续 2 个寄存器，分别保存正常模式和缩减模式安全限制动量参数</p> <p>数据类型：float</p> <p>单位：kg*m/s</p> <p>说明：寄存器 5 对应正常模式安全限制动量参数，寄存器 6 对应缩减模式安全限制动量参数</p>

Adapter Slot 7

各数据域含义	vi. 安全限制	寄存器起始地址：7，寄存器数据长度 32 位
	工具力参数	范围：从 7 开始连续 2 个寄存器，分别保存正常模式和缩减模式安全限制工具力参数
		数据类型：float
		单位：N
		说明：寄存器 7 对应正常模式安全限制工具力参数，寄存器 8 对应缩减模式安全限制工具力参数
	vii. 安全限制	寄存器起始地址：9，寄存器数据长度 32
	肘部力参数	位
		范围：从 9 开始连续 2 个寄存器，分别保存正常模式和缩减模式安全限制肘部力参数
		数据类型：float
		单位：N
		说明：寄存器 9 对应正常模式安全限制肘部力参数，寄存器 10 对应缩减模式安全限制肘部力参数

Adapter Slot 7		
各数据域含义	viii. 安全限制	寄存器起始地址：11，寄存器数据长度 32 位
	速度百分比参数	范围：从 11 开始连续 2 个寄存器，分别保存正常模式和缩减模式速度百分比参数
		数据类型：float
		单位：%
		说明：寄存器 11 对应正常模式安全限制百分比参数，寄存器 12 对应缩减模式安全限制速度百分比参数
Adapter Slot 8		
寄存器类型	46 个 32 位状态寄存器,地址范围 [0,45]	
传输方向	T->O	
各数据域含义	i. 获取系统时间戳	寄存器起始地址：0，寄存器数据长度 32 位
		范围：从 0 开始连续 2 个寄存器，保存控制器从 1970 年 1 月 1 日 0 时 0 分至今的毫秒数
		说明：寄存器 0 保存最低位数据，寄存器 1 保存最高位数据。例如：寄存器 0 得到的数据为 2738461289，寄存器 1 得到的数据为 369，则系统时间戳为 1587581393513 (2738461289 + (369<<32))

Adapter Slot 9		
寄存器类型	64 个 8 位状态	
	寄存器,地址范围 [0,63]	
传输方向	O->T	
各数据域含义	i. 输入 M 变量映射	寄存器起始地址: 0, 寄存器数据长度 8 位 范围: 从 0 开始连续 64 个寄存器, 映射到控制器上从起始地址为 912 开始的 64*8 个 M 变量
Adapter Slot 10		
寄存器类型	64 个 8 位状态	
	寄存器,地址范围 [0,63]	
传输方向	T->O	
各数据域含义	i. 输出 M 变量映射	寄存器起始地址: 0, 寄存器数据长度 8 位 范围: 从 0 开始连续 64 个寄存器, 保存控制器上从起始地址为 400 的 64*8 个 M 变量

4.2 Ethernet/IP 寄存器访问方式

4.2.1 Ethernet/IP 主站设备

4.2.1.1 配置方式

支持Ethernet/IP协议的主站设备丰富繁多，但配置方式并不复杂，此处总结了配置的普遍适用性，具体步骤如下：

- 配置主站IP地址，使与从站在同一子网
- 加载本产品对应的EDS文件，如需下载，请登录官网：www.elibot.com/service/technical
- 设定从站对应的IP地址
- 添加由EDS中提供的连接(slot)，用于访问指定的Ethernet/IP从站的寄存器组
- 按需修改此连接中相关配置项，如RPI、超时时间等
- 修改完毕后将配置载入主站设备并运行

4.2.2 SDK 接口

详细信息请参见 SDK 手册。

4.2.3 Lua 函数

详细信息请参见 Lua 手册。

明天比今天更简单一点

- 联系我们

商务合作: market@elibot.cn

技术咨询: technical@elibot.cn

- 苏州公司 (生产基地)

苏州市工业园区长阳街 259 号中新钟园工业坊 4 栋

+86-400-189-9358

- 北京公司

北京市经济技术开发区荣华南路 2 号院 6 号楼 1102 室

- 上海公司 (研创中心)

上海市浦东新区张江科学城学林路 36 弄 18 号

- 深圳公司

深圳市宝安区航空路泰华梧桐岛科技创新园 1A 栋 202 室

- 美国公司

10521 Research Dr., Ste. 104, 37932, Knoxville, TN (USA)

- 德国公司

Münchener Str. 53, 85290, Geisenfeld, Bavaria (Germany)

- 日本公司

TOSHIN Hirokoji Honmachi Bldg., 1F, 2-4-3 Sakae, Naka-ku, 460-0008, Nagoya (Japan)

- 墨西哥公司

Calzada del pedregal 523, fraccionamiento el pedregal



关注公众号了解更多