

ModbusTcp使用说明书

高速机器人专家
智能制造探索者

atomrobot
阿童木机器人

声 明

本手册适用于阿童木ModbusTcp。

本手册中与产品有关的规格和信息如有改动，恕不另行通知。本手册中提出的所有陈述、信息和建议均已经过慎重处理，但不保证完全正确。虽然在编制本手册时注意了一切可能事项，但对于仍然可能出现的错误和遗漏，本公司不承担任何责任。同样，由于使用本手册所包含信息而造成的损害本公司也不承担任何责任。用户必须对其应用任何产品负全部责任。

本手册所有内容的解释权归属阿童木机器人。

本手册未对任何一方授权许可，不得以任何方式复制和拷贝其中的全部或部分内容。

版权所有：阿童木机器人 @2020, All Rights Reserved

联系电话：400-653-7789/13752773699

咨询建议：wjf@tjchenxing.com

委托生产地址：江苏省苏州市昆山市张浦镇振新东路586号40号房。

版本变更记录

修订日期	发布版本	变更内容
2024-6	V1.0	• 整体内容更新

目 录

前言.....	01
第1章 ModbusTcp简介.....	02
1.1 报文头MBAP.....	02
1.2 帧结构PDU.....	02
1.3 Modbus功能码介绍及使用.....	03
1.3.1 线圈读取.....	03
1.3.2 读取离散输入.....	03
1.3.3 读保持寄存器.....	04
1.3.4 读取输入寄存器.....	05
1.3.5 写入单个线圈.....	06
1.3.6 写入单个寄存器.....	06
1.4 PDU详细结构.....	07
第2章 Atom控制器ModbusTcp介绍.....	09
2.1 主从站建立.....	09
2.2 支持数据类型.....	10
2.3 控制器控制状态反馈和控制支持.....	10
第3章 示例.....	13
3.1 获取机器人实时空间位置.....	13
3.2 自动运行程序.....	14
3.3 修改HMI界面变量数值.....	17

前 言

首先感谢您使用阿童木机器人ModbusTcp。

本手册主要内容包含了三个部分：ModbusTcp简介、Atom控制器ModbusTcp介绍、示例。

关于本手册：

本手册的读者及为使用该机型设备的厂家，包括安装、调试、维护该设备的人员。任何安装、调试、使用、维护该设备的人员必须得到本公司的培训及认真阅读过本手册才能进行有关设备的活动。

用户可通过阿童木机器人官方网站进行需求与使用咨询：www.tjchenxing.com

第1章 ModbusTcp简介

本系统采用标准ModbusTcp通讯协议。

Modbus设备可分为主站(master)和从站(slave)。主站只有一个，从站可以有多个，主站向各从站发送请求帧，从站给予响应。在使用TCP通信时，主站为client端，主动建立连接；从站为server端，等待连接。

主站请求：功能码+数据

从站正常响应：请求功能码+响应数据

从站异常响应：异常功能码+异常码，其中异常功能码即将请求功能码+0x80，异常码指示差错类型。

ModbusTCP的数据帧可分为两部分：MBAP+PDU。

1.1 报文头MBAP

MBAP为报文头，长度为7字节，组成如表1-1：

表 1-1 报文头

事务处理标识	协议标识	长度	单元标识符
2字节	2字节	2字节	1字节

MBAP的具体内容解释如表1-2：

表 1-2 报文头解释

内容	解释
事务处理标识	可以理解为报文的序列号，一般每次通信之后就要加1以区别不同的通信数据报文。
协议标识符	00 00表示ModbusTCP协议。
长度	表示接下来的数据长度，单位为字节。
单元标识符	可以理解为设备地址。

1.2 帧结构PDU

PDU由功能码+数据组成。功能码为1字节，数据长度不定，由具体功能决定。

功能码

Modbus的操作对象有四种：线圈、离散输入、保持寄存器、输入寄存器。

对象	含义
线圈	PLC的输出位，开关量，在Modbus中可读可写
离散量	PLC的输入位，开关量，在Modbus中只读
输入寄存器	PLC中只能从模拟量输入端改变的寄存器，在Modbus中只读
保持寄存器	PLC中用于输出模拟量信号的寄存器，在Modbus中可读可写

根据对象的不同，Modbus的功能码有：

对象	含义
0x01	读线圈
0x05	写单个线圈
0x0F	写多个线圈
0x02	读离散量输入
0x04	读输入寄存器
0x03	读保持寄存器
0x06	写单个保持寄存器
0x10	写多个保持寄存器

1.3 Modbus功能码介绍及使用

1.3.1 线圈读取

此功能代码用于读取远程设备中线圈的1到2000个连续状态。请求PDU指定起始地址，即指定的第一个线圈的地址和线圈数量。在PDU中，线圈从零开始寻址。因此，编号为1-16的线圈被寻址为0-15。响应消息中的线圈被打包为每个数据字段位一个线圈。状态指示为1表示开和状态为0表示关。

客户端发出请求（Request）

功能码	1 Byte	0x01
起始地址	2 Byte	0x0000 to 0xFFFF
线圈数量	2 Byte	1 to 2000 (0x7D0)

服务端响应（Response）

功能码	1Byte	功能码 (0x01)+0x80
字节计算	1Byte	N
线圈状态	nByte	N=N or N+1

异常响应（Error）

错误码	1Byte	功能码（0x01）+0x80
异常代码	1Byte	01 or 02 or 03 or 04

举例如下：

客户端发送请求		服务端响应	
字段名称	Hex	字段名称	Hex
功能码	01	功能码	01
地址高8位	00	字节计算	03
地址低8位	13	输出状态1	CD
输入字节数量高8位	00	输出状态2	6B
输入字节数量低8位	13	输出状态3	05

注1：地址有两个字节组成，其中高8位表示标识寄存器或线圈在设备中的位置的参数，低8位表示相对于寄存器相对位置，以此表示所读和所写存储位置。对于服务端响应部分，其中输出状态由发送请求线圈的数量决定，列如本例中共需13个线圈状态，故输出状态数量 $N = (0x13) 19/8$ ，如果余数不为0， $N=N+1$ ，故 $N=3$ 。此外，服务端字节计算部分应与输出状态数量N保持一致，故字节数量（Byte count）=3。

1.3.2 读取离散输入

此功能代码用于读取远程设备中离散输入的1到2000个连续状态。在PDU离散输入中，从零开始重新寻址。因此，编号为1-16的离散输入被寻址为0-15。响应消息中的离散输入打包为数据字段的每个位一个输入。

客户端请求发送（Request）

功能码	1Byte	0x02
起始地址	2Byte	0x0000 to 0xFFFF
输入数量	2Byte	1 to 2000 (0x7D0)

服务端响应（Respose）

功能码	1Byte	0x02
字节计算	1Byte	N
输入状态	N	

异常响应（Error）

错误码	1Byte	0x82
异常代码	1Byte	01 or 02 or 03 or 04

下面是客户端读取离散输入 197-218 的请求示例

客户端发送请求		服务端响应	
字段名称	Hex	字段名称	Hex
功能码	02	功能码	02
地址高8位	00	字节计算	03
地址低8位	C4	输入状态 204-197	AC
输入字节数量高8位	00	输入状态212-205	DB
输入字节数量低8位	16	输入状态218-213	35

注2：离散读取与线圈读取方式大致类同，具体计算请看注1，后续不一一说明。输入离散输入204-197是以16进制方式显示为字节值AC十六进制，或二进制1010 1100。输入204是该字节的MSB，输入197是LSB。离散输入218-213的状态是以16进制方式显示为35形式，或二进制0011 0101。输入218（MSB）位于左起第三位位置，输入213为LSB。

1.3.3 读保持寄存器

此功能代码用于读取远程设备中保持寄存器的连续块的内容。请求PDU指定起始地址和寄存器数量。在PDU中，寄存器从零开始寻址。因此，编号为1-16的寄存器被寻址为0-15。响应消息中的寄存器数据打包为每个寄存器两个字节，二进制内容在每个字节内对齐。对于每个寄存器，第一个字节包含高位，第二个字节包含低位。

客户端发出请求（Request）

功能码	1 Byte	0x03
起始地址	2 Byte	0x0000 to 0xFFFF
寄存器数量	2 Byte	1 to 125 (0x7D)

服务端响应（Respose）

功能码	1Byte	0x03
起始地址	1Byte	2×N
寄存器值	N×2Bytes	

异常响应（Error）

错误码	1Byte	0x83
异常码	1Byte	01 or 02 or 03 or 04

下面是客户端读取寄存器 108 – 110 的请求示例：

客户端发送请求		服务端响应	
字段名称	Hex	字段名称	Hex
功能码	03	功能码	03
地址高8位	00	字节计算	06
地址低8位	6B	寄存器108高8位	02
输入寄存器数量高8位	00	寄存器108低8位	2B
输入寄存器数量低8位	03	寄存器109高8位	00
——		寄存器109低8位	00
——		寄存器110高8位	00
——		寄存器110低8位	64

注3：寄存器108的内容以十六进制方式显示为022B。寄存器109的内容以十六进制方式显示为0000，同理，寄存110以16进制方式显示为0000和0064，--代表空项。

1.3.4 读取输入寄存器

此功能代码用于读取远程设备中的1到125个连续输入寄存器。请求PDU指定起始寄存器地址和寄存器数量。在PDU中，寄存器从零开始寻址。因此，输入寄存器numbered1-16被寻址为0-15。响应消息中的寄存器数据打包为每个寄存器两个字节，二进制内容在每个字节内对齐。对于每个寄存器，第一个字节包含高阶位，第二个字节包含低阶位。

客户端发出请求（Request）

功能码	1 Byte	0x04
起始地址	2 Byte	0x0000 to 0xFFFF
输入寄存器数	2 Byte	0x0001 to 0x007

服务端响应（Response）

功能码	1Byte	0x04
字节计算	1Byte	2×N
寄存器状态显	N×2Bytes	

异常响应（Error）

错误码	1Byte	0x84
异常码	1Byte	01 or 02 or 03 or 04

下面是客户端读取2输入寄存器9的请求示例

客户端发送请求		服务端响应	
字段名称	Hex	字段名称	Hex
功能码	04	功能码	04
起始地址高8位	00	字节计算	02
起始地址低8位	08	寄存器9高8位	00
寄存器数量高8位	00	寄存器9低8位	0A
寄存器数量低8位	01	——	

1.3.5 写入单个线圈

此函数代码用于将单个输出写入远程设备中的ON或OFF。请求的ON/OFF状态由请求数据字段中的常量指定。值FF00十六进制要求输出为ON。值0000请求将其关闭。所有其他值都是非法的，不会影响输出。请求PDU指定要强制的线圈的地址。线圈从零开始寻址。因此，编号为1的线圈被寻址为0。请求的开/关状态由线圈值字段中的常量指定。值为0xFF00要求线圈打开。值为0x0000要求线圈关闭。所有其他值都是非法的，不会影响线圈。

客户端发出请求（Request）

功能码	1 Byte	0x05
指定的寄存器地址	2 Byte	0x0000 to 0xFFFF
指令录入量	2 Byte	0x0000 or 0xFF00

服务端响应（Response）

功能码	1Byte	0x05
指定寄存器的地址	2Bytes	0x0000 to 0xFFFF
服务端反馈值示	2Bytes	0x0000 to 0xFFFF

注4：正常情况下服务端反馈值与录入指令应该相同。

异常响应（Error）

错误码	1Byte	0x85
异常码	1Byte	01 or 02 or 03 or 04

下面是客户端将线圈173写入的请求示例

客户端发送请求		服务端响应	
字段名称	Hex	字段名称	Hex
功能码	05	功能码	05
起始地址高8位	00	服务端反馈地址高8位	00
起始地址低8位	AC	服务端反馈地址低8位	AC
指令录入高8位	FF	服务端反馈地址高8位	FF
指令录入低8位	00	服务端反馈地址低8位	00

1.3.6 写入单个寄存器

此功能代码用于在远程设备中写入单个保持寄存器。请求PDU指定要写入的寄存器的地址。寄存器从零开始寻址。因此，编号为1的寄存器被寻址为0。正常响应是请求的回显，在寄存器内容写入后返回。

客户端发出请求（Request）

功能码	1Byte	0x06
寄存器地址	2Byte	0x0000 to 0xFFFF
寄存器值	2Byte	0x0000 to 0xFFFF

服务端响应（Response）

功能码	1Byte	0x06
寄存器地址	2Bytes	0x0000 to 0xFFFF
寄存器值	2Bytes	0x0000 to 0xFFFF

异常响应 (Error)

错误码	1Byte	0x86
异常码	1Byte	01 or 02 or 03 or 04

下面是将寄存器2写入0003十六进制的请求示例

客户端发送请求		服务端响应	
字段名称	Hex	字段名称	Hex
功能码	06	功能码	06
起始地址高8位	00	起始地址高8位	00
起始地址低8位	01	起始地址低8位	01
寄存器值高8位	00	寄存器值高8位	00
寄存器值低8位	03	寄存器值低8位	03

1.4 PDU详细结构

0x01: 读线圈

在从站中读1~2000个连续线圈状态，ON=1,OFF=0。

请求: MBAP功能码起始地址H起始地址L数量H数量L (共12字节)

响应: MBAP功能码数据长度数据 (一个地址的数据为1位)

如: 在从站0x01中, 读取开始地址为0x0002的线圈数据, 读0x0008位00 01 00 00 00 06 01 01 00 02 00 08

回: 数据长度为0x01个字节, 数据为0x01, 第一个线圈为ON, 其余为OFF 00 01 00 00 00 04 01 01 01 01

0x05: 写单个线圈

0x05: 写单个线圈

请求: MBAP 功能码 输出地址H 输出地址L 输出值H 输出值L (共12字节)

响应: MBAP 功能码 输出地址H 输出地址L 输出值H 输出值L (共12字节)

如: 将地址为0x0003的线圈设为ON 00 01 00 00 00 06 01 05 00 03 FF 00

回: 写入成功 00 01 00 00 00 06 01 05 00 03 FF 00

0x0F: 写多个线圈

0x0F: 写多个线圈

■ 将一个从站中的一个线圈序列的每个线圈都强制为ON或OFF, 数据域中置1的位请求相应输出位ON, 置0的位请求响应输出为OFF

请求: MBAP 功能码 起始地址H 起始地址L 输出数量H 输出数量L 字节长度 输出值H 输出值L

响应: MBAP 功能码 起始地址H 起始地址L 输出数量H 输出数量L

0x02: 读离散量输入

0x02: 读离散量输入

请求: MBAP 功能码 起始地址H 起始地址L 数量H 数量L (共12字节)

响应: MBAP 功能码 数据长度 数据 (长度: 9+ceil(数量/8))

如: 从地址0x0000开始读0x0012个离散量输入00 01 00 00 00 06 01 02 00 00 00 12

回: 数据长度为0x03个字节, 数据为0x01 04 00, 表示第一个离散量输入和第11个离散量输入为ON, 其余为OFF 00 01 00 00 00 06 01 02 03 01 04 00

0x04: 读输入寄存器

0x04: 读输入寄存器

请求: MBAP 功能码 起始地址H 起始地址L 寄存器数量H 寄存器数量L (共12字节)

响应: MBAP 功能码 数据长度 寄存器数据(长度: 9+寄存器数量×2)

如：读起始地址为0x0002，数量为0x0005的寄存器数据 00 01 00 00 00 06 01 04 00 02 00 05

回：数据长度为0x0A，第一个寄存器的数据为0x0c，其余为0x00 00 01 00 00 00 0D 01 04 0A 00 0C 00 00 00 00 00 00 00 00

■ 0x03：读保持寄存器

从远程设备中读保持寄存器连续块的内容

请求：MBAP 功能码 起始地址H 起始地址L 寄存器数量H 寄存器数量L（共12字节）

响应：MBAP 功能码 数据长度 寄存器数据(长度：9+寄存器数量×2)

如：起始地址是0x0000，寄存器数量是 0x0003 00 01 00 00 00 06 01 03 00 00 00 03

回：数据长度为0x06，第一个寄存器的数据为0x21，其余为0x00 00 01 00 00 00 09 01 03 06 00 21 00 00 00 00

■ 0x06：写单个保持寄存器

在一个远程设备中写一个保持寄存器

请求：MBAP 功能码 寄存器地址H 寄存器地址L 寄存器值H 寄存器值L（共12字节）

响应：MBAP 功能码 寄存器地址H 寄存器地址L 寄存器值H 寄存器值L（共12字节）

如：向地址是0x0000的寄存器写入数据0x000A 00 01 00 00 00 06 01 06 00 00 00 0A

回：写入成功 00 01 00 00 00 06 01 06 00 00 00 0A

■ 0x10：写多个保持寄存器

在一个远程设备中写连续寄存器块（1~123个寄存器）

请求：MBAP 功能码 起始地址H 起始地址L 寄存器数量H 寄存器数量L 字节长度 寄存器值（13+寄存器数量×2）

响应：MBAP 功能码 起始地址H 起始地址L 寄存器数量H 寄存器数量L（共12字节）

如：向起始地址为0x0000，数量为0x0001的寄存器写入数据，数据长度为0x02，数据为0x000F 00 01 00 00 00 09 01 10 00 00 00 01 02 00 0F

回：写入成功 00 01 00 00 00 06 01 10 00 00 00 01

第2章 Atom控制器ModbusTcp介绍

Atom控制器的ModbusTcp功能内嵌在PLC软件中。启动界面选择启动PLC软件后可以进行ModbusTcp的配置。

2.1 主从站建立

Atom控制器的ModbusTcp功能支持用户建立多个主站和从站，与不同的外部设备同时进行通信。

添加Modbus支持

选择界面的+号，添加Modbus支持，如图 2.1。

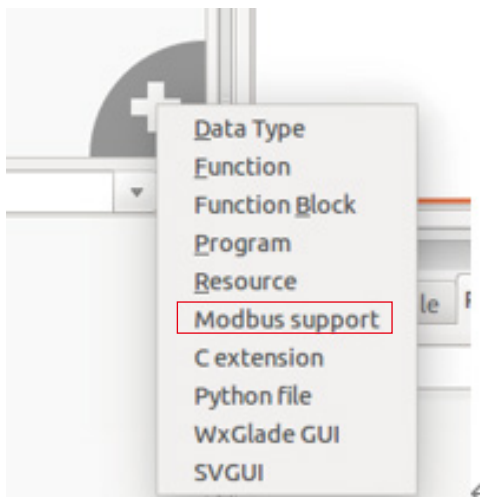


图 2.1 Modbus支持

建立主站/从站

点击生成Modbus节点。可添加ModbusTCPserver(从站)/ ModbusTCPclient(主站)，如图 2.2。

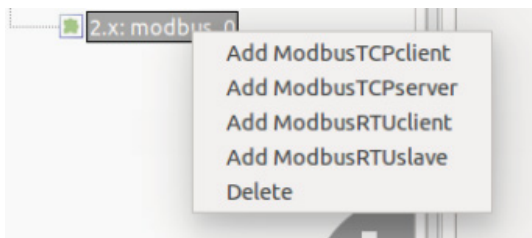


图 2.2 建立主/从站

IP地址可设置为默认，所有网口都可进行连接。也可设置为需要进行通信的网口IP地址。设置从站ID和端口号(端口号控制器内部已经占用502，因此可自定义502以外的端口)，如图 2.3。

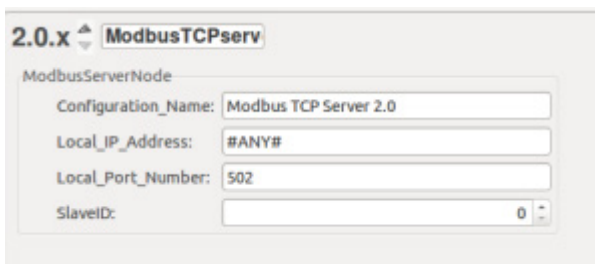


图 2.3 设置id、端口号

以ModbusTCPserver为例，在ModbusTCPserver_0节点上右键，增加MemoryArea。如图 2.4。



图 2.4 添加MemoryArea

ModbusTCPserver的存储区域类型为01-线圈，02-离散输入，03-保持寄存器，04-输入寄存器。如图 2.5。

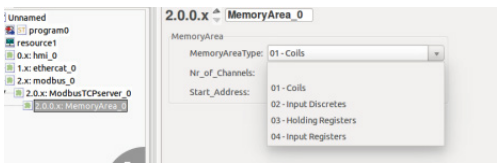


图 2.5 ModbusTCPserver的MemoryArea

ModbusTCPclient数据请求类型与功能码一致，如图 2.6。

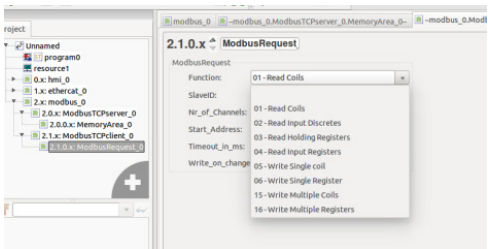


图 2.6 ModbusTCPclient的MemoryArea

2.2 支持数据类型

从上述主从站数据类型可以看出，目前ModbusTCP的数据支持类型为：bool和word。

用户通过ModbusTcp将bool和word数据传送到PLC，PLC收到对应数据后进行相关操作。

需要注意的是，目前的PLC无法处理小数。因为real类型变量在数据从word进行转换时会丢失小数部分，因此建议需要传送小数的部分根据精度将相关数据*10/*100/*1000转换为整数后写入word。对当前位置的处理如果有精度要求，需要将数值*10/*100/*1000。

2.3 控制器控制状态反馈和控制支持

控制器通过PLC开放了部分状态显示和控制变量，具体支持内容如下：

如图 2.7为机器人状态变量：

#	Name	Class	Type	Location
0	RunState	IntOutput	DINT	
1	EnableStatus	BoolOutput	Bool	
2	AlarmCode	IntOutput	DINT	
3	EmergencyStatus	BoolOutput	Bool	
4	ReturnHomeStatus	BoolOutput	Bool	
5	RunMode	IntOutput	DINT	
6	Production	IntOutput	DINT	
7	StopContinueMode	BoolOutput	Bool	
8	CurrentJoint1	RealOutput	LREAL	
9	CurrentJoint2	RealOutput	LREAL	
10	CurrentJoint3	RealOutput	LREAL	
11	CurrentJoint4	RealOutput	LREAL	
12	CurrentJoint5	RealOutput	LREAL	
13	CurrentJoint6	RealOutput	LREAL	
14	CurrentWorldX	RealOutput	LREAL	
15	CurrentWorldY	RealOutput	LREAL	
16	CurrentWorldZ	RealOutput	LREAL	
17	CurrentWorldA	RealOutput	LREAL	
18	CurrentWorldB	RealOutput	LREAL	
19	CurrentWorldC	RealOutput	LREAL	
20	CurrentProgram	IntOutput	DINT	
21	CurrentLineNumber	IntOutput	DINT	

图 2.7 机器人状态变量

表 2-1为机器人状态变量变量对应的系统状态：

表 2-1 机器人状态变量对应系统状态

变量名称	控制器对应状态	说明
RunStatus	运行状态	1：运行；2：暂停；3：停止；4：停止中
Enable status	使能状态	0：未使能；1：使能
Alarm code	报警代码	10进制报警代码
Emergency status	急停状态	1：急停；0：未急停
Return home status	零点状态	1：位于零点；0：未位于零点
Run mode	运行模式	1：自动；2：手动
Production	产量	需要功能块关联产量变量
Step continue mode	单步连续模式	1：单步；2：连续
Current joint1-6	关节1-6的位置	
Current worldX/Y/Z/A/B/C	笛卡尔空间坐标坐标系位置	
Current program	当前加载程序	对应程序编号
Current line number	当前执行的行号	

如图 2.8为机器人控制变量：

#	Name	Class	Type	Location
0	Sync.PlcControlEnable	BoolInput	Bool	
1	Sync.ManualEnable	BoolInput	Bool	
2	Sync.ExternalAxisActive	BoolInput	Bool	
3	Sync.RunMode	IntInput	DINT	
4	Sync.StepContinueMode	BoolInput	Bool	
5	Sync.VelocityPercent	IntInput	DINT	
6	Sync.ExternalVelocityPerce	IntInput	DINT	
7	Sync.RobotAxisStart	BoolInput	Bool	
8	Sync.ExternalAxisStart	BoolInput	Bool	
9	Rising.AutoModeEnable	BoolInput	Bool	
10	Rising.ClearAllAlarm	BoolInput	Bool	
11	Rising.ClearAlarm	BoolInput	Bool	
12	Rising.StartProgram	BoolInput	Bool	
13	Rising.StopProgram	BoolInput	Bool	
14	Rising.PauseProgram	BoolInput	Bool	
15	Rising.ReturnHome	BoolInput	Bool	
16	Rising.SetLineNumber	BoolInput	Bool	
17	Rising.SetHome	BoolInput	Bool	
18	Rising.LoadProgram	BoolInput	Bool	
19	Register.RobotAxisIndex	IntInput	DINT	
20	Register.RobotAxisDirectio	BoolInput	Bool	
21	Register.ExternalAxisindex	IntInput	DINT	
22	Register.ExternalAxisDirec	BoolInput	Bool	
23	Register.ClearAlarmCode	IntInput	DINT	
24	Register.StartProgramNurr	IntInput	DINT	
25	Register.SetLineNumber	IntInput	DINT	
26	Register.SetHomeindex	IntInput	DINT	
27	Register.LoadProgramNurr	IntInput	DINT	
28	Register.RobotCartSys	IntInput	DINT	

图 2.8 机器人控制变量

表 2-2为机器人控制变量前缀格式说明：

表 2-2 机器人控制变量变量前缀

变量名称	说明
Sync	用于同步控制，比如RunMode，机器人的运行模式会与该变量表示的模式保持同步。 PlcControlEnable：这个变量为true时用于启用plc控制，否则不启用plc控制。 因此当开启plc控制权后需要在plc程序中将该值置为true。
Rising	用于上升沿控制，控制系统会检测这种类型变量的上升沿， 只有产生了一次上升沿时才会认为进行了一次控制。
Register	寄存器变量，用于在控制机器人时需要输入给机器人的参数。

表 2-3为RobotControl对应的系统控制信号，这些控制信号需要控制器处于PLC控制并且PlcControlEnable置为TRUE后才生效。RobotStatus读取状态可实时读取，不受上述条件约束。需要注意的是Rising信号，需要给上升沿才会触发，触发后需要复位后，再次给出上升沿才可以继续触发(该功能类似按钮被按下和松开的操作)。

表 2-3 机器人控制变量对应系统控制信号

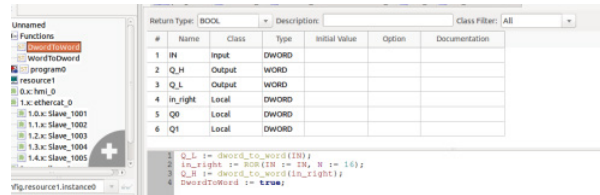
变量名称	控制器对应触发	说明
Sync.PlcControlEnable	PLC是否开启PLC控制	True: 开启; False: 关闭
Sync.ManualEnable	手动使能	True: 上使能; False: 下使能
Sync.ExternalAxisActive	外部轴指令执行开启	True: 开启; False: 关闭
Sync.RunMode	设置运行模式	1: 自动; 2: 手动
Sync.StepContinueMode	设置单步连续模式	True: 单步; False: 连续
Sync.VelocityPercent	设置速度百分比	
Sync.ExternalVelocityPercent	设置外部轴运行速度百分比	
Sync.RobotAxisStart	开始/关闭点动	需配合点动轴号+方向使+点动坐标系使用
Sync.ExternalAxisStart	开始点动点动外部轴点动	需配合外部轴点动轴号+方向使用
Rising.AutoModeEnable	自动使能	
Rising.ClearAllAlarm	清除所有错误	
Rising.ClearAlarm	清除指定错误	需配合清错报警代码使用
Rising.StartProgram	开启程序运行	
Rising.StopProgram	停止程序运行	
Rising.PauseProgram	暂停程序运行	
Rising.ReturnHome	回零	
Rising.SetLineNumber	设置行号	需配合设置的行号使用
Rising.SetHome	设零	需配合设零的轴号使用
Rising.LoadProgram	加载程序	需配合加载的程序号使用
Register.RobotAxisIndex	点动轴号	从0开始。世界坐标系: 0对应X方向。 世界坐标系: 1对应X方向。 关节坐标系: 2对应关节1。 轴坐标系: 0对应轴1。依次类推。
Register.RobotAxisDirection	点动方向	True: 正向; False: 负向
Register.ExternalAxisIndex	外部轴点动轴号	从0开始, 0对应外部轴1
Register.ExternalAxisDirection	外部轴点动方向	True: 正向; False: 负向
Register.ClearAlarmCode	要清错的报警代码	
Register.StartProgramNumber	加载的程序号	
Register.SetLineNumber	设置的行号	
Register.SetHomeIndex	设零轴号	Index从0开始。0对应1轴
Register.LoadProgramNumber	加载的程序号	
Register.RobotCartSys	点动坐标系	1: 世界坐标系; 2: 基坐标系; 3: 关节坐标系; 4: 轴坐标系

第3章 示例

3.1 获取机器人实时空间位置

本示例以Z位置为例，其他位置按照此示例即可获得。

示例前提，已经新建Function: DwordToWord，将一个Dword拆分为两个word。

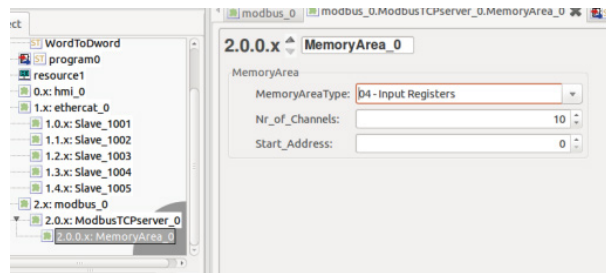


1.打或新建工程，URI路径配置完成，添加Modbus支持。

2.生成Modbus节点。添加ModbusTcpServer。

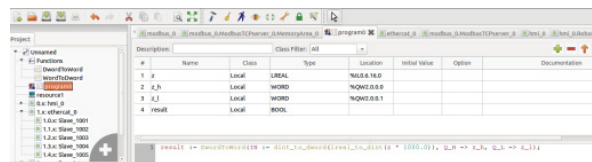
3.设置IP地址和端口号(50280)，设置设备ID为1。

4.在ModbusTCPserver_0节点上右键，增加MemoryArea。设置区域类型为04-输入寄存器，通道号10，起始地址0。

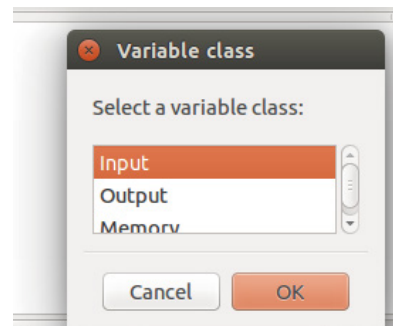
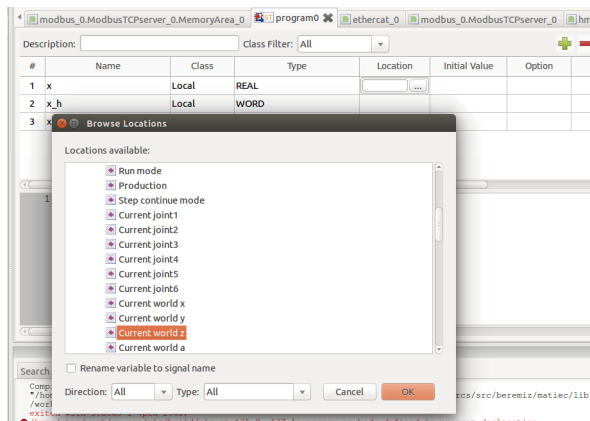


5.新建如下变量和程序：

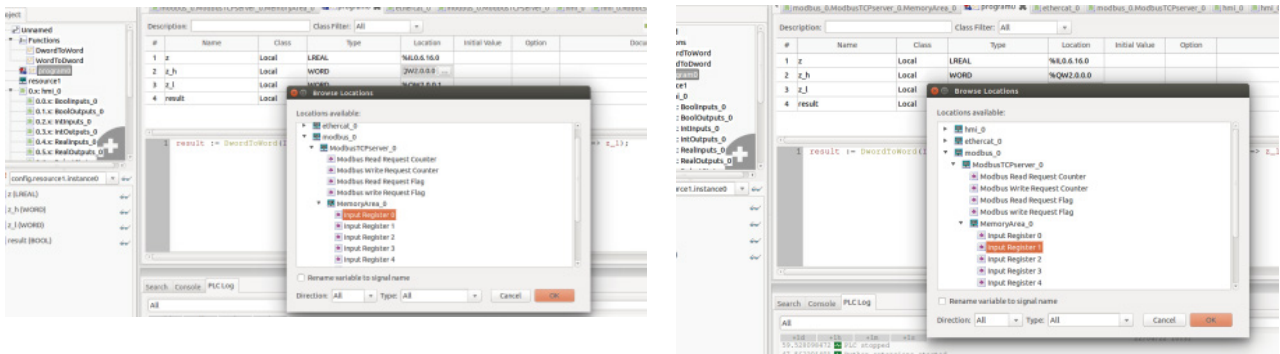
z表示当前z坐标。z_h表示写入ModbusTcp中的位置高16位。z_l表示写入ModbusTcp中的位置低16位。



6.将z关联机器人状态RobotStatus中的Current word z变量引脚，数据类型选择输入：

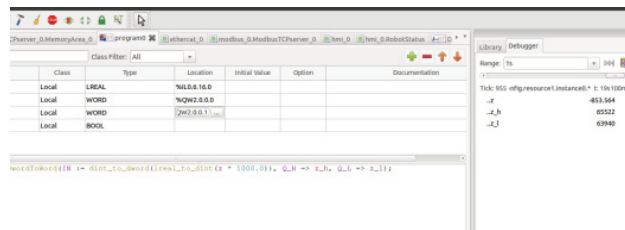


将zh、zl分别关联ModbusTcp输入寄存器的地址0和1。数据类型选择输出。



7.编译运行程序。

8.监控当前变量数值



示教器坐标数值如下图：

系 统	新程序 新建项目 删除 撤销 打开 加载 卸载 其他						
调 试	X:	0.000 mm	Y:	0.000 mm	Z:	-853.564 mm	基坐标系
	A:	0.000 °	B:	0.000 °	C:	0.000 °	默认工具

机器人实时空间Z坐标为-853.564。

输入寄存器地址0数值为：65522(16进制0xff2)，

地址1数值为：63940(16进制0xf9c4)。

高16位低16组合后数值为-853564(16进制0xffff2 f9c4) = -853.564 * 1000.0。

9.外部主站可读取输入寄存器0和1的数值，组合后除以1000.0即为实时Z位置。

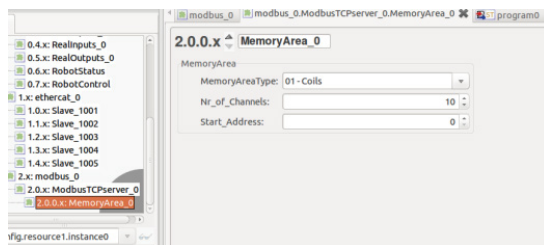
3.2 自动运行程序

示例前提：

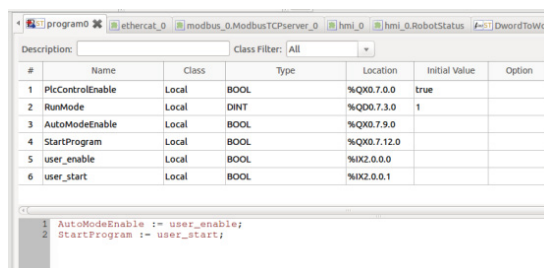
已经将HMI控制权设置为PLC控制：



- 1.打或新建工程，URI路径配置完成，添加Modbus支持。
- 2.生成Modbus节点。添加ModbusTcpServer。
- 3.设置IP地址和端口号(50280)，设置设备ID为1。
- 4.在ModbusTCPserver_0节点上右键，增加MemoryArea。设置区域类型为01-线圈，通道号10，起始地址0。

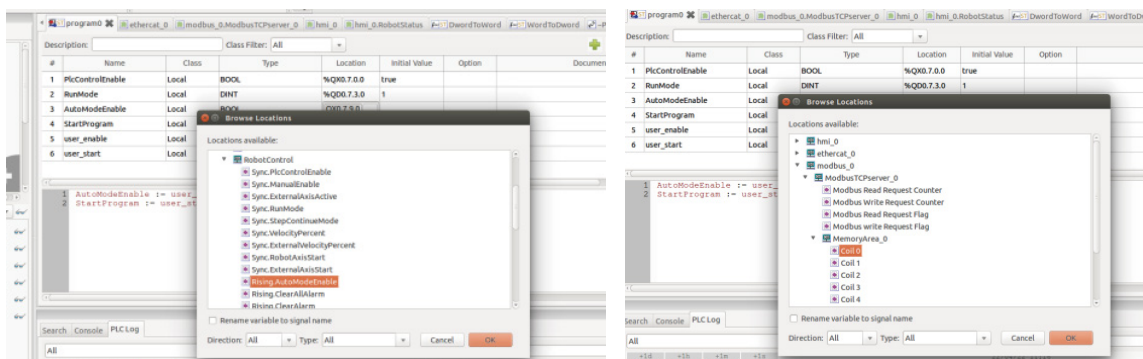


- 5.新建如下变量和程序：



变量说明:

变量名称	控制器对应触发	说明
PlcControlEnable	Sync.PlcControlEnable	是否开启PLC控制，初始值设置为true，表示开启PLC控制。 关联时选择变量类型为Output。
RunMode	Sync.RunMode	设置运行模式，初始值设置为1，表示自动模式。 关联时选择变量类型为Output。
StartProgram	Rising.StartProgram	开启程序运行(上升沿触发状态变化)关联时选择变量类型为Output。
AutoModeEnable	Rising.AutoModeEnable	自动使能(上升沿触发状态变化)关联时选择变量类型为Output。
user_enable	ModbusTcp线圈地址0	用户设置使能，关联时选择变量类型为Input。
user_start	ModbusTcp线圈地址1	用户设置开始，关联时选择变量类型为Input。

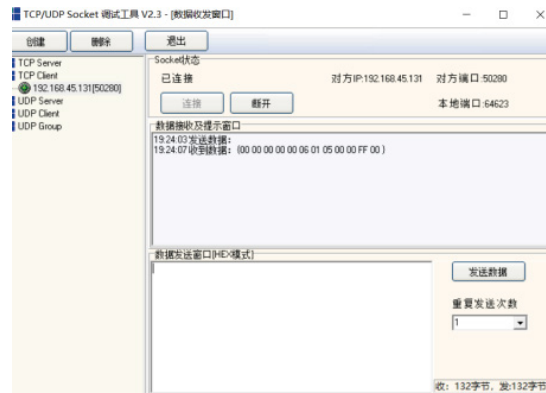


- 6.编译运行程序。

- 7.通过外部电脑连接工控机网口(设置ModbusTcp IP地址的网口)，使用网络调试助手，新建对应IP和端口的客户端，连接。

地址_Coils(01)	功能	说明
0	自动使能	00000000000601050000ff00 000000000006010500000000
1	开始	00000000000601050001ff00 000000000006010500010000

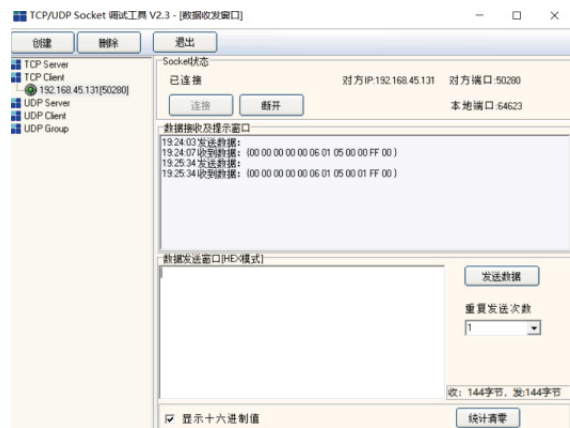
16进制发送自动使能上升沿：
00000000000601050000ff00，如图：



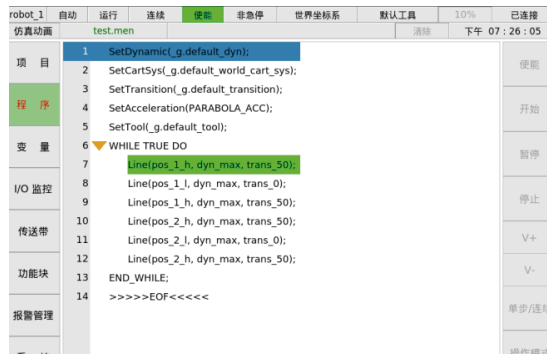
控制器进入使能状态：



16进制发送自动使能上升沿：
00000000000601050001ff00，如图：



控制器进入运行状态：



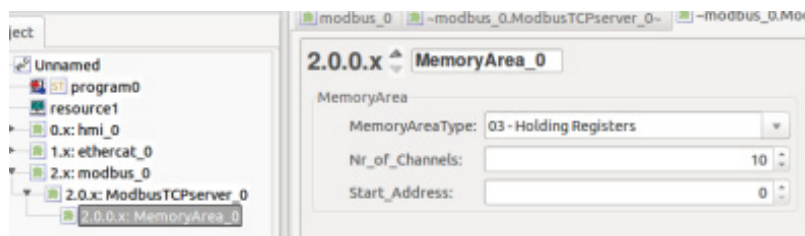
3.3 修改HMI界面变量数值

示例前提：

已经将HMI控制权设置为PLC控制：



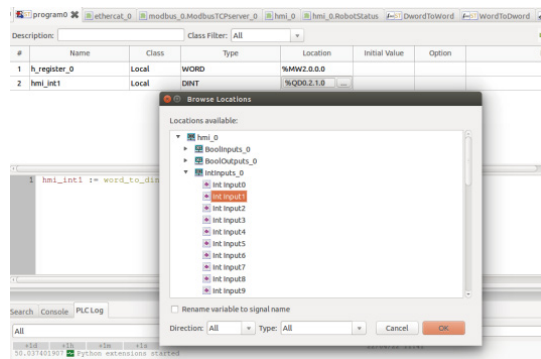
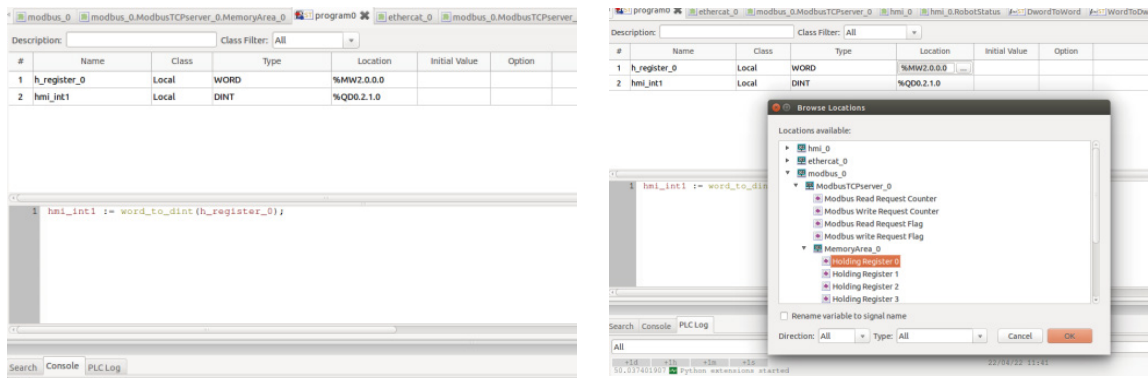
- 1.打或新建工程，URI路径配置完成，添加Modbus支持。
- 2.生成Modbus节点。添加ModbusTcpServer。
- 3.设置IP地址和端口号(50280)，设置设备ID为1。
- 4.在ModbusTCPserver_0节点上右键，增加MemoryArea。设置区域类型为03-保持寄存器，通道号10，起始地址0。



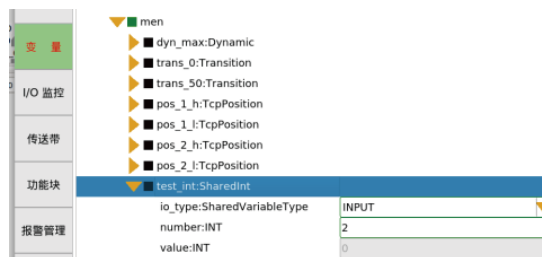
5.新建如下变量和程序：





word变量h_register_0，通过位置属性连接到Holding Register0。关联时选择变量类型为Memory。

Hmi_int1通过位置属性连接到IntInputs_0的1号引脚。关联时选择变量类型为Output。



6.HMI新建名称为test_int的SharedInt(需要在HMI控制权下)。



7.点击  按钮进行构建。点击  连接按钮(需要开启控制器)。点击  传送按钮。点击  启动按钮启动PLC。

8.通过外部电脑连接工控机网口(设置ModbusTcp IP地址的网口)，使用网络调试助手进行测试，助手新建TcpClient，IP地址输入控制器和TCP助手通讯所用网口地址(需要TCP助手电脑与控制器在同一网段)，输入设置的端口号。连接成功后，发送数据修改保持寄存器地址0的数据，监控HMI的test_int变量

修改发送示例(16进制):

协议格式: 事务标识符(2byte) + 协议标识符(2byte) + 合计字节数(2byte) + 单元标识符和功能码(2byte) + 开始位置(2byte) + 写数量(2byte) + 写字节数(1byte) + 写字节数*1byte;

写字节数= 写数量*2

000000000009011000000001020064

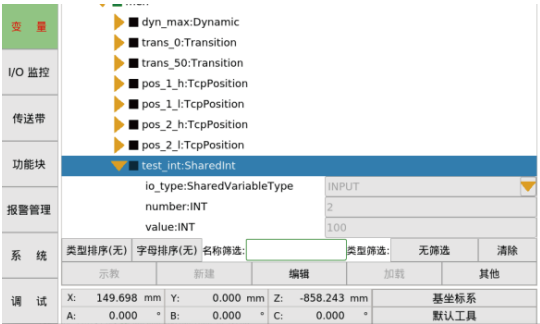
01: 设备ID

10: 功能码

0000: 起始地址



9.HMI变量的值被修改为100。



阿童木机器人
www.tjchenxing.com
400-653-7789

天津总部

辰星(天津)自动化设备有限公司
天津滨海新区泰达智能无人装备产业园29号厂房
400-653-7789

苏州子公司

辰星(苏州)自动化设备有限公司
江苏省苏州市吴江经济技术开发区交通南路1268号
0512-63161326

深圳子公司

深圳小百自动化科技有限公司
深圳宝安区西乡街道华丰机器人产业园 C栋1楼厂房
0755-23148852

昆山子公司

江苏小野智能装备有限公司
昆山市张浦镇振新东路振新东路浩盛工业园 C-6
0512-87886505

成都子公司

四川省成都市郫都区蜀新大道306号
汇创天下科技园B210

