UNIVERSAL ROBOTS

UR 机器人与 PC 通讯

版本 1.0.1
2017 年 4 月

	创建时间	编辑人员	修改内容
1	2017-3	CSL	创建 v1.0.0
2	2017-4	CSL	Add ModbusTCP

目 录

1. 总述

UR 机器人与 PC 通讯主要介绍 UR 机器人与工控机，具备 TCP/IP 通讯模块的 PLC 以及嵌入式系统等等如何进行通讯。意在让读者了解 UR 机器人的通讯原理，从而**根据实际需求选择最佳的通讯方式**。对于每种通讯方式的详细介绍，本文不涉及，需要的读者可以登录 UR 官网技术支持的 How-to articles 中进行搜索和下载。

早期版本的 UR 机器人版本提供了 C-API 接口，目前版本的 UR 机器人（CB3 及之上）不再支持；关于 ROS 接口，其实是在 UR 提供的接口上封装的一层，感兴趣的请参考 ROS 官网上的项目。

UR 机器人提供的通讯接口如图 1 所示，接口的比较见表格 1。

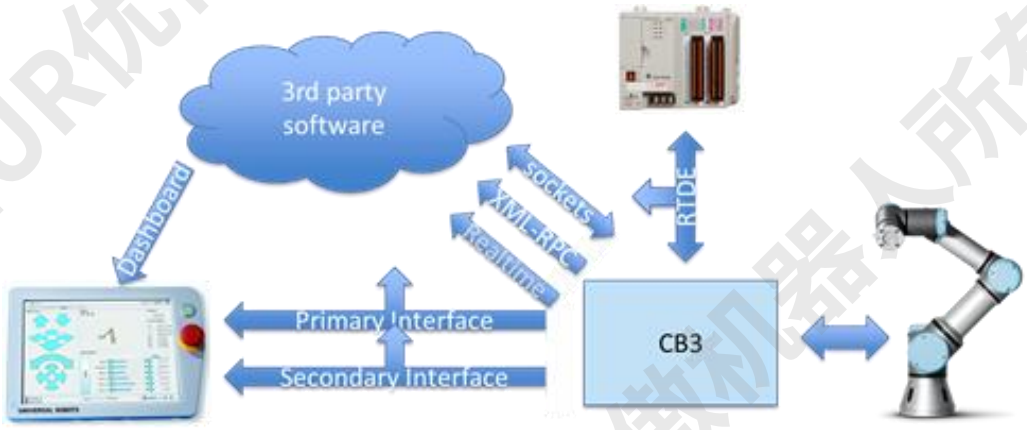

图 1、UR 机器人通讯接口

表格 1、UR机器人通讯接口

	Dashboard	Primary	Secondary	Realtime	RTDE	Socket	XML-RPC
发送内容[1]	返回结果字符串	机器人状态以及附加信息	机器人状态（位置、速度等等）	机器人状态（位置、速度等等）	机器人状态（位置、速度等）	任意	任意
接收内容[2]	Dashboard 命令	脚本指令	脚本指令	脚本指令	想设置的信息	任意	任意
端口号	29999	30001	30002	30003	30004	任意空闲端口	任意空闲端口
频率(Hz)	--	10	10	125	125	--	--

UR 机器人目前支持 Modbus-TCP, EthernetIP 以及 Profinet 三种总线，其中 UR 机器人自带 Modbus-TCP 服务器，PC 或者外部智能设备可通过该端口获取机器人的状态信息。

2. 通讯接口介绍

各通讯接口相互独立，可以同时使用；下面将对各通讯接口进行详细介绍。

[1] 发送内容：机器人向外发送的信息
[2] 接收内容：机器人从上位机接收的信息

2.1 Dashboard（29999）接口

Dashboard 接口是由机器人人机界面进程（示教器显示）负责维护和执行的一个端口。该端口主要负责接收上位机指令，**执行机器人初始化、加载程序、开始和暂停程序运行以及设置用户角色等**操作，上位机可以远程操作机器人就如同操作示教器一样。该接口主要应用在自动启动机器人，无示教器应用（无示教器情况下如何设置 UR 机器人请参考无示教器设置）等场合。

Dashboard 接口接收上位机发送的 Dashboard 命令字符串，机器人接收后，返回执行结果字符串。详细请参考 Dashboard 使用。

2.2 Primary & Secondary（30001 & 30002）接口

Primary & Secondary 接口由机器人控制进程维护和执行的端口。开机后，机器人一直从这两个端口以 10Hz 频率对外发送机器人的状态信息（机器人位置，IO，运行状态等），除此之外 Primary 端口还会发送一些全局变量更新等信息。因此如果需要编写上位机在线编程以及运行监视，例如编写一个类似示教器界面的上位机程序时可以使用 Primary 端口（当然对于加载程序，运行程序等操作需要使用 Dashboard（29999）接口）。

Primary & Secondary 接口在接收到上位机发送的脚本指令（脚本指令下载）字符串后会立即中断当前执行程序（如果机器人正在运行），然后运行接收到的脚本指令。值得提醒的是，一些具有返回值的脚本指令，其返回值并不会通过这两个端口返回，因为脚本的执行是在机器人控制进程中，其返回值只传递给该进程中的变量。

如果想一次性发送一段程序给机器人，需要将发送的程序按照如下格式发送：

```
def functionName():
  脚本指令
  脚本指令
  脚本指令
  #对于函数或者 while 等结构语句
  #需注意同层次具有相同缩进
  脚本指令
  ……
end
```

1. Primary & Secondary

```
sec functionName():
  非运动脚本指令
  非运动脚本指令
  非运动脚本指令
  #对于函数或者 while 等结构语句
  #需注意同层次具有相同缩进
  非运动脚本指令
  ……
end
```

2. Secondary

```
实际发送字符串(缩进用空格，换行用\n):
“def functionName():\n  脚本指令\n  脚本指令\n ……\nend\n “
“sec functionName():\n  脚本指令\n  脚本指令\n ……\nend\n “
```

如果向 Secondary 接口发送 def 类型的一段脚本给机器人，机器人当前执行的程序会被中断，如果需要当前执行的程序不被中断（例如设置 IO），可以向 Secondary 程序发送 sec 类型的一段脚本，当然这段脚本中不能够包含运动指令，因为同一时间机器人只能从一处获取运动指令，否则无法预期运行后果。

Primary&Secondary 详细解析和介绍请参考 Remote Control via TCP/IP。

2.3 Realtime (30003) 接口

Realtime 通讯接口始终以 125Hz 的频率往外发送机器人的详细状态信息（位置，速度，关节电流，力矩等），同时也可以接收上位机发送的脚本指令（脚本指令的发送规则参考 Primary & Secondary（30001 & 30002）接口介绍中相应部分）并立即执行。

Realtime 通讯的详细解析请参考 Remote Control via TCP/IP。

2.4 RTDE (30004) 接口

RTDE 全称 Realtime Data Exchange，RTDE 接口需由上位机进行配置，并根据配置以 125Hz 的频率往外发送机器人的详细状态信息（位置，速度，关节电流，力矩等），机器人也只会接收配置的输入信息（IO，寄存器等）。

与 Realtime 接口相比，RTDE 接口不能接收脚本指令，只能接收配置好的输入信息；但是 RTDE 接口不会中断当前程序的执行，而且与输入输出信息都是经过配置，传输的信息没有不需要的数据，可以按需启动和停止，因此能够大大降低对网络带宽的占用。

RTDE 接口的详细资料请参考 RTDE 通讯。

2.5 Socket 通讯

Socket 通讯是指机器人通过 Socket 相关脚本指令（脚本指令下载）与上位机进行通讯的方式。详细指令请参考脚本指令文档(搜索关键字 socket)和示例文章。

2.6 XML-RPC 通讯

XML-RPC 全称 XML remote procedure call（XML-RPC wiki）。是一种通过 XML 格式传输结构化数据的协议，其优点在于传输的数据是结构化的，无需用户自己进行序列化和解析操作，例如从上位机传输 float 数据，机器人接收到后就已经是 float 类型了。目前 XML-RPC 库已支持多种语言，例如 C++和 Python 等，读者可从网上自行下载相关资料。

UR 机器人具有 XML-RPC 客户端指令，可参考使用示例和脚本手册。

2.7 Modbus-TCP 通讯

UR 机器人支持 Modbus-TCP，既可以作为客户端也可以作为服务器。UR 机器人作为客户端，只需配置好网络，然后使用相应的脚本即可（脚本指令下载）。UR 机器人作为服务器，可以提供机器人当前位置，速度等状态信息，也提供通用寄存器可用于互相通讯（例如相机将数据传输给机器人）。如何使用机器人作为 Modbus-TCP 服务器请参考机器人做 ModbusTCP 服务器。

3. UR 机器人通讯开发 Q&A

3.1 实际应用中各通讯端口如何进行选择？

答：阅读本文前段部分对于各个通讯端口的介绍，了解各个端口所提供的功能和限制，例如是否提供了应用中所需的位置、速度信息？通讯频率是否满足要求？应用中是否对带宽占用有限制？ 如果选择 XML-RPC 方式，自身的开发语言是否有现成的库？这样能大大降低开发工作量。

3.2 开发通讯软件可以使用的工具？

答：对于目前 UR 机器人相关的通讯开发，常使用的软件小工具有 Socket 测试小软件（例如 SocketTest），URSim（UR 机器人仿真软件）。特别是 URSim 软件（官网下载，百度盘下载），其提供了实际机器人一样的通讯接口，在开发中非常方便，例如在同一台电脑上通过虚拟机运行 URSim，配置好虚拟机的 IP 地址，打开 URSim 即可让上位机程序与仿真机器人进行通讯。

3.3 开发通讯驱动时注意事项？

答：UR 机器人是运行在 Debian Linux 操作系统之上，由于不同操作系统在存储数据时格式不尽相同（有的是 Big-endian，有的则是 Little-endian），而通过网络传输时都是采用 Big-endian，因此开发时注意确认自己的 PC 采用的是那种格式。

3.4 通讯端口在 UR+开发中的应用？

答：UR+中开发的运行在 UR 机器人控制器上的插件（URCap）可与上位机、机器人程序以及其他 URCap 进行通讯。例如一个力控组装的应用组成有力传感器，控制算法（通过 URCap 实现），机器人运动程序。控制算法，也就是运行在机器人控制器里的软件通过 socket 读取力传感器的数据并经过计算得到机器人需要运动的位置和速度信息，控制算法再通过 socket 或者 xml-rpc 将数据传递给机器人运动程序并由它执行；同时控制算法也在监控力传感的数值，一旦超过某些限制即可通过 29999 到 30003 端口终止机器人的程序或者运动。

3.5 有无现成的通讯端口驱动？

答：UR 全球目前还没有通讯端口驱动发布，目前 UR 中国开发了基于.Net4.0 的驱动，还处于内测阶段，也欢迎大家试用和反馈（网盘地址），目前还没有其他语言版本的驱动。