机器人控制器（DCS）自由口 TCP/IP 通讯方法

简介：

机器人与外界设备可以自由口 Scoket 通讯。但是目前 DCS 控制器只能作为客户端使用。可以实现发送接收功能，字符串解析。以下介绍常见自由口通讯几个函数的使用。本文以 DRAStudio_v1.00.07.18 版本进行测试。调试助手为 NetAssist.exe。

一、 函数介绍

1. 连接服务器：

Ret=SocketClass(host,port,spacing,delimiter,cmd,sleeptime,timeout,tcp_close)		
参数名称	**参数功能**	**说明**
Ret	创建通讯对象，通讯成功后接收对象	如连接的基恩士相机可使用：Keyence
host	服务器 IP 地址，字符串类型	如服务器 IP：“192.168.1.1”
port	服务器端口号	如服务器端口号：502
spacing	接收数据的分隔符，默认以【“ ,”】分隔	不设置需用【nil】填充参数
delimiter	发送数据的结束符，默认为【“\r\n”】结束	不设置需用【nil】填充参数
cmd	发送数据无内容时，默认发送为【“”】	不设置需用【nil】填充参数
sleeptime	发送数据的延时时间，默认为【0.1】秒	不设置需用【nil】填充参数，需依实际设置
timeout	接收数据的超时时间，默认为【10】秒	不设置需用【nil】填充参数，需依实际设置，超时后不再接受数据，执行下面程序。
tcp_close	发送数据后是否断开连接，默认为【fasle】	true 为发送后断线，fasle 为不断线
例如： Keyence= SocketClass("192.168.1.1",502,nil,nil,nil,0.05,13,false) --创建连接对象【Keyence】，连接的 IP 为【“192.168.1.1”】，端口号为【502】，使用默认\r\n 发送数据，使用默认【 ,】为接收数据的分隔符，发送间隔延时为【0.05】，接收超时时间为【13】秒，发送后不断线。		

2. 发送数据：

Ret:Send(cmd,delimiter)		
参数名称	参数功能	说明
Ret	连线成功的对象名称	如上述连接的：Keyence
cmd	设置发送的数据，若不填写则默认发送连线时的 cmd	依实际需要填写发送的数据内容，字符串
delimiter	发送数据的结束符，默认为【"\r\n"】结束	一般为默认结束符，可不填写
例如： Keyence:Send（"Trig"） --以初始化连接的 Keyence 为对象，发送内容为 "Trig(触发)"，可依实际情况填写发送内容，以【\r\n】结束符，发送数据。		

3. 接收数据：

Data=Ret:Recive(Receive_spacing,Pattern) 注意：接收数据时，默认对象发送的结束符必须为【\r\n】，否则手臂会认为数据一直未发送而超时。		
参数名称	参数功能	说明
Data	接收数据的变量，数据分隔后保存到数组中返回	Data 为数组，根据实际数据变化长度
Ret	连线成功的对象名称	如上述连接的：Keyence
Receive_spacing	设置当前接收数据的分隔符，可不填写	若不设置则以初始化 spacing 分隔
Pattern	设置接收数据的模式，默认为【*l（逐行读取）】	一般为默认，可不填写
例如： Data=Keyence: Recive（ ） --以初始化连接的 Keyence 为对象，接收服务器发送的数据，可依实际情况填写参数，读取结果赋值给 Data 数组。		

4. 分隔数据：

Data=split(str, pat)		
参数名称	参数功能	说明
Data	接收数据的变量，数据分隔后保存到数组中返回	Data 为数组，根据实际数据变化长度
str	设置分隔的字符串数据，字符串类型参数	依实际填写需要分隔的字符串
pat	分隔字符串数据的分隔符，字符串类型参数	需要设置字符串的分隔符
例如： Data= split(“X:100,Y:200,RZ:300”，“ ,”) --以“ ,”为分隔符 ,分隔字符串“X:100,Y:200,RZ:300” ,会得到数组 Data={“X:100”,“Y:200” ,“RZ:300”}。		

5. 断开连接：

Ret:Close()		
参数名称	参数功能	说明
Ret	连线成功的对象名称	如上述连接的：Keyence
例如： Keyence:Close() --断开连接对象 Keyence。		

二、 范例说明

1、 说明：这里范例以调试助手和手臂控制器进行说明，实际设备操作方法相同。调试时电脑 IPv4 设置为固定 IP，地址为 192.168.1.96，调试助手端口号改为 600。手臂以发送字符串“Trig”模拟发送触发拍照信号，然后再接收相机返回拍照结果。范例特意使用两次分隔字符串，讲解指令使用。实际使用中最好只要分隔一次，减少程序的复杂性，这需要与对应设备沟通定义发送与接收格式。

2、 接收数据的功能块

```
function ReciveData()
    while not Data do
        Data=Keyence:Receive()  --接收数据，赋值给Data
    end
    if Data~=nil and #Data==3 then --当数组不为nil且长度为3时(#Data获取数组长度)
        --此时Data={"X:100","Y:200","RZ:300"}
        --Data[1]   --以【,】分隔后的第一组数据
        --Data[2]   --以【,】分隔后的第二组数据
        --Data[3]   --以【,】分隔后的第三组数据
        --分隔X数据，得到实际的X坐标
        Data_X=split(Data[1],":")  --此时Data_X={"X","100"}
        x=tonumber(Data_X[2])   --Data_X数组的第二位是实际的X坐标数据
        WriteModbus(0x1000,"DW",x*1000) --写入寄存器要是整数
        --分隔X数据，得到实际的X坐标
        Data_Y=split(Data[2],":")  --此时Data_X={"Y","200"}
        y=tonumber(Data_Y[2])   --Data_X数组的第二位是实际的X坐标数据
        WriteModbus(0x1002,"DW",y*1000)
        --分隔X数据，得到实际的X坐标
        Data_RZ=split(Data[3],":")  --此时Data_X={"RZ","300"}
        rz=tonumber(Data_RZ[2])  --Data_X数组的第二位是实际的X坐标数据
        WriteModbus(0x1004,"DW",rz*1000)
    else
        QUIT() --实际项目中需要编写异常处理程序
    end
end
```

接收数据的功能块，接收数据和解析字符串。在接收数据时一定告知对方，我们接收数据必须以【\r\n】结尾，否则收不到数据。

a) 这里编写 while 循环一直等待数据的接收，条件 not Data 表示 Data 为 nil 时一直等待，直到 Data 有数据时跳出循环。

b) 判断 Data 数据长度是否为 3(这里长度 3 表示字符串被分隔成 3

部分），#Data 可以获取数组的长度，可以当做条件判断数据是否正确。else 条件这里简单编写为 QUIT()，实际项目中要编写异常处理程序。

c) 解析分隔后的数据，这里举例说明 split 函数使用，实际中数据格式定义尽可能分隔一次就好，减少程序复杂性。tonumber 函数可以将字符串转换为数值。用以操作坐标数据运算等。

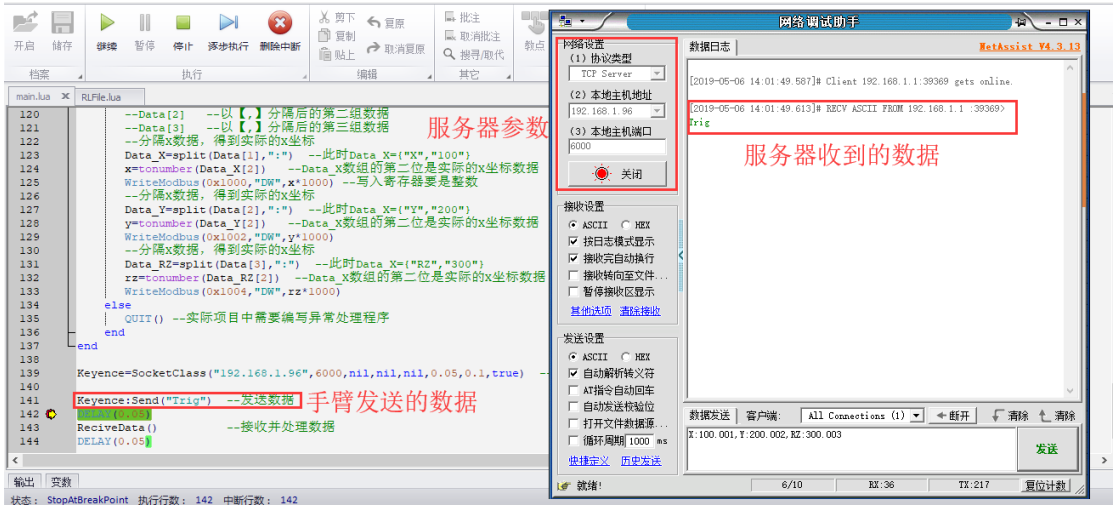
3、 连线与发送数据

```
138
139  Keyence=SocketClass("192.168.1.96",6000,nil,nil,nil,0.05,13,true)   --连接服务器
140
141  Keyence:Send("Trig")   --发送数据
142  DELAY(0.05)
143  ReciveData()           --接收并处理数据
```

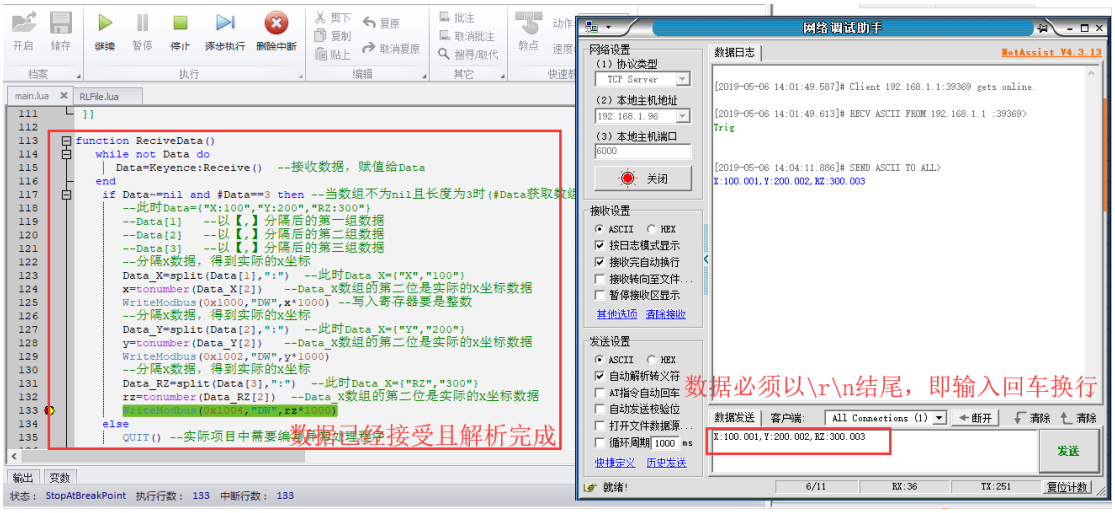
a) 根据实际情况填入参数，可在程序初始化时连接一次服务器。

b) 发送数据，默认以\r\n 结束，结束符在发送与接收时非常重要，要与对方设备定义相同，不然会出现数据收不到的情况。

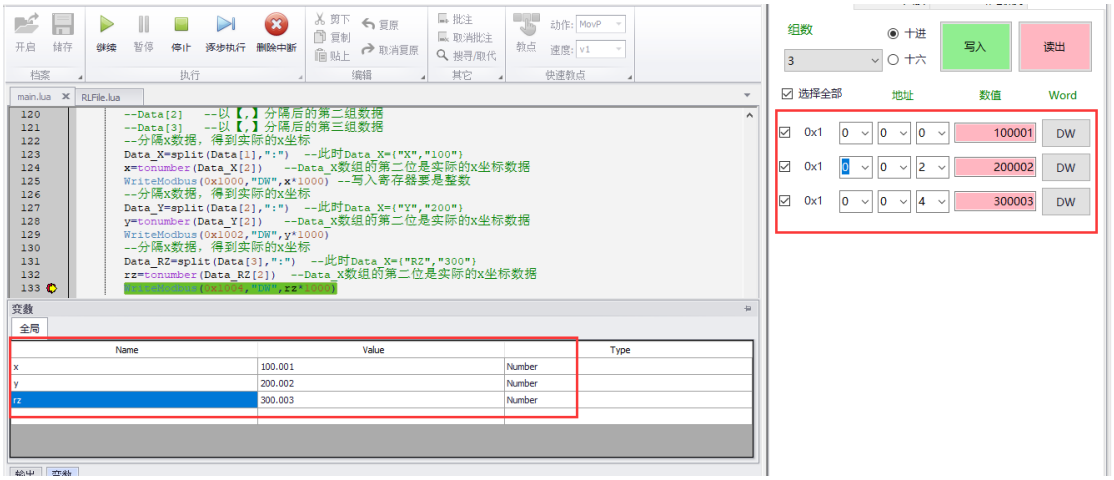
4、 实际操作截图



发送数据



数据接收与解析



变量与地址监控结果