如何使用末端的485

末端的485 主要被用来控制末端的执行器 如 电爪，吸盘，六维力传感器。目前对于末端485 的控制主要是使用脚本和使用TCP的数据透传进行控制。

脚本的控制主要被应用在夹爪等配件中。 TCP 的数据透传则被用于插件和配件之间的数据传输。

1. 脚本指令

1.0 设置末端的波特率 SetToolBaudRate(baudRate)

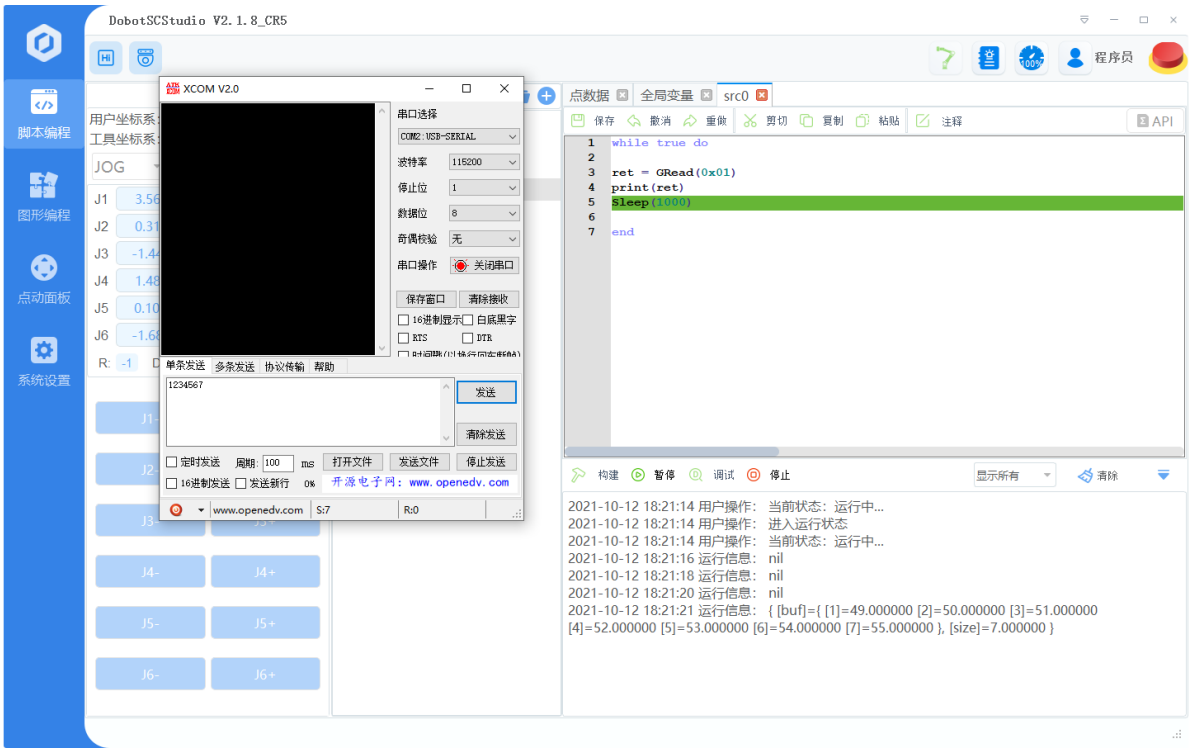
这个API 用于设置末端的波特率，例如

```
SetToolBaudRate(115200)   -- 理论上可以支持 2400 ~ 2.5M
```

1.1 GRead()

GRead() 指令主要是从末端的485 读取数据，指令使用的参考如下：

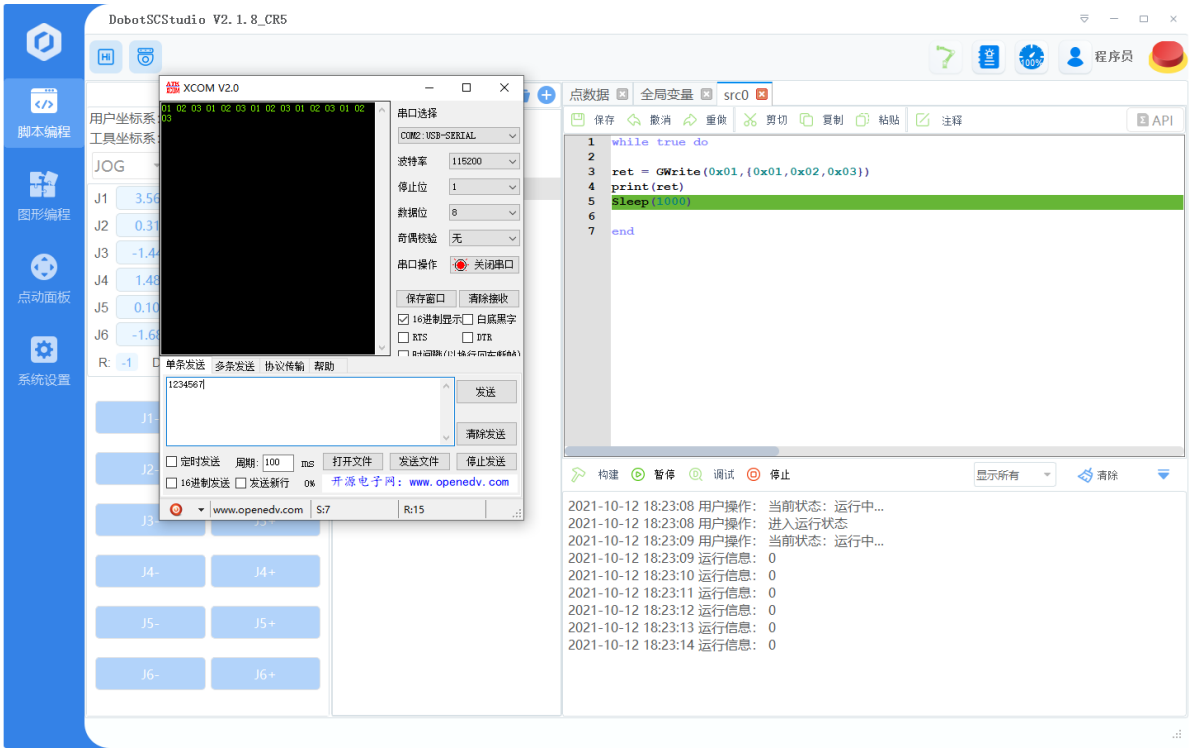
```
while true do
ret = GRead(0x01)
print(ret)
Sleep(1000)
end
```



如果接收到数据 那么会返回数据的table 以及数据的size.

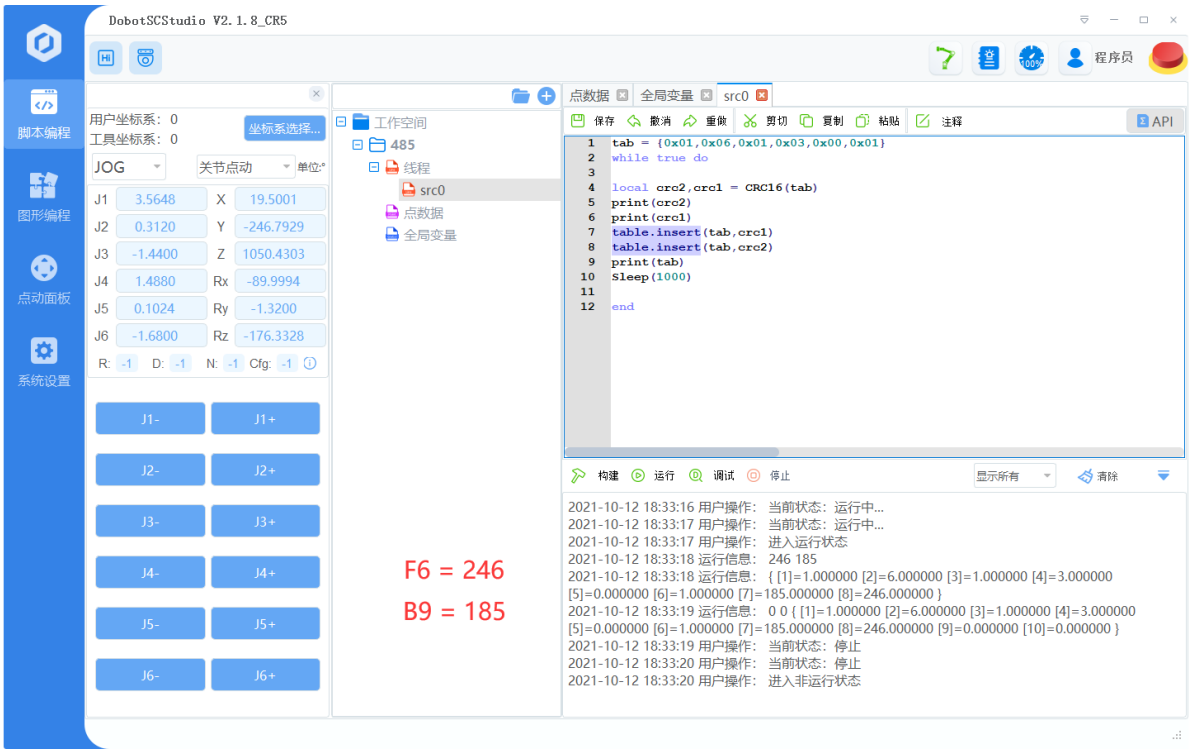
1.2 GWrite()

```
while true do
ret = GWrite(0x01,{0x01,0x02,0x03})
print(ret)
Sleep(1000)
end
```



1.3 CRC16()

这个函数是用来计算modbus RTU的CRC的




```
http://www.ip33.com/crc.html
```

2. 使用参考

2.1 modbus RTU 读写寄存器

```
function terminal_modbus_write_regs(id,addr,val_tab)
 if nil == id or nil == addr or nil == val_tab then
  print("[terminal_modbus_write_regs] : id , addr ,value tab has something
disappear!")
  return {}
 end
 send_tab =
{id,0x10,math.floor(addr/256),math.floor(addr%256),math.floor((#val_tab)/256),ma
th.floor((#val_tab)%256),math.floor(2*(#val_tab))}
 for k,v in ipairs(val_tab) do
  table.insert(send_tab,math.floor(v/256))
  table.insert(send_tab,math.floor(v%256))
 end
 crc2,crc1 = CRC16(send_tab)
 table.insert(send_tab,crc1)
 table.insert(send_tab,crc2)
 --print(send_tab)
 GWrite(0x01,send_tab)
end

function terminal_modbus_read_regs(id,addr,length)
 if nil == id or nil == addr or nil == length then
  print("[terminal_modbus_read_regs] : id , addr ,length has something
disappear!")
  return {}
 end
```

```
 send_tab =
{id,0x03,math.floor(addr/256),math.floor(addr%256),math.floor(length/256),math.f
loor(length%256)}
 crc2,crc1 = CRC16(send_tab)
 table.insert(send_tab,crc1)
 table.insert(send_tab,crc2)
 --print(send_tab)
 GWrite(0x01,send_tab)
 local value = GRead(0x01)
 --print(value)
 if nil == value then
  return {}
 else
  temp_tab = value.buf
  crc_ret_2 = temp_tab[#temp_tab]
  table.remove(temp_tab)
  crc_ret_1 = temp_tab[#temp_tab]
  table.remove(temp_tab)
  crc_calc2,crc_calc1 = CRC16(temp_tab)
  --print(string.format("%d ;%d ;%d
;%d;\n",crc_ret_1,crc_ret_2,crc_calc1,crc_calc2))
  tab_ret = {}
  --print(temp_tab)
  if crc_calc2 == crc_ret_2 and crc_calc1 == crc_ret_1 then
   for i=4,#temp_tab,2 do
    table.insert(tab_ret,temp_tab[i]*256 + temp_tab[i+1])
   end
   return tab_ret
  end
 end
 return {}
end
while true do
    terminal_modbus_write_regs(1,1000,{123,222,4565})
    -- 01 10 03 E8 00 03 06 00 7B 00 DE 11 D5 25 87
    Sleep(1000)
end
```


XCOM V2.0
01 10 03 E8 00 03 06 00 7B 00 DE 11 D5 25
87
串口选择
COM2:USB-SERIAL
波特率
115200
停止位
1
数据位
8
奇偶校验
无
串口操作
关闭串口
保存窗口
清除接收
16进制显示
白底黑字
RTS
DTR
单条发送
多条发送
协议传输
帮助
1234567
发送
清除发送
定时发送
周期:
100
ms
打开文件
发送文件
停止发送
16进制发送
发送新行
0%
开源电子网：www.openedv.com
www.openedv.com
S:0
R:15