目 录

ZModbusSdk 配置函数库使用手册

V 1.07

1 函数库说明

简介：

ZModbusSdk 函数库为标准的 MODBUS(主机)协议提供函数接口。开发人员可以使用此函数库方便开发出遵循 MODBUS 协议的程序。如果需要开发 MODBUS 从机程序，开发人员需要使用其他的函数库或方法。

函数库在连接时使用对不同的链路层提供不同的函数接口如：以太网的为：ZMB_TCPConnectMDBServer,串口的为：ZMB_SerConnectMDBServer。其他的函数接口都相同，不区分连接链路。

（本文档中使用的术语服务器对应 MODBUS 协议里的从机模块，客户机对应 MODBUS 协议里的主机模块）

1.1 提供的函数接口

ZModbusSdk 提供了连接到从机（服务器）和读写线圈与寄存器的函数接口。

1.1.1 以太网链路连接

- ZMB_TCPConnectMDBServer

描述

调用此函数连接到从机模块（或服务器）。

```
HANDLE ZMB_TCPConnectMDBServer(   char    * szIp,
                                  int     nDstPort,
                                  int     nConnTimeout);
```

参数

szIp

指定要进行连接的从机（服务器）的 IP 地址。

nDstPort

指定从机的端口号。

nConnTimeout

指定连接超时。

返回值

返回“非 NULL”表示成功，否则为错误。

1.1.2 串口链路连接

- ZMB_SerConnectMDBServer（串口主机使用）

描述

调用此函数连接到从机模块（或服务器）。使用者应该注意对于串口的操作每个串口在同一时间只能打开一次，因此在操作串口时应该控制号程序使它每次只打开一次。

```
HANDLE ZMB_SerConnectMDBServer (  int     iRtuAscii,
                                  char    *szCom,
                                  int     iBautRate,
```

```
                                    int         iByteSize,
                                    int         iParity,
                                    int         iStopBits,
                                    int         iDtrCtl,
                                    int         iRtsCtl,
                                    int         iCtsCtl,
                                    int         iDsrCtl,
                                    int         iResponse);
```

参数

iRtuAscii

的模块工作方式：RTU:0, ASCII:1

szCom

需要打开的计算机串口。”COM1”,”COM2”,……

iBautRate

模块工作的波特率。9600,19200,……

iByteSize

模块工作的数据位。数据位数（字节表示 4-8 位）

iParity

模块工作的校验位，奇偶校验 0-4：表示：不校验，奇校验，偶校验，标号，空格

iStopBits

模块工作的停止位，0(ONESTOPBIT)表示 1 个停止位，1(ONE5STOPBITS)表示 1.5 个停止位，2(TWOSTOPBITS)表示 2 个停止位

iDtrCtl

模块是否需要 DTR 控制，0 不需要，1 需要

iRtsCtl

模块是否需要 RTS 控制，0 不需要，1 需要

iCtsCtl

模块是否需要 CTS 控制，0 不需要，1 需要

iDsrCtl

模块是否需要 Dsr 控制，0 不需要，1 需要

iResponse

模块的超时响应，默认 1000 ms

返回值

返回“非 NULL”表示成功，否则为失败。

1.1.3 以太网与串口链路断开连接

- ZMB_DisConnectMDBServer

描述

调用此函数断开和从机的连接。

```
void  ZMB_DisConnectMDBServer(HANDLE      hHand);
```

参数

hHand

此 handle 是 ConnectMDBServer 操作的返回值。

返回值

无。

1.1.4 以太网与串口链路的数据采集（读写）

- ZMB_ReadCoil

描述

调用此函数按参数中的 ID 号和地址读取线圈状态。

```
long ZMB_ReadCoil(      HANDLE      hand,
                        BYTE        serID,
                        int         nStart,
                        int         nCount,
                        BYTE        * bRet/*arr of coils*/,
                        WORD        TranID);
```

参数

hand

此 handle 是 ConnectMDBServer 操作的返回值。

serID

设备 ID 号。

nStart

线圈的起始地址。

nCount

线圈的个数。

bRet

返回的线圈状态，已分配号的数组地址。

TranID

在 TCP 的 MODBUS 协议里需要用到任务 ID 号，如果为 0 使用系统自生成的，串口 MODBUS 此参数无效。

返回值

返回 0 表示成功，否则为错误码。

- ZMB_ReadInput

描述

调用此函数按参数中的 ID 号和地址读取线圈状态（离散输入量数据）。

```
long ZMB_ReadInput (    HANDLE      hand,
                        BYTE        serID,
                        int         nStart,
                        int         nCount,
                        BYTE        * bRet /*arr of coils*/,
                        WORD        TranID);
```

参数

hand

此 handle 是 ConnectMDBServer 操作的返回值。

serID

设备 ID 号。

nStart

线圈的起始地址。

nCount

线圈的个数。

bRet

返回的线圈状态，已分配号的数组地址，此数组的大小应该和 nCount 指定的大小相同。

TranID

在 TCP 的 MODBUS 协议里需要用到任务 ID 号，如果为 0 使用系统自生成的，串口 MODBUS 此参数无效。

返回值

返回 0 表示成功，否则为错误码。

- ZMB_ReadHoldReg

描述

调用此函数按参数中的 ID 号和地址读取保持寄存器的数据。

```
long ZMB_ReadHoldReg (      HANDLE      hand,
                            BYTE        serID,
                            int         nStart,
                            int         nCount,
                            WORD        * bRet/*arr of reg*/,
                            WORD        TranID);
```

参数

hand

此 handle 是 ConnectMDBServer 操作的返回值。

serID

设备 ID 号。

nStart

寄存器的起始地址。

nCount

寄存器的个数。

bRet

返回的寄存器数据，已分配号的数组地址，此数组的大小应该和 nCount 一样。

TranID

在 TCP 的 MODBUS 协议里需要用到任务 ID 号，如果为 0 使用系统自生成的，串口 MODBUS 此参数无效。

返回值

返回 0 表示成功，否则为错误码。

- ZMB_ReadInputReg

描述

调用此函数按参数中的 ID 号和地址读取输入寄存器的数据。

```
long ZMB_ReadInputReg (     HANDLE      hand,
                            BYTE        serID,
                            int         nStart,
                            int         nCount,
```

```
                                WORD        * Ret/*arr of reg*/,
                                WORD        TranID);
```

参数

hand

此 handle 是 ConnectMDBServer 操作的返回值。

serID

设备 ID 号。

nStart

寄存器的起始地址。

nCount

寄存器的个数。

bRet

返回的输入寄存器数据，已分配号的数组地址，此数组的大小应该和 nCount 一样大。

TranID

在 TCP 的 MODBUS 协议里需要用到任务 ID 号，如果为 0 使用系统自生成的，串口 MODBUS 此参数无效。

返回值

返回 0 表示成功，否则为错误码。

- ZMB_WriteCoilSingle

描述

调用此函数按参数中的 ID 号和地址修改单个线圈状态。

```
long ZMB_WriteCoilSingle (      HANDLE      hand,
                                BYTE        serID,
                                int         nAddress,
                                WORD        wWriteData,
                                WORD        TranID);
```

参数

hand

此 handle 是 ConnectMDBServer 操作的返回值。

serID

设备 ID 号。

nAddress

线圈的起始地址。

wWriteData

写入的线圈数据：“0”，“1”。

TranID

在 TCP 的 MODBUS 协议里需要用到任务 ID 号，如果为 0 使用系统自生成的，串口 MODBUS 此参数无效。

返回值

返回 0 表示成功，否则为错误码。

- ZMB_WriteCoilMulitiple

描述

调用此函数按参数中的 ID 号和地址修改 nCount 指定的多个线圈状态。

```
long ZMB_WriteCoilMulitiple (   HANDLE      hand,
                                BYTE        serID,
                                int         nAddress,
                                int         nCount,
                                BYTE        * bWriteDataArr,
                                WORD        TranID);
```

参数

hand

此 handle 是 ConnectMDBServer 操作的返回值。

serID

设备 ID 号。

nAddress

线圈的起始地址。

nCount

写入的线圈的个数。

bWriteDataArr

写入的线圈数据的数组，数组的每个结构保存一个线圈状态，此数组大小应该和 nCount 指定的大小相同。

TranID

在 TCP 的 MODBUS 协议里需要用到任务 ID 号，如果为 0 使用系统自生成的，串口 MODBUS 此参数无效。

返回值

返回 0 表示成功，否则为错误码。

- ZMB_WriteRegSingle

描述

调用此函数按参数中的 ID 号和地址修改单个保持寄存器数据。

```
long ZMB_WriteRegSingle (   HANDLE      hand,
                            BYTE        serID,
                            int         nAddress,
                            WORD        wWriteData,
                            WORD        TranID);
```

参数

hand

此 handle 是 ConnectMDBServer 操作的返回值。

serID

设备 ID 号。

nAddress

保持寄存器的地址。

wWriteData

写入的寄存器数据。

TranID

在 TCP 的 MODBUS 协议里需要用到任务 ID 号，如果为 0 使用系统自生成的，串口

MODBUS 此参数无效。

返回值

返回 0 表示成功，否则为错误码。

- ZMB_WriteRegMulitiple

描述

调用此函数按参数中的 ID 号和地址修改 nCount 指定多个保持寄存器。

```
long ZMB_WriteRegMulitiple (    HANDLE      hand,
                                BYTE        serID,
                                int         nAddress,
                                int         nCount,
                                WORD        * writeDataArr,
                                WORD        TranID);
```

参数

hand

此 handle 是 ConnectMDBServer 操作的返回值。

serID

设备 ID 号。

nAddress

保持寄存器的起始地址。

nCount

写入的寄存器的个数。

bWriteDataArr

写入的寄存器数据的数组，数组的每个结构保存一个寄存器的值，此数组个数应该和 nCount 指定的个数相同，大小应该等于 nCount * 2 个字节。

TranID

在 TCP 的 MODBUS 协议里需要用到任务 ID 号，如果为 0 使用系统自生成的，串口 MODBUS 此参数无效。

返回值

返回 0 表示成功，否则为错误码。

- ZMB_WriteCmdBuf

描述

调用此函数写命令串，根据 nAddCheck 参数决定是否添加校验。其中 TCP Modbus 协议不会添加必要的头部，而串口 Modbus 则可以根据 nAddCheck 指定是否添加校验，校验的格式（CRC 或 LRC）由打开参数 iRtuAscii 指定，这里不需要再指定。

```
long ZMB_WriteCmdBuf (  HANDLE      hand,
                        BYTE        *pBuf,
                        int         iLen,
                        int         nAddCheck       );
```

参数

hand

此 handle 是 ConnectMDBServer 操作的返回值。

pBuf

写数据的缓冲区。

iLen

缓冲区长度。

nAddCheck

是否添加校验，只在串口中有效，如果打开时为 RUT，且使用了添加校验，则添加 CRC 校验。如果打开时为 ASCII，且使用了添加校验，则添加 LRC 校验。

TranID

在 TCP 的 MODBUS 协议里需要用到任务 ID 号，如果为 0 使用系统自生成的，串口 MODBUS 此参数无效。

返回值

返回写成功的长度。

- ZMB_ReadRecBuf

描述

调用此函数读取设备的响应数据。

```
long ZMB_ReadRecBuf (   HANDLE      hand,
                        BYTE        *pBuf,
                        int         iBufLen,
                        int         *iRecedLen      );
```

参数

hand

此 handle 是 ConnectMDBServer 操作的返回值。

pBuf

读数据的缓冲区。

iBufLens

缓冲区的大小。

iRecedLen

接收到的数据的长度，（输出参数）。

TranID

在 TCP 的 MODBUS 协议里需要用到任务 ID 号，如果为 0 使用系统自生成的，串口 MODBUS 此参数无效。

返回值

返回读取到的长度。

2 错误代码

ZModbusSdk 使用了标准的 MODBUS 错误代码和自定义错误代码来描述 SDK 在操作过程中会出现的错误。

在使用的过程中应该注意错误的定义，有些使用 0 作为成功的标志，而有些使用 0 作为错误的标志。

标准 MODBUS 的读写操作都以 ERROR_SUCCESS（0）表示正确，Connect 却以 NULL（0）表示错误，特殊命令的读写返回的读写的数据长度。

错误码	错误含义
自定义错误码	
0	成功
100	普通错误，未定义
101	不是 MODBUS 协议
102	TCP 连接出错（可能主机不存在或本地网络有问题）
103	TCP 发送命令出错（发送不成功，可能为网络问题）
104	TCP 接收响应出错（可能指示不正确或网络有问题）
105	TCP 发送命令超时未完成
106	TCP 接收响应超时未完成
107	串口写出错
108	串口读出错
109	返回的功能码不是发出请求的功能码
200	发送不成功，如果出现这个错误，需要重新连接
201	读超时，可能是下位机响应不过，也可能是其他原因，需要具体分析
202	接收到非标准的 MODBUS 协议数据
标准 MODBUS 错误	
1	未定义的功能码
2	地址出错
3	数据出错
0x81	读线圈出错
0x82	读离散输入量出错
0x83	读保持寄存器出错
0x84	读输入寄存器出错
0x85	写单个线圈出错
0x86	写单个寄存器出错
0x8F	写多个线圈出错
0x90	写多个寄存器出错

3 示例代码

1.2 以太网链路读取线圈状态

```
HANDLE hConn = ZMB_TCPConnectMDBServer("192.168.21.28", 502 );
BYTE arr[10];
ZMB_ReadCoil(hConn, 1, 1, 1, arr, NULL, NULL, 0);
```

1.3 串口链路读取线圈状态

```
HANDLE hMaster = ZMB_SerConnectMDBServer(0,"COM1",9600, 8, 0, 1, 0, 0, 0, 0, 1000 );
BYTE arr[10];
ZMB_ReadCoil(hMaster, 1, 1, 1, arr, NULL, NULL, 0);
```

1.4 写多个线圈

假设 m_hEthMaster 和 m_hSerMaster 时 SetConnectMdbServer 的正确返回值或时 NULL，同时 m_iEthID,m_iSerID,为正确的从机 ID 号，addr 为起始地址（或起始偏移），count 为写入的个数，arr 为写入线圈的数组，这些参数都被赋予合理的值。

```
if (m_hEthMaster)
{
    if (ZMB_WriteCoilMulitiple(m_hEthMaster, m_iEthID, addr, count, arr, 0) == ERROR_SUCCESS)
    {
        MessageBox("写以太网线圈成功");
    }
}
if (m_hSerMaster)
{
    if (ZMB_WriteCoilMulitiple(m_hSerMaster, m_iSerID, addr, count, arr,  0) == ERROR_SUCCESS)
    {
        MessageBox("写串口线圈成功");
    }
}
```