

设置动态库接口说明

```
#define WIAPI extern "C" __declspec(dllimport)
```

```
typedef struct _DEVICEINFO
```

```
{
```

```
    char strMAC[20]; //转换器 MAC 地址,格式为 00.09.F6.01.02.03
```

```
    char strIP[20];    //转换器 IP 地址, 格式为 10.1.1.1
```

```
    DWORD dwModel;    //转换器型号
```

```
    DWORD dwVersion;  //版本号
```

```
    char strName[20]; //名字
```

```
    BYTE pbUnused[100];//未使用
```

```
}DEVICEINFO;
```

```
typedef struct _C2000NETSETTING
```

```
{
```

```
    int nNetIndex; //索引号, 从 0 开始编号
```

```
    int bUseStaticIP; //0: 自动 IP; 1: 指定 IP
```

```
    char ipAddr[20];    //C3000 的 IP 地址
```

```
    char ipNetMask[20]; //网络所在掩码
```

```
    char ipGateway[20]; //网关的 IP 地址
```

```
    char ipDnsServer[20]; //DNS 服务器的 IP 地址
```

```
    BYTE pbUnused[200];//未使用
```

```
}C2000NETSETTING;
```

```
typedef struct _C2000SOCKETSETTING
```

```
{
```

```
    int nComIndex;    //串口号, 该套接口参数是属于哪个串口的,
```

```
                //如果不属于任何串口, 该值为负数, 具体值另行解释
```

```
    int nSocketIndex; //该套接口在某个串口中编号, 从 0 开始编号
```

```
    int nWorkMode; //0: TCP 客户; 1: TCP 服务器; 2: UDP1 或自动;
```

```
                //3: UDP2; 4: 自动, 根据具体的产品而定
```

```
    int nLocalPort; //本地端口
```

```
    char szPeerName[64]; //对方主机的域名或 IP
```

```
    int nPeerPort; //对方端口
```

```
    int bUseSocket; //是否使用 Socket。0: 否, 1: 是
```

```
    BYTE pbUnused[200];//未使用
```

```
}C2000SOCKETSETTING;
```

```
typedef struct _C2000COMSETTING
```

```
{
```

```
    int nComIndex;    //串口号, 该串口设置参数是属于哪个串口的, 从 0 开始编号
```

```
    int nBaudrate; //波特率
```

```

    int nDatabit; //数据位数
    int nParity; //校验方式
    int nStopbit; //停止位数
    int nFlowCtrl; //流控方式
    u_int nIntervalTimeout; //间隔超时时间（最少发送时间）
    u_int nMaxFrameLength; //最大帧长度（最小发送字节）
    int nInceptBytesLength; //包头长度（暂不支持）
    BYTE pbInceptBytes[4]; //包头字符（暂不支持）
    int nTermBytesLength; //包尾长度（暂不支持）
    BYTE pbTermBytes[4]; //包尾字符（暂不支持）
    int nWorkMode; //0: 不变 1: 232 2: 485 3: 422
    BYTE pbUnused[200]; //未使用
}C2000COMSETTING;

typedef struct _C2000NETSTATUS
{
    int nNetIndex; //索引号，从 0 开始编号
    char ipAddr[20]; //C3000 的 IP 地址
    char ipNetMask[20]; //网络所在掩码
    char ipGateway[20]; //网关的 IP 地址
    char ipDnsServer[20]; //DNS 服务器的 IP 地址
    BYTE pbUnused[200]; //未使用
}C2000NETSTATUS;

typedef struct _C2000SOCKETSTATUS
{
    int nComIndex; //串口号，该套接口参数是属于哪个串口的，
    //如果不属于任何串口，该值为负数，具体值另行解释
    int nSocketIndex; //该套接口在某个串口中编号，从 0 开始编号
    int nWorkMode; //0: TCP 客户；1: TCP 服务器；2: UDP 或 自动
    int nLocalPort; //本地端口
    char szPeerName[64]; //设置的对方 IP 地址或域名
    int nPeerPort; //对方端口
    int bUseSocket; //是否使用 Socket。0: 否，1: 是
    int bConnected; //连接情况,0:未连接 1: 连接
    char ipCurPeerAddr[20]; //当前对方的 IP 地址
    int nCurPeerPort; //当前对方的端口号
    BYTE pbUnused[200]; //未使用
}C2000SOCKETSTATUS;

typedef struct _C2000COMSTATUS
{
    int nComIndex; //串口号，该串口状态是属于哪个串口的
    int nBaudrate; //波特率

```

```

    int nDatabit;    //数据位数
    int nParity;     //校验方式
    int nStopbit;    //停止位数
    int nFlowCtrlMode; //流控方式
    int nFlowCtrlStatus; //流控状态
    int nIntervalTimeout; //最少发送时间
    int nMaxFrameLength; //最小发送字节
    int nWorkMode; //0: 不明 1: 232 2: 485 3: 422
    BYTE pbUnused[200]; //未使用
}C2000COMSTATUS;

typedef struct _C2000MODEL
{
    DWORD dwModel; //型号
    DWORD dwNetCount; //网络参数的数目
    DWORD dwComCount; //串口数
    DWORD dwSocketCount; //套接口数
    char strModel[60]; //型号字符串
    BYTE pbUnused[200]; //未使用
}C2000MODEL;

//建立通讯用的套接口
//iProtocol: 采用的协议
//    IPPROTO_UDP          = 17,
//    IPPROTO_TCP          = 6,
//ulPeerAddr: 对端 IP 地址（低位在前）
//ulPeerPort: 对端端口（低位在前）
//dwTimeout: 超时时间（毫秒）
WIAPI BOOL __stdcall CreateSocket(int iProtocol, u_long ulPeerAddr, u_short ulPeerPort,
    DWORD dwTimeout);

//获得所有可用的 C2000 型号编号列表
//pModelID: 保存型号编码。调用者分配空间。
//nCount: 指定 pModelID 的空间大小。
//返回值: 实际支持型号数目。
WIAPI int __stdcall GetAllModelID(DWORD *pModelID, int nCount);

//获得指定型号字符串
//dwModel: 型号编码。
//lpBuffer: 存放型号描述字符串。调用者分配空间。
//nLen: lpBuffer 的大小。
//返回值: 成功 :非 0. 失败: 0
WIAPI BOOL __stdcall GetModelString(DWORD dwModel, LPSTR lpBuffer, int nLen);

```

////////////////////////////////////

//功能：搜索设备

//参数：

//pDI：用来保存搜索到的设备信息

//nCount：pDI 中可保存的设备信息的数量

//dwTimeout：执行这个命令的最大时间，超过这个时间，函数就返回。单位：毫秒。

//返回值：返回找到的设备数量。

WIAPI int __stdcall SearchDevice(DEVICEINFO *pDI, int nCount, DWORD dwTimeout);

////////////////////////////////////

//功能：获得指定设备的信息

//参数：

//lpszMac：要获得其信息的设备的 MAC 地址

//pDI：用来保存返回的设备信息

//dwTimeout：执行这个命令的最大时间，超过这个时间，函数就返回。单位：毫秒。

//返回值：成功返回非零，失败返回零。

WIAPI BOOL __stdcall GetDeviceInfo(const char *lpszMac, DEVICEINFO *pDI, DWORD dwTimeout);

////////////////////////////////////

//功能：获得 C2000 型号的信息

//参数：

//dwModel：型号

//pC2000Model：用来保存返回的型号信息。

//返回值：成功返回非零，失败返回零。

WIAPI BOOL __stdcall GetC2000ModelInfo(DWORD dwModel, C2000MODEL *pC2000Model);

////////////////////////////////////

//功能：获得指定 C2000 的设置信息

//参数：

//lpszMac：要获得其设置信息的 C2000 的 MAC 地址

//pNS：用来保存返回的网络设置信息。调用者分配空间。

// nNetCount：输入时为 pNS 中能保存的设置信息个数。输出时为返回的实际数目。

//pSS：用来保存返回的套接字设置信息。调用者分配空间。

// nSocketCount：输入时为 pSS 中能保存的设置信息个数。输出时为返回的实际数目。

//pCS：用来保存返回的串口设置信息

//nComCount：输入时为 pCS 中能保存的设置信息个数。输出时为实返回的实际数目。

//dwTimeout：执行这个命令的最大时间，超过这个时间，函数就返回。单位：毫秒。

//返回值：成功返回非零，失败返回零。

WIAPI BOOL __stdcall GetC2000Setting(const char *lpszMac, C2000NETSETTING *pNS, int *nNetCount, C2000SOCKETSETTING *pSS, int *nSocketCount, C2000COMSETTING *pCS, int *nComCount, DWORD dwTimeout);

////////////////////////////////////

//功能：设置指定 C2000

//参数：

//lpszMac：要设置的 C2000 的 MAC 地址

//pNS：要设置的网络参数

//nNetCount：网络参数数目。

//pSS：要设置的套接字参数

//nSocketCount：套接字参数数目。

//pCS：要设置的串口参数

//nComCount：串口参数数目。

//dwTimeout：执行这个命令的最大时间，超过这个时间，函数就返回。单位：毫秒。

//返回值：成功返回非零，失败返回零。

WIAPI BOOL __stdcall SetC2000Setting(const char *lpszMac, C2000NETSETTING *pNS, int nNetCount, C2000SOCKETSETTING *pSS, int nSocketCount, C2000COMSETTING *pCS, int nComCount, DWORD dwTimeout);

////////////////////////////////////

//功能：应用指定 C2000 的设置

//参数：

//lpszMac：要应用设置的 C2000 的 MAC 地址

//dwTimeout：执行这个命令的最大时间，超过这个时间，函数就返回。单位：毫秒。

//返回值：成功返回非零，失败返回零。

WIAPI BOOL __stdcall ApplySetting(const char *lpszMac, DWORD dwTimeout);

////////////////////////////////////

//功能：复位的指定 C2000

//参数：

//lpszMac：要复位的 C2000 的 MAC 地址

//dwTimeout：执行这个命令的最大时间，超过这个时间，函数就返回。单位：毫秒。

//返回值：成功返回非零，失败返回零。

WIAPI BOOL __stdcall Reset(const char *lpszMac, DWORD dwTimeout);

////////////////////////////////////

//功能：获得指定 C2000 的状态信息

//参数：

//lpszMac：要获得其状态信息的 C2000 的 MAC 地址

//pNS：用来保存返回的网络状态信息.调用者分配空间。

//nNetCount：输入为 pNS 中可存放信息数目。输出时为返回的实际数目。

//pSS：用来保存返回的套接字状态信息。调用者分配空间。

//nSocketCount：输入为 pSS 中可存放信息数目。输出时为返回的实际数目。

//pCS：用来保存返回的串口状态信息。调用者分配空间。

//nComCount：输入为 pCS 中可存放信息数目。输出时为返回的实际数目。

//dwTimeout：执行这个命令的最大时间，超过这个时间，函数就返回。单位：毫秒。

//返回值：成功返回非零，失败返回零。

```
WIAPI BOOL __stdcall GetC2000Status(const char *lpszMac, C2000NETSTATUS *pNS, int
*nNetCount, C2000SOCKETSTATUS *pSS, int *nSocketCount, C2000COMSTATUS *pCS, int
*nComCount, DWORD dwTimeout);
```

```
////////////////////////////////////
```

```
//功能：获得 C2000 的网页登陆密码
```

```
//lpszMac：要获得其网页登陆密码的 C2000 的 MAC 地址
```

```
//strWebPwd：用来保存获得的密码的缓冲区
```

```
//nLen：strWebPwd 缓冲区的长度
```

```
//dwTimeout：执行这个命令的最大时间，超过这个时间，函数就返回。单位：毫秒。
```

```
//返回值：成功返回非零，失败返回零。
```

```
WIAPI BOOL __stdcall GetWebPassword(const char *lpszMac, char *strWebPwd, size_t nLen,
DWORD dwTimeout);
```

```
////////////////////////////////////
```

```
//功能：设置 C2000 的网页登陆密码
```

```
//lpszMac：要获得其网页登陆密码的 C2000 的 MAC 地址
```

```
//strWebPwd：要设置的密码，最长不超过 16 个字节
```

```
//dwTimeout：执行这个命令的最大时间，超过这个时间，函数就返回。单位：毫秒。
```

```
//返回值：成功返回非零，失败返回零。
```

```
WIAPI BOOL __stdcall SetWebPassword(const char *lpszMac, const char *strOldPwd, const
char *strNewPwd, DWORD dwTimeout);
```

```
////////////////////////////////////
```

```
//功能：获得 C2000 的名字
```

```
//lpszMac：要获得其名字的 C2000 的 MAC 地址
```

```
//strName：用来保存获得的名字的缓冲区
```

```
//nLen：strName 缓冲区的长度
```

```
//dwTimeout：执行这个命令的最大时间，超过这个时间，函数就返回。单位：毫秒。
```

```
//返回值：成功返回非零，失败返回零。
```

```
WIAPI BOOL __stdcall GetC2000Name(const char *lpszMac, char *strName, size_t nLen,
DWORD dwTimeout);
```

```
////////////////////////////////////
```

```
//功能：设置 C2000 的名字
```

```
//lpszMac：要获得其名字的 C2000 的 MAC 地址
```

```
//strName：要设置的名字，最长不超过 20 个字节
```

```
//dwTimeout：执行这个命令的最大时间，超过这个时间，函数就返回。单位：毫秒。
```

```
//返回值：成功返回非零，失败返回零。
```

```
WIAPI BOOL __stdcall SetC2000Name(const char *lpszMac, const char *strName, DWORD
dwTimeout);
```

使用流程：

