1. SDK 组成

sc200drv.dll　接口动态库

sc200drv.lib　**sc200drv.dll** 的 Visual C++（以下简称 VC）用静态输入库

sc200drv.h　库函数的 VC 头文件，包括宏定义、结构体定义、函数声明，对 **sc200drv**.dll 采用隐示调用的方式，用户使用 VC 编程时可以直接使用。

2. 如何使用 SDK

用户进行软件开发时所有需要调用的函数都被封装在 **sc200drv.dll** 中，用户只需要调用这个动态库即可。

VC 调用动态库的方法主要有两种：

一、动态库的隐示调用：

在 VC 工程中直接链接输入库 **sc200drv.lib**，然后即可像调用其它源文件中的函数一样调用 API 函数接口了。

二、动态库的显式调用：

显式调用动态库步骤：

1、 创建一个函数指针，其指针数据类型要与调用的 DLL 引出函数相吻合。
2、 通过 Win32 API 函数 LoadLibrary（）显式的调用 DLL，此函数返回 DLL 的实例句柄。
3、 通过 Win32 API 函数 GetProcAddress（）获取要调用的 DLL 的函数地址,把结果赋给自定义函数的指针类型。
4、 使用函数指针来调用 DLL 函数。
5、 最后调用完成后，通过 Win32 API 函数 FreeLibrary()释放 DLL 函数。

VC 调用动态库的方法请参考 VC 示例（位于二次开发\VCDemo 目录下）。C#调用动态库的方法请参考 C#示例（位于二次开发\C#Demo 目录下）。Labview 调用动态库的方法请参考 Labview 示例（位于二次开发\LabviewDemo 目录下）。

3. 技术支持

用户在使用此 SDK 进行软件开发时，遇到任何问题或者您有更好的建议请随时通过下面的邮箱与我们联系，我们将竭诚为您服务。

技术支持邮箱：support@symc-tec.com

1. SDK API 函数快速索引列表

功能	函数名称	函数描述
枚举控制器	SC200_Dev_List	枚举控制器及序号
基本功能	SC200_Open	打开控制器
	SC200_Close	关闭控制器
开闭环设置	SC200_SetServo	设置开闭环
读取位置	SC200_GetPos	读取当前位置
设置位置	SC200_SetPos	设置目标位置

2. SDK API 函数详细介绍

SC200_Dev_List(int *devsn)

描述 ：枚举控制器的个数及保存对应的序号
参数 ：devsn[4]该数组保存每个控制器的序号，打开设备时传递对应的序号
返回值：控制器的个数。
说明 ：进行相关操作前必须使用此函数枚举控制器个数及序号；

VC 示例代码：

```
int devsn[4]={-1, -1, -1, -1};
int devnum=SC200_Dev_List(devsn);
```

c# 示例代码

```
using System.Runtime.InteropServices;
[DllImport("sc200drv.dll", EntryPoint =
"SC200_Dev_List",CallingConvention=CallingConvention.Cdecl, CharSet = CharSet.Ansi)]
private static extern int SC200_Dev_List();
int[] devsn = new int[4] { -1, -1, -1, -1};
GCHandle hObject = GCHandle.Alloc(devsn, GCHandleType.Pinned);
IntPtr ptr_devsn = hObject.AddrOfPinnedObject();
int num = SC200_Dev_List(ptr_devsn);
```

SC200_Open(int index)

描述 ：打开控制器
参数 ：index 控制器对应的序号
返回值：1 打开成功，0 打开失败。

说明 ：进行相关操作前必须使用此函数打开控制器；

VC 示例代码：

```
int sn = devsn[0];
if(SC200_Open(sn)==1)
{
    //打开成功
}
```

c# 示例代码

```
using System.Runtime.InteropServices;
[DllImport("sc200drv.dll", EntryPoint =
"SC200_Open",CallingConvention=CallingConvention.Cdecl, CharSet = CharSet.Ansi)]
private static extern int SC200_Open(int index);
int sn = devsn[0];
if(SC200_Open(sn)==1)
{
    //打开成功
}
```

SC200_Close(int index)

描述 ：关闭控制器
参数 ：index 控制器对应的序号
返回值：无。
说明 ：操作结束后使用此函数关闭控制器；

VC 示例代码：

```
int sn = devsn[0];
SC200_Close(sn);
```

c# 示例代码

```
[DllImport("sc200drv.dll", EntryPoint =
"SC200_Close",CallingConvention=CallingConvention.Cdecl, CharSet = CharSet.Ansi)]
private static extern void SC200_Close(int index);
int sn = devsn[0];
SC200_Close(sn);
```

SC200_SetServo(int index,int channel, int servo)

描述 ：设置开闭环
参数 ：index 控制器序号；channel 通道号 1~3；servo 开环为 0，闭环 1
返回值：1 设置成功，0 设置失败。

VC 示例代码：

```
int sn = devsn[0];
if(SC200_SetServo(sn,1, 0)==1)//通道 1 设置开环
{
    //设置成功
}
```

c# 示例代码

```
[DllImport("sc200drv.dll", EntryPoint = "SC200_SetServo", CallingConvention =
CallingConvention.Cdecl, CharSet = CharSet.Ansi)]
private static extern int SC200_SetServo(int index,int channel, int servo);
int sn = devsn[0];
if(SC200_SetServo(sn,1, 0)==1)//通道 1 设置开环
{
    //设置成功
}
```

SC200_SetPos(int index,int channel, float pos)

描述 ：设置目标值
参数 ：index 控制器序号；channel 通道号 1~3；pos 为设定的目标值
返回值：1 设置成功，0 设置失败。

VC 示例代码：

```
int sn = devsn[0];
if(SC200_SetPos(sn,1, 1)==1)//通道 1 设置值为 1
{
    //设置成功
}
```

c# 示例代码

```
[DllImport("sc200drv.dll", EntryPoint = "SC200_SetPos", CallingConvention =
CallingConvention.Cdecl, CharSet = CharSet.Ansi)]
private static extern int SC200_SetPos(int index,int channel, float pos);
int sn = devsn[0];
if(SC200_SetPos(sn,1, 1)==1)//通道 1 设置值为 1
{
    //设置成功
}
```

SC200_GetPos(int index,int channel, float *pos)

描述 ：读取当前值

参数 ：index 控制器序号；channel 通道号 1~3；pos 为返回的当前位置值

返回值：1 读取成功，0 读取失败。

VC 示例代码：

```
int sn = devsn[0];
float pos=0;
if(SC200_GetPos(sn,1, &pos)==1)//读取通道 1 的当前位置值存储在 pos 变量里
{
    //读取成功
}
```

c# 示例代码

```
[DllImport("sc200drv.dll", EntryPoint = "SC200_GetPos", CallingConvention =
CallingConvention.Cdecl, CharSet = CharSet.Ansi)]
private static extern int SC200_GetPos(int index,int channel, ref float pos);
int sn = devsn[0];
float pos=0;
if(SC200_GetPos(sn,1, ref pos)==1)//读取通道 1 的当前位置值存储在 pos 变量里
{
    //读取成功
}
```