# 验证

1. 基本类型，如，char， int，double，ushort，byte等，可处理到二级指针，如int, int\*, int\*\*
2. 结构体，可处理到一级指针，如struct\*
3. 结构体内可含有一级指针，二级指针暂时处理不了，结构体内可含有结构体。

# 示例一

## C++结构体

typedef struct {

BYTE ipAddress[4];

WORD portNo;

BYTE reserved[2];

} CL3IF\_ETHERNET\_SETTING;

## C#结构体

[StructLayout(LayoutKind.Sequential)]

public struct CL3IF\_ETHERNET\_SETTING

{

[MarshalAs(UnmanagedType.ByValArray, SizeConst = 4)]

public byte[] ipAddress;

public ushort portNo;

[MarshalAs(UnmanagedType.ByValArray, SizeConst = 2)]

public byte[] reserved;

};

## C++函数

CL3\_IF\_API LONG WINAPI CL3IF\_OpenEthernetCommunication(LONG deviceId, CL3IF\_ETHERNET\_SETTING\* ethernetSetting, DWORD timeout);

## C#函数

[DllImport(DllName)]

internal static extern int CL3IF\_OpenEthernetCommunication(int deviceId, ref CL3IF\_ETHERNET\_SETTING ethernetSetting, uint timeout);

## 传参过程

CL3IF\_ETHERNET\_SETTING ethernetSetting = new CL3IF\_ETHERNET\_SETTING();

ethernetSetting.ipAddress = new byte[4];

ethernetSetting.reserved = new byte[2];

int returnCode = CLP030.CL3IF\_OpenEthernetCommunication(0, ref ethernetSetting, 2000);

总结：用C++ dll封装API时候，如果是数组，就写 int[]形式，如果是一个变量的指针，就写 int \*

用C#封装时，如果看到int[]，就写int[],传进去一个数组，如果看到int \*，就创建一个int变量，写 ref int 形式。

## 总结

1. C#的封装形式和C++形式相同，但是C++的数组不需要new就已经有了内存，而C#需要new出内存，为结构体内的数组一一分配内存，才能传入参数。
2. 基元类型不需要分配内存，因为C#的基元类型是结构体，内存分配在栈上。
3. C++的参数如果是一维指针，如double\* tag. 那么C#的封装参数形式有三种：ref double tag， out double tag，double[ ] tag. 如果tag是传入参数变量，供C++函数读取使用，那么就用ref double tag。如果tag是传出参数变量，供C++函数写入传出结果，那么就用out double tag。如果tag是个数组，则采用tag[ ]. 针对数组参数，C++内部不分配任何内存，只是拿到外部分配好的内存的指针，读写操纵内存。C#分配的内存大小最好不小于C++操纵的范围。
4. C++结构体中含有数组，一般是int[2] arr, C#结构体中是int[ ] arr, SizeConst = 2，数组的大小放到外面了。
5. 往封装后的C#函数中传递参数的时候，要保证每一个变量都有了内存，而不是只有个引用，没有对照的堆中内存。

# 示例二

[StructLayout(LayoutKind.Sequential)]

public struct CL3IF\_VERSION\_INFO

{

public int majorNumber;

public int minorNumber;

public int revisionNumber;

public int buildNumber;

};

# 示例三

结构体中含有结构体变量和结构体数组。

[StructLayout(LayoutKind.Sequential)]

public struct CL3IF\_MEASUREMENT\_DATA

{

public CL3IF\_ADD\_INFO addInfo;

public CL3IF\_OUTMEASUREMENT\_DATA[] outMeasurementData;

};

## C++函数

Void Fun(my\_struct \* ms);