# GPIO Driver 使用方法

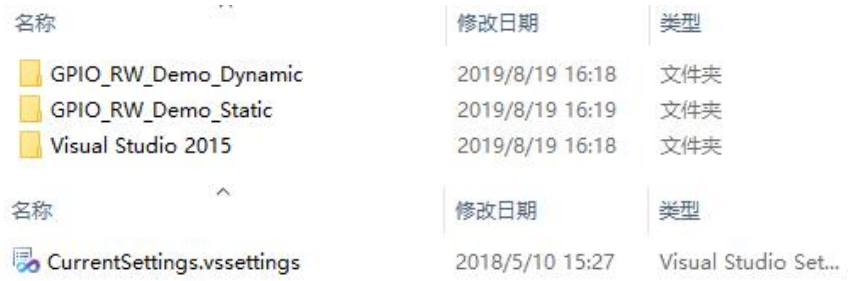
1. 文件包含


文件包含有：Binary、Demo_Code、Dll_Source_Code

1) Binaries 中包含 32 位/64 位的可调用文件，以及两个 C++编写的 Demo 可执行文件。如下：


其中，请确保调用文件在执行时与可执行文件(.exe)位于同一目录)。另外，两种 EXE 分别是静态调用编译出的和动态调用编译出的。

2) Demo_Code 中包含调用函数的 C++例代码、此示例产生的可执行文件、Visual Studio 的编译设置。




3) Source 中包含了 SioGpioDriver 的原工程(包含代码)、编译设置。


此为 API 的核心代码，如有需要，可自行修改。

2. 调用方式

1) C/C++静态调用。

```
#include "stdafx.h"
#include "GPIO_RW_Demo.h"
#include "GPIO_RW_DemoDlg.h"
#include "afxdialogex.h"
#include "SioGpioDriver.h"                                  #1

#ifdef _M_X64                                               #2
#pragma comment(lib, "rely/x64/SioGpioDriver.lib")
#else
#pragma comment(lib, "rely/x32/SioGpioDriver.lib")
#endif

#ifdef _DEBUG
#define new DEBUG_NEW
#endif


// CGPIO_RW_DemoDlg 对话框


CGPIO_RW_DemoDlg::CGPIO_RW_DemoDlg(CWnd* pParent /*=NULL*/)
    : CDialogEx(IDD_GPIO_RW_DEMO_DIALOG, pParent)
{
    m_hIcon = AfxGetApp()->LoadIcon(IDR_MAINFRAME);
}

void CGPIO_RW_DemoDlg::DoDataExchange(CDataExchange* pDX)
{
    CDialogEx::DoDataExchange(pDX);
}

BEGIN_MESSAGE_MAP(CGPIO_RW_DemoDlg, CDialogEx)
    ON_WM_PAINT()
    ON_WM_QUERYDRAGICON()
    ON_BN_CLICKED(IDC_BUTTON1, &CGPIO_RW_DemoDlg::OnBnClickedButton1)
    ON_BN_CLICKED(IDC_BUTTON2, &CGPIO_RW_DemoDlg::OnBnClickedButton2)
    ON_BN_CLICKED(IDOK, &CGPIO_RW_DemoDlg::OnBnClickedOk)
END_MESSAGE_MAP()


// CGPIO_RW_DemoDlg 消息处理程序

BOOL CGPIO_RW_DemoDlg::OnInitDialog()
{
    CDialogEx::OnInitDialog();

    // 设置此对话框的图标。  当应用程序主窗口不是对话框时，框架将自动
    //  执行此操作
    SetIcon(m_hIcon, TRUE);          // 设置大图标
    SetIcon(m_hIcon, FALSE);         // 设置小图标

    // TODO: 在此添加额外的初始化代码
    if (!SioGpioDriverOpen())                               #3
    {
```

```
156     rsv = SioGpioGetVal(PinNumber_hex);
157     if (-1 == rsv)
158     {
159         GetDlgItem(IDC_EDIT3)->SetWindowText(_T("Read Failed!"));
160         return;
161     }
162     if (((CButton*)GetDlgItem(IDC_RADIO1))->GetCheck())
163         value = 0;
164     else
165         value = 1;
166
167     if (SioGpioSetVal(PinNumber_hex, value))
168     {
169         if (value)
170             GetDlgItem(IDC_EDIT3)->SetWindowText(_T("Write bit to 1 OK !"));
171         else
172             GetDlgItem(IDC_EDIT3)->SetWindowText(_T("Write bit to 0 OK !"));
173     }
174 }
175
176
177 void CGPIO_RW_DemoDlg::OnBnClickedOk()
178 {
179     // TODO: 在此添加控件通知处理程序代码
180     SioGpioDriverClose();
181     CDialogEx::OnOK();
182 }
```

#4

#5

① #1 引用.h 头文件，声明库中函数，注意，编译时要将文件放在合适的目录下。

② #2 调用静态库中函数的定义。

③ #3 调用启动函数，安装 Driver。

④ #4 使用库中的函数。

⑤ #5 程序运行结束前务必卸载 Driver。

2) C/C++动态调用。

```
typedef int(_stdcall *SioGpioDriverOpen) ();
typedef void(_stdcall *SioGpioDriverClose) ();

typedef int(_stdcall *SioGpioSetVal) (unsigned short PinNum, BYTE Value);
typedef int(_stdcall *SioGpioGetVal) (unsigned short PinNum);
```

#1

```
HMODULE hlib;
```

#2

```
SioGpioDriverOpen sioGpioDriverOpen = NULL;
SioGpioDriverClose sioGpioDriverClose = NULL;
SioGpioSetVal sioGpioSetVal = NULL;
SioGpioGetVal sioGpioGetVal = NULL;
```

#3

```
typedef VOID(__stdcall *LPFN_PGNSI)(LPSYSTEM_INFO lpSystemInfo);

BOOL Is64Bit_OS()  //not in use
{
    BOOL bRetVal = FALSE;
    SYSTEM_INFO si = { 0 };
    LPFN_PGNSI pGNSI = (LPFN_PGNSI)GetProcAddress(GetModuleHandle(_T("kernel32.dll")), "GetNativeSyste
    if (pGNSI == NULL)
    {
        return FALSE;
    }
    pGNSI(&si);
    if (si.wProcessorArchitecture == PROCESSOR_ARCHITECTURE_AMD64 ||
        si.wProcessorArchitecture == PROCESSOR_ARCHITECTURE_IA64)
    {
        bRetVal = TRUE;
    }
    return bRetVal;
}


int InitSioGpioLib(void)
{
#ifdef _M_X64
    hlib = LoadLibrary(L"SioGpioDriver64.dll");
#else
    hlib = LoadLibrary(L"SioGpioDriver32.dll");
#endif
```

#4

```
    sioGpioDriverOpen = (SioGpioDriverOpen)GetProcAddress(hlib, "SioGpioDriverOpen");
    sioGpioDriverClose = (SioGpioDriverClose)GetProcAddress(hlib, "SioGpioDriverClose");
    sioGpioSetVal = (SioGpioSetVal)GetProcAddress(hlib, "SioGpioSetVal");
    sioGpioGetVal = (SioGpioGetVal)GetProcAddress(hlib, "SioGpioGetVal");
```

#5

```
    return sioGpioDriverOpen();
}
```

#6

```
    if (sioGpioSetVal(PinNumber_hex, value))
    {
        if (value)
            GetDlgItem(IDC_EDIT3)->SetWindowText(_T("Write bit to 1 OK !"));
        else
            GetDlgItem(IDC_EDIT3)->SetWindowText(_T("Write bit to 0 OK !"));
    }
}


void CGPIO_RW_DemoDlg::OnBnClickedOk()
{
    // TODO: 在此添加控件通知处理程序代码
    sioGpioDriverClose();
    CDialogEx::OnOK();
}
```

#7

#8

① #1 声明一套函数类型，与所要调用的函数相同。
② #2 声明一个句柄，用于调用动态链接库。
③ #3 实例化刚刚声明的函数类型。
④ #4 使用句柄和“LoadLibrary()”函数来调用动态链接库文件。
⑤ #5 使用函数对象承接句柄中引用的函数
⑥ #6 程序开始时调用“开始函数”，启动并安装 Driver。
⑦ #7 使用寄存器读写函数，也可使用本库中的其它函数。
⑧ #8 程序结束时调用“关闭函数”，终止并卸载 Driver

3. IO 寄存器读写，实现读取 GPIO 状态和控制 GPO 的高低电平

GPIO 地址已映射于 IO 空间，只需访问 GPIO 的标号即可，比如访问 GPI15，调用函数时输入参数 15 即可。GPI（DI）寄存器只读，GPO（DO）寄存器可读写，规格书已写明 GPIO 是 GPI（DI）还是 GPO（DO），请见下表。

注意：下表示例仅作参考使用，用户须以具体产品规格书提供的 GPIO 地址对应关系调整 IO 映射空间，才能正确映射，从而正确访问。

| 功能 | GPIO 名称 | 功能 | GPIO 名称 |
|---|---|---|---|
| DI0 | SIO_GP15 | DO0 | SIO_GP47 |
| DI1 | SIO_GP16 | DO1 | SIO_GP51 |
| DI2 | SIO_GP33 | DO2 | SIO_GP64 |
| DI3 | SIO_GP37 | DO3 | SIO_GP65 |