

MPS-140801
8 通道 24 位 USB 信号采集卡
使用说明

Ver 1.0

第一章 产品概述



MPS-140801 8通道 24位 USB 信号采集卡

一、 产品简介

MPS-140801 是一款基于 USB 总线的高性能信号采集卡。MPS-140801 具有八路大量程、高采样率、低噪声的高性能同步信号采集通道。每个通道的量程为 $\pm 10V$ ，采样率高达 52.734Ksps，并能保证实时传输到计算机进行显示与分析。通过高性能 ADC 和先进的 DSP 信号处理技术，MPS-140801 同时还具有极低的采样噪声，在 1.643Ksps 采样率下采样噪声峰峰值仅为 0.00004V，满量程信号的信噪比高达 50 万。多通道、高采样率和低噪声和同步采样使 MPS-140801 能够满足科研与生产中绝大部分信号采集工程的需要。

MPS-140801 采用 USB2.0 高速总线接口，支持即插即用和热插拔，是便携式系统用户的最佳选择。板载 191K 超大 FIFO 缓存，能最大限度防止数据丢失，保证信号采集的完整性。通过扩展计算机的 USB 接口，还能同时将多达 10 套的 MPS-140801 连接到一台计算机，从而扩展了系统的通道个数。

MPS-140801 采用跨平台通用的动态链接库作为驱动函数接口，可工作在 Win9X/Me、Win2000/XP/WIN 7 等常用操作系统下，支持 VB, VC, C++Builder, Dephi, LabVIEW, Matlab 等大多数编程语言，硬件驱动方式清晰明了，软件编程简单快捷。

二、 性能指标

2.1、USB 总线

- USB2.0 高速总线传输，最大传输速度达 480Mb/s
- 支持热插拔和即插即用

2.2、输入通道

- 8 路全差分输入通道
- 同步采集

2.3、电压量程

- $-10V$ 到 $+10V$

2.4、输入阻抗

- 标准型：= $1M\Omega$
- 高阻型：> $1G\Omega$

2.5、采样率

- 52.734K
- 26.367K
- 13.183K
- 6.592K
- 3.286K
- 1.643K

2.6、分辨率

- 52.734K 采样率下，噪声峰峰值 < 400 微伏，无噪声分辨率 15.7bit，有效分辨率 18.6bit，信噪比 103dB；
- 26.367K 分辨率下，噪声峰峰值 < 250 微伏，无噪声分辨率 16.2bit；噪声有效值 < 38 微伏，有效分辨率 19.0bit，信噪比 105dB；
- 13.183K 采样率下，噪声峰峰值 < 150 微伏，无噪声分辨率 17.0bit；噪声有效值 < 23 微伏，有效分辨率 19.8bit，信噪比 109dB；
- 6.592K 采样率下，噪声峰峰值 < 100 微伏，无噪声分辨率 17.7bit；噪声有效值 < 16 微伏，有效分辨率 20.6bit，信噪比 113dB；
- 3.286K 采样率下，噪声峰峰值 < 60 微伏，无噪声分辨率 18.6bit；噪声有效值 < 9 微伏，有效分辨率 21.5bit，信噪比 117dB；
- 1.643K 采样率下，噪声峰峰值 < 40 微伏，无噪声分辨率 18.9bit；噪声有效值 < 6 微伏，信噪比 121dB；

2.7、缓存

- DAQ Buffer: 192K Bytes
- USB FIFO : 1K Bytes

2.8、工作温度

- 0℃ - 85℃

三、应用领域

便携式仪表和测试设备
高精度信号采集与记录
振动与声音信号分析

四、软件资源

Windows95/98/NT/2000/XP/WIN7 操作系统驱动程序；通用 DLL 动态链接库及编程参考例程；附送多功能信号采集及信号回放软件等。

五、配件清单

- [1] MPS-140801 信号采集卡一张；
- [2] 高屏蔽 USB 数据传输电缆一根；
- [3] 信号线 10 根；
- [4] 同步信号传输线 1 根；
- [5] 外接备用电源一个；
- [6] 保修卡一张；

六、售后服务

保修一年。

第二章 设备安装

一、 MPS-140801 信号采集卡硬件接口

GND: 设备信号地
CHx: 差分电压信号输入, x 代表通道序号
DC-5V: 备用外接 5V 电源接口
POWER: 电源状态指示灯
DAQ: 采集状态指示灯
USB: USB 数据线接口
SYNCl: 同步触发信号输入
SYNCo: 同步触发信号输出

二、 接口说明

- 使用设备时请通过 USB 数据线将设备与计算机连接, 设备正常连接后, 可在 WINDOWS 的设备管理器中查询到设备信息。
- 一般情况下, 设备依靠 USB 供电即可正常工作。如出现供电不足, 请使用备用外接电源进行供电。
- 设备可以通过软件启动采集, 也可通过外部触发信号来启动采集。外部信号启动时, 在设备的“同步触发输入端”输入一个脉冲信号, 设备将在脉冲的下降沿启动采集。不管是软件启动还是硬件启动, 设备在正常启动后, 都将通过“同步触发信号输出端”向外输出一个脉冲。该输出脉冲可用来启动其他设备的采集, 以实现多块板的启动进行同步。终止采集时只能通过软件终止。
- 进行信号采集时, 须先将设备连接到计算机 (或上电), 再接入外部信号 (或给信号源上电)。
- 信号接头的外侧为负输入端, 内芯为正输入端。采集到的电压为正输入端电压与负输入端的差值。对于单端信号, 可将信号源的输出信号与信号源的地作为正负输入。
- 设备 GND 输出接头的正端和负端都是 GND, 与计算机的地线相通。当外部信号的信号源是浮地时, 建议将设备 GND 输出与信号源地线短接, 以实现共地。在信号源不是浮地时, 请谨慎选择共地, 以免对设备造成损害。
- 设备的红色 LED (POWER) 为电源指示灯。LED 亮显示系统电源正常。
- 设备的蓝色 LED (DAQ) 为采集与传输的状态指示灯。LED 亮则系统正在正常采集; LED 灭, 则采集停止或数据传输中断。

三、 驱动安装

1. 首次使用本卡时, 计算机将提示“发现新硬件”, 如下图所示。选择“从列表或指定位置安装 (高级)”, 并点击“下一步”。



2. 选择“在搜索中包括这个位置”，并点击“浏览”选择 MPS-140801 驱动文件所在的文件夹，点击“下一步”。



3. 开始安装驱动。



4. 驱动安装完成。



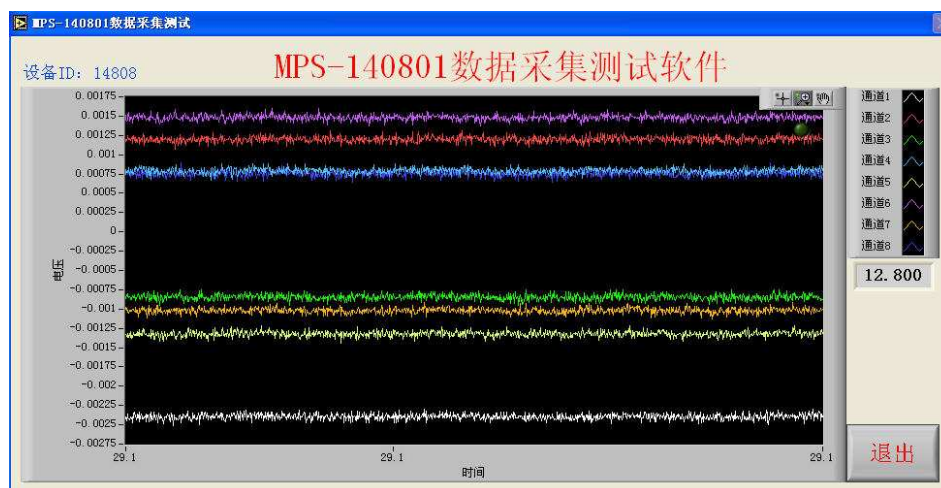
四、 基本功能测试



图一 设置采样率



图二 数据采集界面



图三 自动调整 Y 轴显示范围

MPS-140801 各项基本功能测试:

1. 将 MPS-140801 信号采集卡与计算机通过 USB 接口连接。
2. 按第三小节所描述的过程安装驱动程序。
3. 打开测试程序。
4. 出现图一的界面，选择合适的采样速率，并点击确定。
5. 出现图二的界面，波形图中将显示八个输入通道的信号波形。
6. 波形图左上角将显示当前设备的 ID 编号。
7. 在波形图上点击右键，可以进行一些特殊操作。其中“Copy Data”可以复制当前信号数据；“Export Simplified Image...”可以保存当前波形图；“Clear Chart”可以清空显示；“AutoScale Y”可以使波形图的 Y 轴显示范围自动适应当前信号，以获得最佳的显示效果。
8. 修改波形图电压轴的最大值和最小值可以控制显示范围，但最大不应超过量程($\pm 10V$)。图三为修改 Y 轴（自适应模式）后的信号显示图。
9. 波形图右上角的控件可以对波形显示进行调整，如缩放、定位、拖动波形等。
10. 波形图右侧标注不同通道波形的显示颜色，此处点击右键可以修改信号的绘图特性。
11. 软件右侧中部显示当前采样率的计算值，与实际的硬件采样率近似。
12. 点击右侧底部的“退出”按钮退出程序。
13. 测试过程中，建议将闲置的输入通道短接后接地(GND)，以免干扰有效通道的内容显示。
14. 测试过程中，设备硬件绿色指示灯应保持明亮不闪烁。出现闪烁说明有数据丢失；频繁闪烁说明计算机性能不足以支持当前的采集与显示，请选择较低的采样率或者升级计算机。指示灯熄灭说明信号采集或数据传输被终止。

第三章 用户编程

一、 动态链接库 (DLL)

MPS-140801 采用 DLL (Dynamic Linkable Library, 动态链接库) 的方式来进行编程驱动。DLL 的编制与具体的编程语言及编译器无关, 只要遵循约定的 DLL 接口规范和调用方式, 用各种语言编写的 DLL 都可以相互调用。

DLL 可以方便的在 VC、VB、LabVIEW 等语言下被调用, 具体方式分别为:

- VC 下调用 DLL

```
typedef void ( * FUNC )(void);           //定义一个函数指针
FUNC Func;                               //定义一个函数指针变量
HINSTANCE hDLL=LoadLibrary("DllTest.dll"); //加载 dll
Func=(FUNC)GetProcAddress(hDLL, "FuncInDLL"); //找到 dll 中的函数
Func();                                  //调用 dll 里的函数
```

- VB 下调用 DLL

```
[Public | Private] Declare Function name Lib " libname " [Alias "
aliasname " ] ([arglist]) [ As type] "
```

Public (可选) 用于声明在所有模块中的所有过程都可以使用的函数; **Private** (可选) 用于声明只能在包含该声明的模块中使用的函数。

Name (必选) 任何合法的函数名。动态链接库的入口处 (entry points) 区分大小写。

Libname (必选) 包含所声明的函数动态链接库名或代码资源名。

Alias (可选) 表示将被调用的函数在动态链接库 (DLL) 中还有另外的名称。当外部函数名与某个函数重名时, 就可以使用这个参数。当动态链接库的函数与同一范围内的公用变量、常数或任何其它过程的名称相同时, 也可以使用 **Alias**。如果该动态链接库函数中的某个字符不符合动态链接库的命名约定时, 也可以使用 **Alias**。

Aliasname (可选) 动态链接库。如果首字符不是数字符号 (#), 则 **aliasname** 是动态链接库中该函数入口处的名称。如果首字符是 (#), 则随后的字符必须指定该函数入口处的顺序号。

Arglist (可选) 代表调用该函数时需要传递参数的变量表。

Type (可选) **Function** 返回值的数据类型; 可以是 **Byte**、**Boolean**、**Integer**、**Long**、**Currency**、**Single**、**Double**、**Decimal** (目前尚不支持)、**Date**、**String** (只支持变长) 或 **Variant**, 用户定义类型, 或对象类型。

arglist 参数的语法如下:

```
[Optional] [ByVal | ByRef] [ParamArray] varname [()] [As type]
```

Optional (可选) 表示参数不是必需的。如果使用该选项, 则 **arglist** 中的后续参数都必需是可选的, 而且必须都使用 **Optional** 关键字声明。如果使用了 **ParamArray**, 则任何参数都不能使用 **Optional**。

ByVal (可选) 表示该参数按值传递。

ByRef (可选) 表示该参数按地址传递。

- LabVIEW 下调用 DLL

在 LabVIEW 中, 调用 DLL 是通过 CLF 节点来完成的。所谓 CLF 节点 (Call Library Function, 调用函数库节点), 是指可以在 LabVIEW 调用其他语言封装的 DLL, CLF 节点位于位于 LabVIEW 功能模板中的 Advanced 子模板中, 其配置过程如下:

- 在 CLF 节点的右键菜单中选择“Configure”，弹出 CLF 节点配置对话框；
- 点击“Browse”按钮，在随后弹出的选择 DLL 文件对话框中找到你需要用的 DLL 文件，此时，LabVIEW 就会自动装载选定的 DLL 文件，并检测 DLL 文件中所包含函数。但是函数中的参数和参数的数据类型需要用户根据函数的输入、输出参数手动设置。因而在调用 DLL 文件时，要求用户对 DLL 文件有较为详细的了解。
- 在 FunctionName 下拉列表框中选定动态连接库中所包含的所需要 API 函数；
- 在 Calling Convention 下拉菜单中选择 StdCall (WINAPI) 和 C 两个选项，若用户选定的是 Windows API 函数，则选用 StdCall (WINAPI) 选项；若用户选用的 DLL 中的函数是非 Windows API 函数，则选用 C 选项；
- 设置函数的返回参数。函数参数的类型要与 DLL 中函数本身所定义的函数参数类型相对应，如果不对应，函数就会出现数据错误和强制类型转换；
- 根据所选函数的函数原型，设置函数的输入参数及数据类型。点击 Add a Parameter 按钮，即可以添加一个新的输入参数。

二、编程函数及参数

MPS-140801 提供的驱动 DLL 文件名为 MPS-140801.dll，内部共有五个驱动函数，分别为：

- `Handle OpenDevice (int DeviceNumber)`

`Handle OpenDevice` 函数执行打开设备的功能。如设备打开成功将返回设备的句柄；打开失败则返回-1。

`int DeviceNumber` 当前打开设备的序号。当有多套 MPS-140801 设备同时连接到计算机时，将按照设备连接到计算机的先后顺序从 0 到 9 依次分配序号，打开特定序号的设备所返回的句柄将用于对设备的后续操作。只有一块卡连接时，默认设备号为 0。最多支持同时连接 10 个设备。

- `int GetDeviceID (Handle DeviceHandle)`

`int GetDeviceID` 函数执行获取设备 ID 编号的功能。每套 MPS-140801 设备有唯一的 ID 号与之对应，ID 号可用于多套设备同时使用时对设备进行标识。若函数执行成功，返回设备 ID 号；执行失败返回 0。

`Handle DeviceHandle`：操作所针对的设备句柄。

- `int Control (int Command, Handle DeviceHandle)`

`int Control` 函数执行设置设备参数的功能。若函数执行成功，返回 1；执行失败返回 0。

`int Command`：MPS-140801 的配置参数代码。其值定义如下：

- 0：停止采集；
- 1：开始采集；
- 2：设置采样率为 26.367K；
- 3：设置采样率为 13.183K；
- 4：设置采样率为 6.592K；
- 5：设置采样率为 3.286K；
- 6：设置采样率为 1.643K；

7: 设置采样率为 52.734K;

Handle DeviceHandle: 操作所针对的设备句柄。

- `int DAQ(int * DataBuffer, int SampleNumber , Handle DeviceHandle)`

`int DAQ` 函数执行获取数据的功能。若函数执行成功，返回 1；执行失败返回 0。

`int * DataBuffer`: 采集数据的缓存区首地址。若函数执行成功，该数组内数据被自动更新为最新采集到的数据（更新的元素个数由 SampleNumber 决定）；若函数执行失败，该数组内数据无效。DataBuffer 为一个一维数组，其每个元素依次循环对应一个通道采样点的数据，如 DataBuffer[0]对应 CH1 的第一个采样点数据，DataBuffer[1]对应 CH2 的第一个采样点数据，……DataBuffer[7]对应 CH8 的第一个采样点数据，DataBuffer[8]对应 CH1 的第二个采样点数据……DataBuffer[15]对应 CH8 的第二个采样点数据，以此类推。每个采样点数据是一个 int 型的 32 位带符号整形数据，其值大小与信号电压值的对应关系为： **$Voltage1[0] = ((double)DataBuffer[0]/8388608) * 10.0352$** ，转换时注意预先将整型数据转换为浮点型，以免在运算过程中损失精度。

`int SampleNumber`: 函数支持一次采集的样点个数。该参数决定函数执行一次数据数组中所更新的数据个数，当从采集卡中读到 SampleNumber 个数据点后函数成功返回。该参数的最小值为 128，最大值为 8192，设定值应为 128 的整倍数，否则根据向下就近原则自动配置为 128 的倍数。

Handle DeviceHandle: 操作所针对的设备句柄。

- `int CloseDevice (Handle DeviceHandle)`

`int CloseDevice` 函数执行关闭设备的功能。若函数执行成功，返回 1；执行失败返回 0。设备打开并执行完所有操作后，必须对设备进行关闭。若没有经过关闭而意外退出，必须关闭所有与设备操作函数相关的程序线程，并重新连接设备硬件。

Handle DeviceHandle: 操作所针对的设备句柄。

三、 编程范例

//基本流程：打开设备→获取ID→设置采样率→开始采集→循环获取数据→停止采集→关闭设备

```
#define StopDAQ 0           //开始采集数据
#define StartDAQ 1          //停止采集数据
#define Set26KRate 2        //设置采样率为26.367K
#define Set13KRate 3        //设置采样率为13.183K
#define Set6KRate 4          //设置采样率为6.592K
#define Set3KRate 5          //设置采样率为3.296K
#define Set1KRate 6          //设置采样率为1.648K
#define Handle int           //定义Handle为int类型
```

```
int TEST()
```

```
{
```

```
//DLL函数的声明，一般置于程序初始化部分，声明一次后即可随意调用所声明的函数
HINSTANCE hDll; //打开DLL
hDll=LoadLibrary("MPS-140801.dll");
if (NULL==hDll)
{
    AfxMessageBox("Cann't find DLL");
    return 0;
}

typedef Handle(*lpOpenDevice)(int DeviceNumber); //打开设备函数的声明
lpOpenDevice OpenDevice=(lpOpenDevice)GetProcAddress(hDll, "OpenDevice");
if (NULL==OpenDevice)
{
    AfxMessageBox("Cann't find <OpenDevice> function");
}

typedef int(*lpCloseDevice)(Handle DeviceHandle); //关闭设备函数的声明
lpCloseDevice CloseDevice=(lpCloseDevice)GetProcAddress(hDll, "CloseDevice");
if (NULL==CloseDevice)
{
    AfxMessageBox("Cann't find <CloseDevice> function");
}

typedef int(*lpControl)(int Command, Handle DeviceHandle); //向设备发送设置命令函数的
声明
lpControl Control=(lpControl)GetProcAddress(hDll, "Control");
if (NULL==Control)
{
    AfxMessageBox("Cann't find <Control> function");
}

typedef int(*lpDAQ)(int *dataArray, int SampleNumber, Handle DeviceHandle); //采集函数的
声明
lpDAQ DAQ=(lpDAQ)GetProcAddress(hDll, "DAQ");
if (NULL==DAQ)
{
    AfxMessageBox("Cann't find <DAQ> function");
}

typedef int(*lpGetDeviceID)(Handle DeviceHandle); //获取板卡ID序号函数的声明
lpGetDeviceID GetDeviceID=(lpGetDeviceID)GetProcAddress(hDll, "GetDeviceID");
if (NULL==GetDeviceID)
{
    AfxMessageBox("Cann't find <GetDeviceID> function");
}
```

```
}

//数据采集程序主体，用于完成数据采集的工作
Handle DeviceHandle;
int Buffer[1024*8] = {0}; //变量定义Buffer为数据缓存数组，用来临时保存采集到的数据，以
待后续程序处理
double Voltage[8][1024];
int flag = 1; //函数执行成功标志
int i= 0, j = 0;
int DeviceID;

//数据采集前的准备，包括打开设备、获取ID、设置参数、启动采集等
DeviceHandle = OpenDevice(0); //打开设备
if(DeviceHandle == -1) //若打开失败，报错并返回
{
    AfxMessageBox("OpenDeviceError");
    return 0;
}

DeviceID = GetDeviceID(DeviceHandle); //获取ID

Control(Set13KRate, DeviceHandle); //设置采样率为.183K

Control(StartDAQ, DeviceHandle); //开始采集数据

//循环调用采集函数，这里以不停的获取数据并进行分析处理为例，实际应用中根据实际需要设置循
环跳出的条件
// while(1)
{
    flag = DAQ(Buffer, 1024*8, DeviceHandle); //数据采集
    if(flag != 0) //如果采集成功
    {
        for(i = 0; i < 1024; i++) //在此添加数据处理及其他代码
        {
            //8个通道的数据在缓存中循环排列；电压值= (采集值/(65536*128))*10.0352
            Voltage[0][i] = ((double)Buffer[i*8] / (65536 * 128)) * 10.0352;
            //通道1的电压
            Voltage[1][i] = ((double)Buffer[i*8 + 1] / (65536 * 128)) * 10.0352;
            //通道2的电压
            Voltage[2][i] = ((double)Buffer[i*8 + 2] / (65536 * 128)) * 10.0352;
            //通道3的电压
            Voltage[3][i] = ((double)Buffer[i*8 + 3] / (65536 * 128)) * 10.0352;
            Voltage[4][i] = ((double)Buffer[i*8 + 4] / (65536 * 128)) * 10.0352;
            Voltage[5][i] = ((double)Buffer[i*8 + 5] / (65536 * 128)) * 10.0352;
```

```
Voltage[6][i] = ((double)Buffer[i*8 + 6] / (65536 * 128)) * 10.0352;
Voltage[7][i] = ((double)Buffer[i*8 + 7] / (65536 * 128)) * 10.0352;
    }
    AfxMessageBox("DAQSuccess");
}
else
{
    AfxMessageBox("DAQError"); //如果采集失败报错并跳出采集循环
//    break;
}

}

//停止采集与关闭设备
Control(StopDAQ, DeviceHandle); //停止采集
CloseDevice(DeviceHandle); //关闭设备
return 1;
}
```

第四章 注意事项

- 拔插设备接线端口请用力适度，以免损害接口。
- 一般情况下设备可通过计算机 USB 接口供电进行工作，如出现供电不足，请使用标配的外接电源进行供电，禁止随意更换其他外部电源。
- 信号输入端允许接入的电压范围为-10V 至+10V，超出量程之外的电压可能损害设备。
- 用户须注意电源的开关顺序，使用时要求先将设备上电或连至计算机，而后开启信号源电源；关闭时先关信号源电源，后将设备断电。
- 将采集卡拔离计算机后，请间隔一定时间再重新插入。拔插过快有可能造成计算机系统配置驱动异常，此时请重新拔插采集卡。
- 本设备属于精密电子仪器，请注意妥善保管，注意防尘防潮与防静电。
- 禁止用户自行打开设备外壳，一经打开即失去保修资格，出现的故障与其他后果由用户自行承担。
- MPS-140801 自出厂之日起一年内，凡用户遵守贮存、运输和使用要求，由于产品质量导致的故障，凭保修卡免费维修。因违反操作规定和使用要求而造成损坏的，需交纳器件维修费。
- MPS 系列信号采集卡由 Morpheus Electronic 提供，更多产品和相关信息请浏览：www.mps-electronic.com.cn, 咨询邮箱 mail@mps-electronic.com.cn。