

ACM9226 应用手册

——高速数据采集程序

小梅哥编写，未经许可，严禁用于任何商业用途

本文中所涉及到的文件均可在 www.corecourse.cn 网站上搜索 “【产品资料】ACM9226” 就能够找到了。如果找不到，也可在我们的芯航线 FPGA 技术群内咨询索取。

特别说明：本资料包会提供两个工程文件，一个是帧间间隔为 2 秒的，主要用于使用 matlab 手动接收数据并绘图，名为“ACM9226_AC606_UART_2s.rar”，另一个是使用该软件来实时绘制波形的，名为“ACM9226_AC606_UART_20ms.rar”。在后续实验中请根据使用场景选择对应的工程文件。

摘要：本应用完成了双通道 AD9226 高速数据采集和串口传输的功能。双路 AD9226 以 50M 的采样率实时采集数据并检测是否达到触发条件，所采样数据在达到设定的触发条件后，系统将设定的触发条件发生之前的 256 个数据和触发点开始的 3840（4096-256）个数据存储在片上 RAM 中，然后由串口将数据发送到 PC 机上，PC 使用串口接收数据，将接收到的数据使用 Matlab 绘制成波形。

本应用需要 3 个按键、4 个 LED 灯、双通道 AD9226 高速采集模块、一个 USB 转串口功能。其中，3 个按键用于设定发送哪个通道的数据，LED 灯的 0 和 1 则指示 2 个通道的数据是否发送的状态。触发采样时可以设定由具体哪个通道触发采样。LED 灯的 2 和 3 指示当前设定的触发通道。具体对应关系为：

按键	功能状态	LED[0] 通道 0 使能	LED[1] 通道 1 使能	LED[2] 通道 0 触发	LED[3] 通道 1 触发	触发 通道
按键 0	使能通道 0 数据 发送	亮	灭	亮	灭	通道 0

按键 1	使能通道 1 数据 发送	灭	亮	灭	亮	通道 1
按键 2	使能通道 0 和通 道 1 数据发送	亮	亮	亮	灭	通道 0

程序中为了后期扩展升级方便，使用了 NIOS II CPU 作为主控系统。NIOS II CPU 根据按键设置的采样触发状态，设置触发通道，然后使能采样并检测采样完成状态，一旦检测到一次突发采样完成，则根据按键设置的通道发送使能状态，将对应通道的数据通过串口发送。对应的程序代码如下所示：

```
#include "sys/alt_stdio.h"
#include "altera_avalon_pio_regs.h"
#include "altera_avalon_uart_regs.h"
#include "system.h"

int main()
{
    char *p;

    char data1,data2;
    char channel_set=0;

    int i;

    while(1)
    {
        //读取通道设定状态
        channel_set = IORD_ALTERA_AVALON_PIO_DATA(CHANNEL_SET_BASE);

        if(channel_set & 0x1) //设置通道1触发
        {
            IOWR_ALTERA_AVALON_PIO_DATA(TRIG_BASE, 2047 | 0x8000);
        }
        else if(channel_set & 0x2) //设置通道2触发
        {
            IOWR_ALTERA_AVALON_PIO_DATA(TRIG_BASE, (2047 | 0x8000) << 16);
        }

        //产生采样使能脉冲信号
        IOWR_ALTERA_AVALON_PIO_DATA(SAMPLE_EN_BASE,1);
    }
}
```

```
IOWR_ALTERA_AVALON_PIO_DATA(SAMPLE_EN_BASE,0);

//等待采样完成
while(!IORD_ALTERA_AVALON_PIO_DATA(STATE_BASE));

p = DPRAM_BASE;

for(i=0;i<16384;)
{
    data1 = *p;
    p++;
    data2 = *p;
    p++;
    if(channel_set & 0x1) //发送通道1数据
    {
        while (!(ALTERA_AVALON_UART_STATUS_TRDY_MSK
                    & IORD_ALTERA_AVALON_UART_STATUS(UART_0_BASE)))
            ;

        IOWR_ALTERA_AVALON_UART_TXDATA(UART_0_BASE, data2);

        while (!(ALTERA_AVALON_UART_STATUS_TRDY_MSK
                    & IORD_ALTERA_AVALON_UART_STATUS(UART_0_BASE)))
            ;

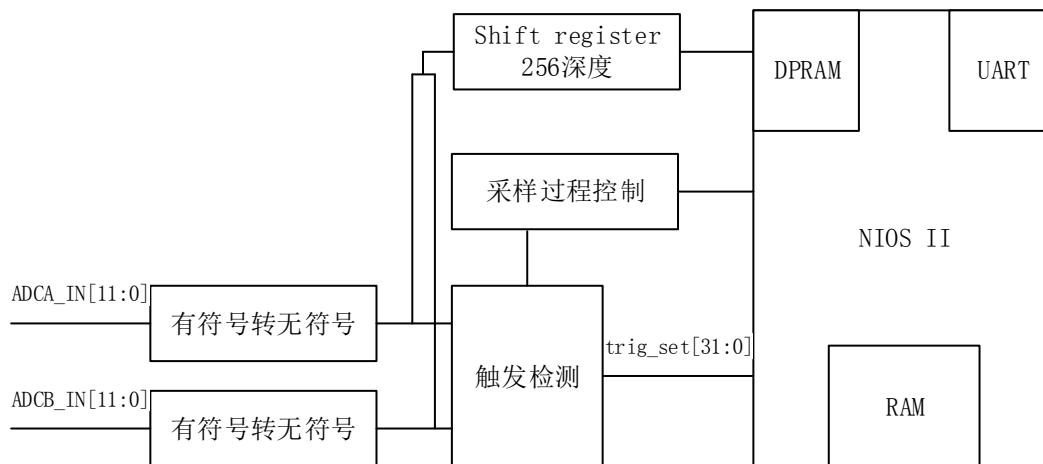
        IOWR_ALTERA_AVALON_UART_TXDATA(UART_0_BASE, data1);
    }
    data1 = *p;
    p++;
    data2 = *p;
    p++;
    if(channel_set & 0x2) //发送通道2数据
    {
        while (!(ALTERA_AVALON_UART_STATUS_TRDY_MSK
                    & IORD_ALTERA_AVALON_UART_STATUS(UART_0_BASE)))
            ;

        IOWR_ALTERA_AVALON_UART_TXDATA(UART_0_BASE, data2);

        while (!(ALTERA_AVALON_UART_STATUS_TRDY_MSK
                    & IORD_ALTERA_AVALON_UART_STATUS(UART_0_BASE)))
            ;

        IOWR_ALTERA_AVALON_UART_TXDATA(UART_0_BASE, data1);
    }
}
```

```
        i=i+4;
    }
    usleep(2000000);
}
return 0;
}
```



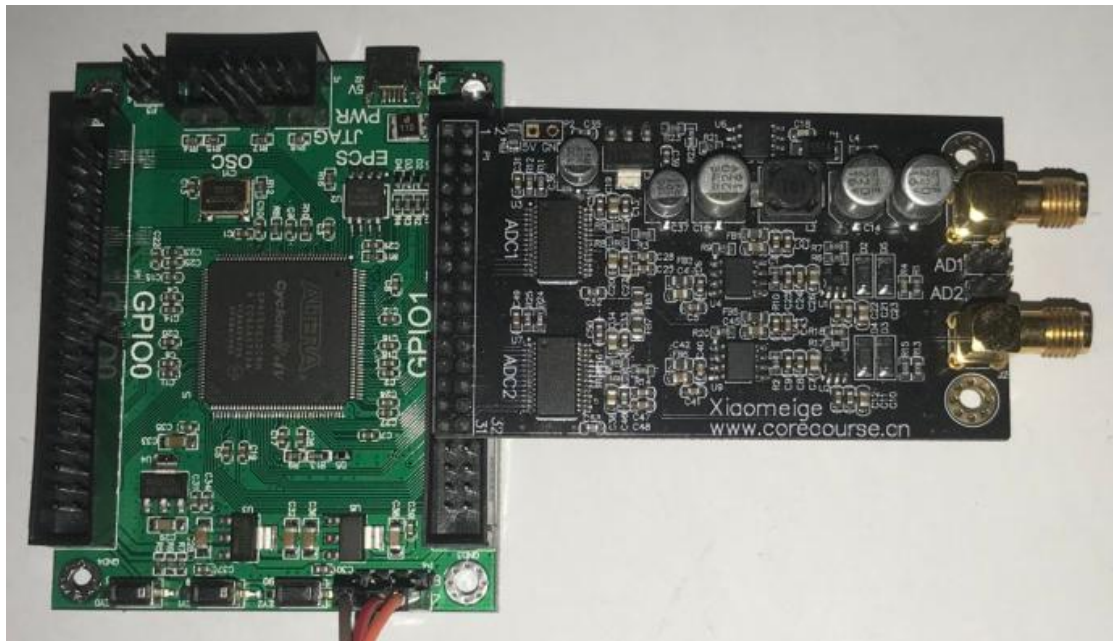
在 FPGA 中，首先将 ADC 的采样数据从有符号数转换为无符号数，然后将转换得到的无符号数提供给触发检测模块，根据 trig_set 设置的触发模式检测是否满足触发条件。另一方面，转换为无符号的数据被送入一个 256 深度的 shift register 中，当触发检测模块检测到触发条件时，就将 shift register 模块的输出依次写入 4096*32 位的双口 RAM 中。由于 shift register 出口的数据相较于入口部分，刚好延迟了 256 个时钟周期，所以，当触发条件产生时，shift register 出口处的数据恰好就是触发点前第 256 个数据，因此从此时开始，shift register 出口输出的连续 4096 个数据就恰好是触发点前 256 个数据加触发点加触发点后 3839 个数据。这样就便于分析触发点前后的数据了。

本工程虽然使用了 NIOS II 作为控制系统，但是日常使用时，如果不需要修改 NIOS II 程序，则仅需下载 sof 文件即可。NIOS II 的软件代码已经生成了 hex 格式的数据，并作为片上程序 RAM 的初始化数据，Quartus 在编译时就

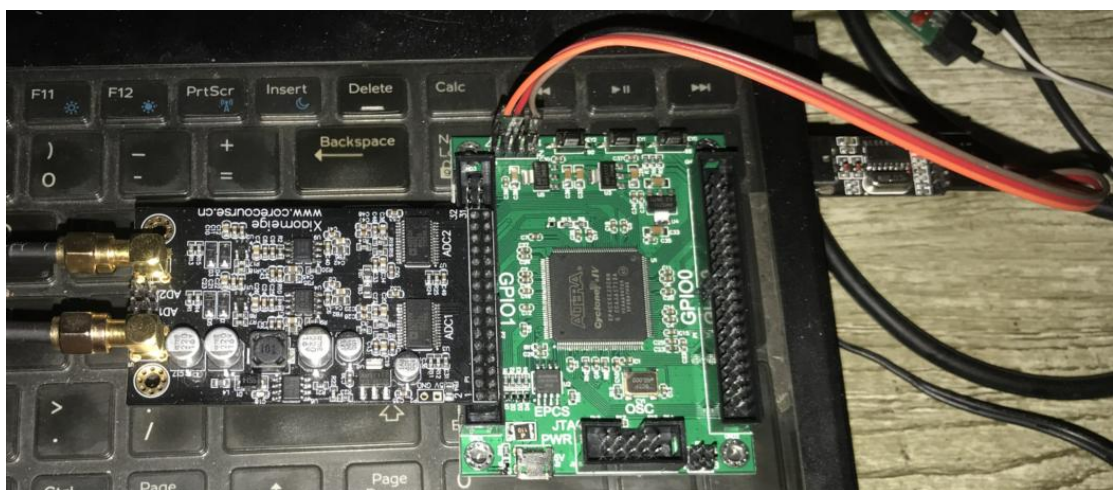
会直接将程序编译进 sof 文件中，因此下载的 sof 文件就已经是包含了 NIOS II 软件代码和 fpga 逻辑代码的固件。

以下以 AC606 FPGA 核心板和 ACM9226 高速 ADC 模块为例，讲解实验过程。
使用说明：

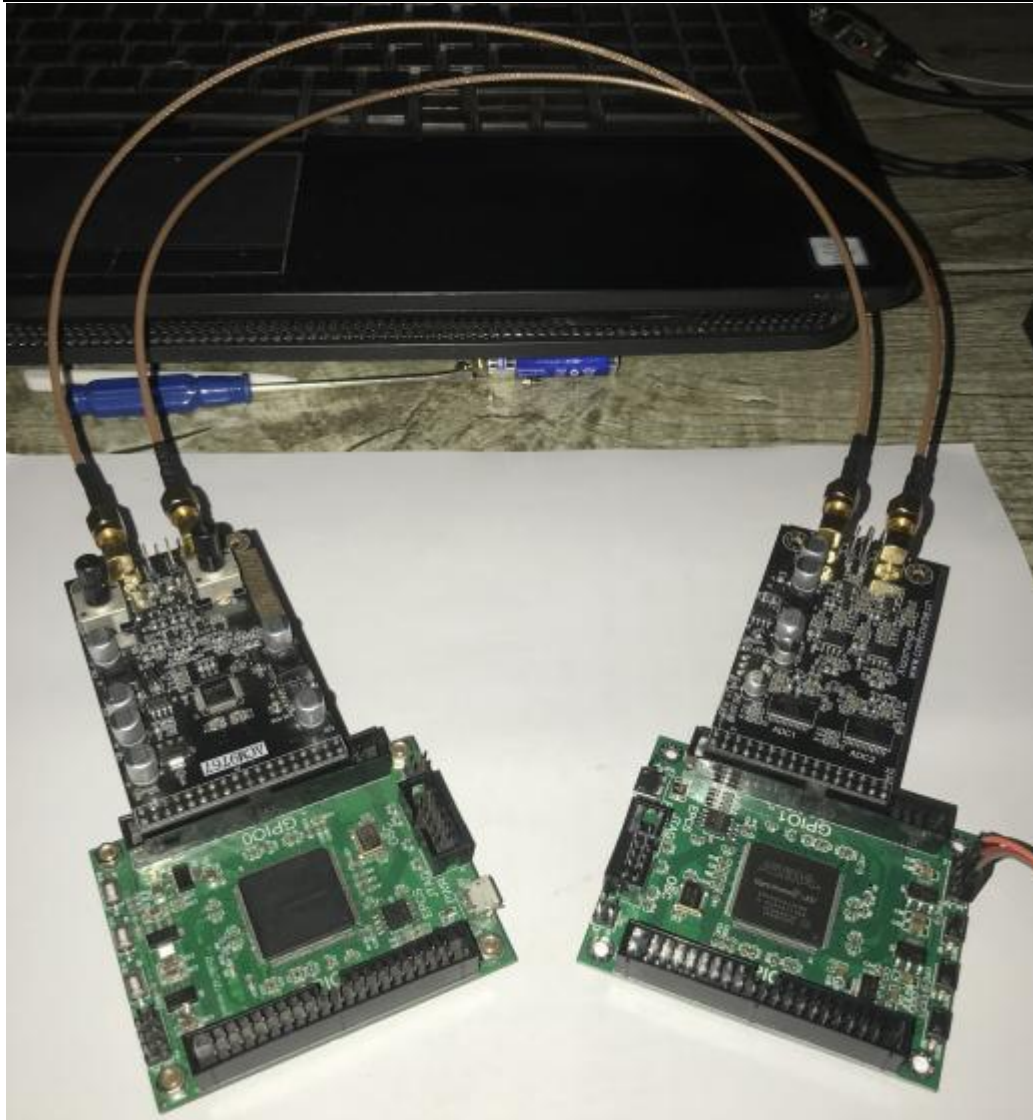
1、连接 ACM9226 模块到 AC606 核心板的 GPIO1 接口上，如下图所示：



2、使用一个 USB 转串口模块（CH340、PL2303、FT232 模块）接到 AC606 核心板上，本工程中是将引脚分配在了 AC606 按键右侧的排针上，如下图所示。



3、给 ADC 模块连接上信号源，用户需要自备信号发生器。我们在实验时使用的是 ACM9767+AC606 核心板搭建的一个双通道信号发生器。如下图所示。

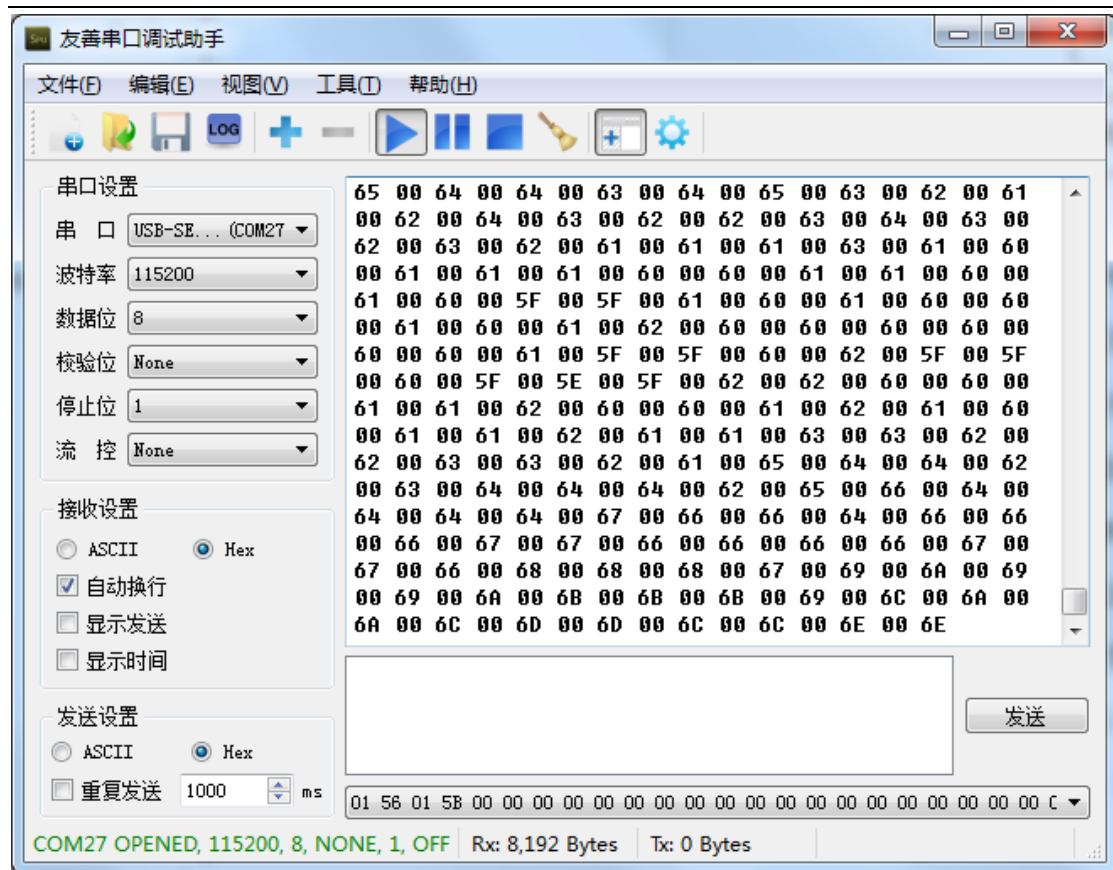


实际上，信号发生器和数据采集器是可以合并到一个工程中，在一个 AC606 核心板上同时连接 ACM9767 和 ACM9226 模块来完成实验的，我这里为了发布的工程文件的纯净，就没有做合并，而是使用了两套独立的系统。

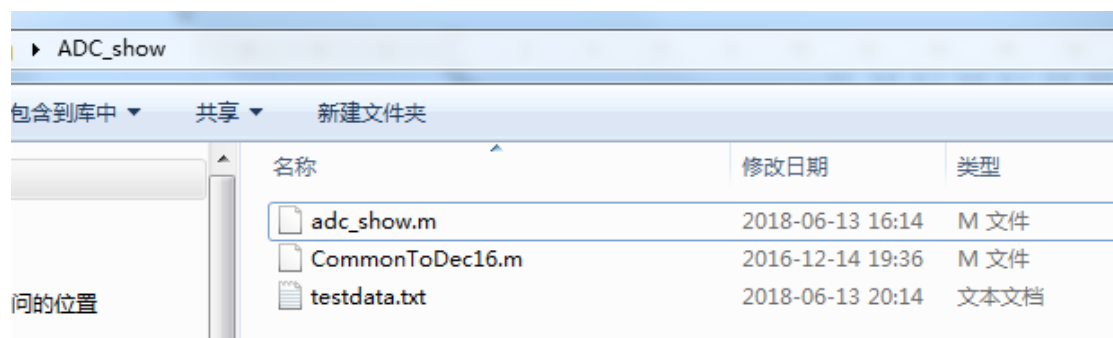
4、下载工程的 sof 文件。

5、调整信号发生器系统的输出频率，使信号频率在 50KHz

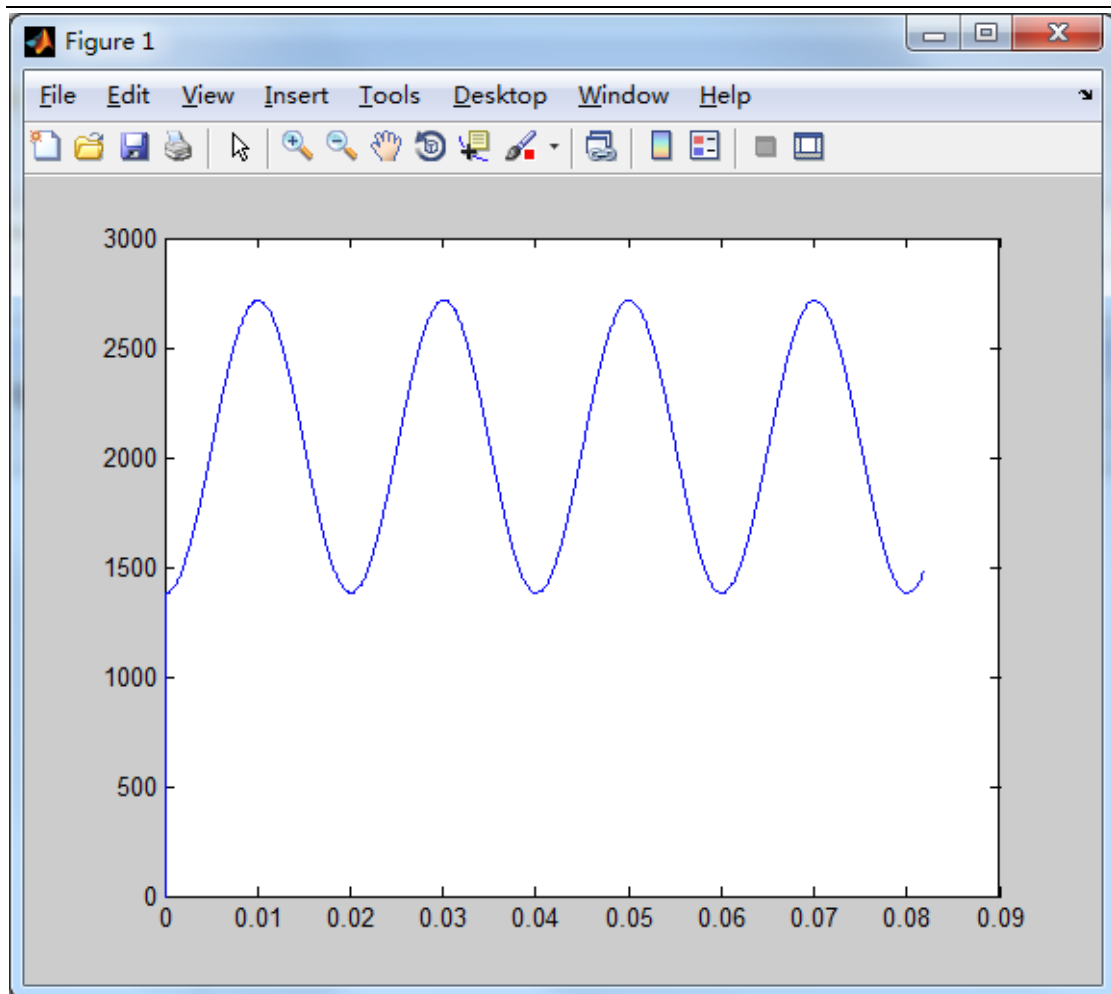
6、使用友善串口助手接收一次 8192 个字节的数据，然后停止接收（关闭串口），将接收窗口中的数据复制并保存在名为 testdata.txt 的文件中。



7、将 testdata.txt 文件拷贝到 ADC_Show 文件夹下替换已有的 testdata.txt 文件。



8、打开 MATLAB 软件，打开 ADC_Show 文件夹下的 ADC_Show.m 文件并运行，就可以看到绘制的波形数据了，如下图所示：



默认是仅发送通道 0 的数据的，因此可以看到 LED0 和 LED2 点亮，串口每隔 2 秒左右会收到 8192 个字节的数据。这里设置为两秒仅为方便手工测试时能够明确的区分一组 8192 个字节的数据。实际运行时，可以调整 NIOS 软件代码中的延时值，比如调整为 20ms。这样波形刷新就很快了。

下面附上 Matlab 的源码，本脚本文件由网友“冰三点水”提供，在此表示感谢。欢迎网友在此基础上优化，比如加入能够自动读取串口数据并实时显示波形的功能。

第一个文件，数据处理文件：CommonToDec16.m

```
%将十六进制转换为十进制（读入的数据格式为 高 8 位 低 8 位）
function ndata=CommonToDec2(filename)
s = filename;
cdata = textread(s,'%2c','delimiter',' ');
[m,n] = size(cdata);
ndata=zeros(1,m/2);
```



```
for i=0:m/2-1;
    k = 2*i + 1 ; %转换高 8 位
    if(cdata(k,1)>='0') && (cdata(k,1)<='9')
        mdata=cdata(k,1)-48;
        ndata(i+1)=ndata(i+1)+mdata*16^3;
    else
        mdata=cdata(k,1)-55;
        ndata(i+1)=ndata(i+1)+mdata*16^3;
    end

    if(cdata(k,2)>='0') && (cdata(k,2)<='9')
        mdata=cdata(k,2)-48;
        ndata(i+1)=ndata(i+1)+mdata*16^2;
    else
        mdata=cdata(k,2)-55;
        ndata(i+1)=ndata(i+1)+mdata*16^2;
    end

    k = 2*i + 2 ; %转换低 8 位
    if(cdata(k,1)>='0') && (cdata(k,1)<='9')
        mdata=cdata(k,1)-48;
        ndata(i+1)=ndata(i+1)+mdata*16^1;
    else
        mdata=cdata(k,1)-55;
        ndata(i+1)=ndata(i+1)+mdata*16^1;
    end

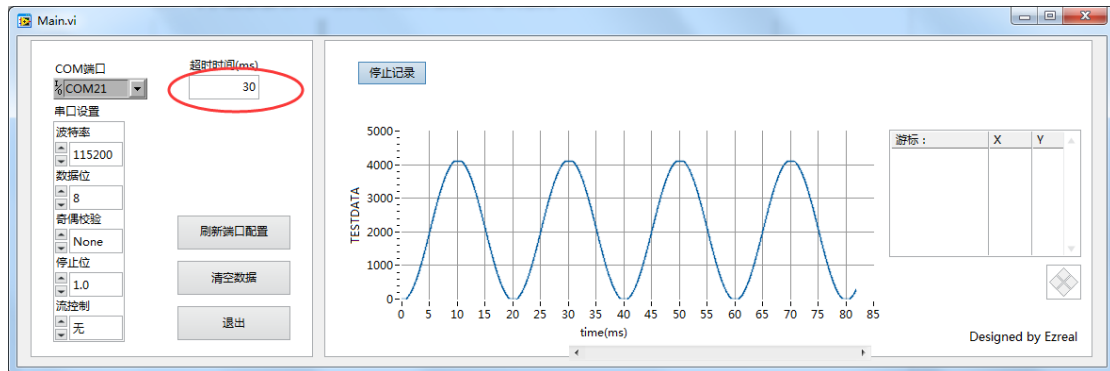
    if(cdata(k,2)>='0') && (cdata(k,2)<='9')
        mdata=cdata(k,2)-48;
        ndata(i+1)=ndata(i+1)+mdata*16^0;
    else
        mdata=cdata(k,2)-55;
        ndata(i+1)=ndata(i+1)+mdata*16^0;
    end
end
```

第二个文件，主文件：adc_show.m

```
%E 6d v1.0
clc
clear
close all
```

```
data_o=CommonToDec16('testdata.txt');  
N = 4096;  
t_s = 0.00002; %20ms  
t = 0:t_s:(N-1)*t_s;  
plot(t,data_o);
```

下图为网友使用 LabView 设计的一个实时接收串口数据并绘制波形的软件，可以直接接收串口发送的 ADC 数据并绘制波形。由于该软件能够自动接收数据并刷新，因此在 NIOS II 软件代码中将两帧间的时间间隔调整为了 20ms，以此提高刷新率。使用时，先选中 COM 端口，然后设置超时时间为 30ms，然后点击刷新端口配置。然后点击开始记录，就能开始刷新波形了。



由于此软件使用 LABVIEW 开发，这个应用的安装包太大，足足有 200 多兆，主要是包含了 LABVIEW 的运行环境。软件传到百度云盘，然后将链接和资料一起放在了 www.corecourse.cn 上。在网站上搜索“【产品资料】ACM9226”就能够找到了。