基于千兆以太网的 ad7606 的多通道数据采集传输系统

文档所涉及文件说明：

本次实验配套的文件压缩包名为，解压后可得到配套的工程文件，采样数据和调试工具等，具体如下：

- ACX720_AD7606_ETH：基于 vivado 的 AD7606 多通道数据采集系统（GMII）源工程
- ADCdata_to_wave_v1_1.m：基于 MATLAB 的采样结果数据处理函数
- 网络调试助手（NetAssist）：网口调试工具
- Wireshark 网络分析器：抓包工具

本案例所使用软硬件均可通过小梅哥 fpga 官方店铺购买获取，https://xiaomeige.taobao.com

概述

本案例基于 Xilinx Artix-7 系列 FPGA，结合 ADI 公司的 16 位 8 通道并行采样 ADC 芯片，并利用 ACX720/ACX735 开发板上一片 Realtek 的 RTL8211 以太网收发器，实现了对 AD7606 型 8 通道 16 位 ADC 的数据转换控制并输出。通过 RTL8211 提供的 GMII 接口接收网口下发的数据并转化成控制命令对 AD7606 的采样频率、数据采样个数以及采样通道进行合理配置，采集完成后的数据经 FIFO 缓存后，通过网口传输到电脑。用户可以在电脑上通过网口调试工具进行指令的下发以及对采样到的数据进行查看，可将输出的数据导出，进行进一步的数据处理分析。

系统原理

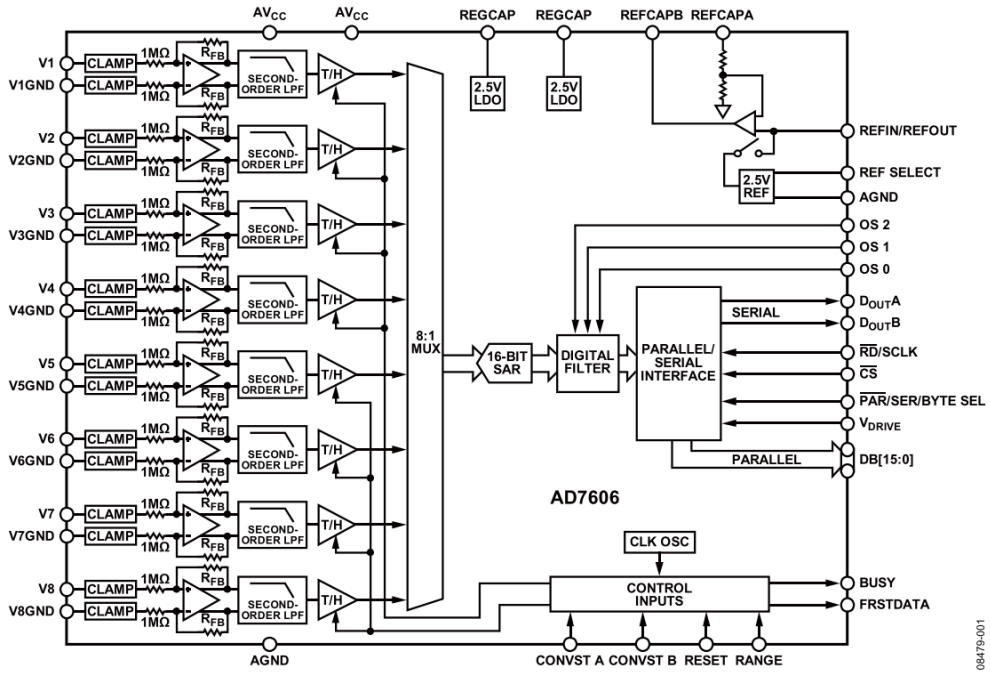
ad7606 概述

本案例使用的是 ADI 公司的 16 位 8 通道同步采样模数转换器 AD7606，AD7606 是 16 位 8 通道同步采样模数数据采集系统（DAS）。内置模拟输入箝位保护、二阶抗混叠滤波器、

跟踪保持放大器、16 位电荷再分配逐次逼近型模数转换器（ADC）、灵活的数字滤波器、2.5V 基准电压源、基准电压缓冲以及高速串行和并行接口。AD7606 采用 5V 单电源供电，可以处理±10V 和±5V 真双极性输入信号，同时所有通道均能以高达 200kSPS 的吞吐速率采样。输入箝位保护电路可以耐受最高达±16.5V 的电压。无论以何种采样频率工作，其模拟输入阻抗均为 1MΩ。采用单电源工作方式，具有片内滤波和高输入阻抗，因此无需驱动运算放大器和外部双极性电源。AD7606 抗混叠滤波器的 3dB 截止频率为 22kHz；当采样频率为 200kSPS 时，它具有 40dB 抗混叠抑制特性。

芯片对外提供 spi 和并行的数字接口。当 AD7606 的 8 个通道全部以 200KSPS 的最高速率进行转换时，数据输出速率达到了 25.6Mbps，需要使用高性能 MCU 的 SPI 外设才能勉强满足该速率要求。因此可以使用 16 位并口来进行数据的传输，提高数据传输速率。当 ad7606 应用在 FPGA 系统中的时候，使用 SPI 串行接口和并行接口都能够轻松的满足数据传输的速率需求。当在 FPGA 系统上应用 AD7606 时，可以通过在 FPGA 上设计 ad7606 的转换控制逻辑，将转换结果数据直接存储到片上的存储器如 fifo 或者 RAM 中，也可以存储到 FPGA 片外的存储器如 SRAM 或 SDRAM 中， 然后由其他主控芯片如 MCU 或 DSP 读出，或者直接在 FPGA 内部进行数据的运算和处理。当然，由于 FPGA 片上可以设计软核控制器，也可以直接使用软和控制器完成数据的处理和传输工作，如典型的，在 Intel FPGA 器件上使用 NIOS II 软核控制器，将 AD7606 的转换结果通过串口或以太网传输到电脑上，也可以直接将数据显示在液晶显示屏上。

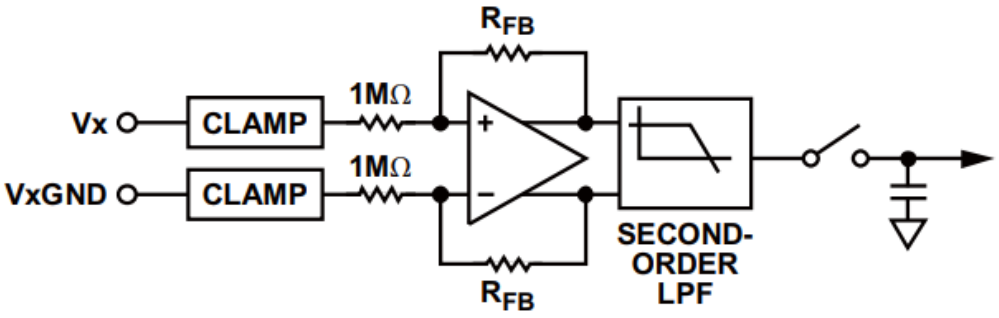
功能框图



AD7606 芯片的功能框图如上图所示，采集到的数据在经过稳压滤波和采样保持后通过 8 选 1 多路选择器被分别送入到 16 位逐次逼近型 ADC 芯片 AD7606 中进行转换，最后经由数字滤波后输出，当数据是以串行模式输出时，数据会从 DoutA、DoutB 中输出，如果数据是以并行方式输出，那么数据将从 DB[15:0]中输出，下面为 AD7606 部分模块的说明介绍。

模拟输入

AD7606 可处理真双极性、单端输入电压。RANGE 引脚的逻辑电平决定所有模拟输入通道的模拟输入范围。如果此引脚与逻辑高电平相连，则所有通道的模拟输入范围为±10 V。如果此引脚与逻辑低电平相连，则所有通道的模拟输入范围为± 5V。AD7606 的模拟输入阻抗为 1MΩ。这是固定输入阻抗，不随 AD7606 采样频率而变化。AD7606 的模拟输入结构如下图，其各路模拟输入均含有箝位保护电路。虽然采用 5V 单电源供电，但此模拟输入箝位保护允许输入过压达到± 16.5V。

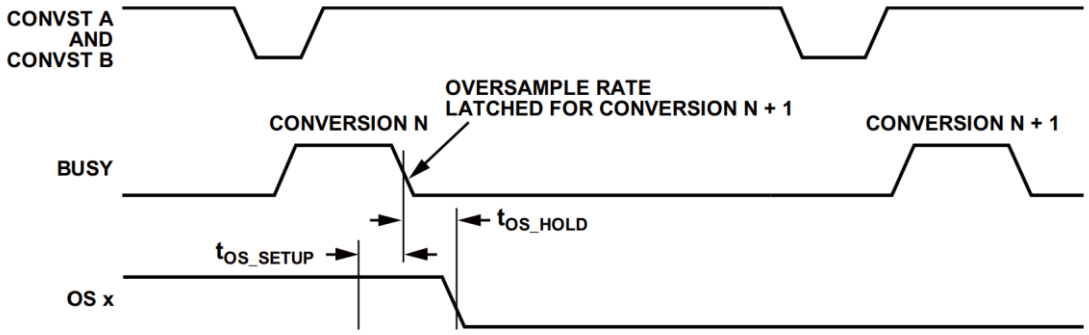


数字滤波器与过采样

AD7606 内置一个可选的数字一阶 sinc 滤波器，在使用较低吞吐率或需要更高信噪比或更宽动态范围的应用中，应使用该滤波器。数字滤波器的过采样倍率有过采样引脚 OS[2:0]控制（具体参考 AD7606 数据手册）。OS 2 为 MSB 控制位，OS 0 为 LSB 控制位，下表提供了用来选择不同过采样倍率的过采样位解码。

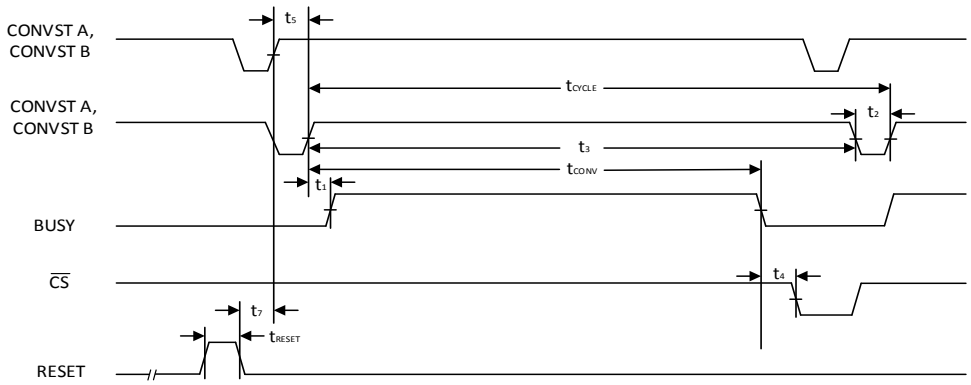
OS[2:0]	过采样倍率	5 V范围SNR(dB)	10 V范围SNR(dB)	5 V范围3 dB带宽(kHz)	10 V范围3 dB带宽(kHz)	最大吞吐量CONVST频率(kHz)
000	No OS	89	90	15	22	200
001	2	91.2	92	15	22	100
010	4	92.6	93.6	13.7	18.5	50
011	8	94.2	95	10.3	11.9	25
100	16	95.5	96	6	6	12.5
101	32	96.4	96.7	3	3	6.25
110	64	96.9	97	1.5	1.5	3.125
111	无效					

OS 引脚在 BUSY 下降沿锁存，从而设置下一个转换的过采样倍率（如下图所示），如果 OS 引脚选择过采样倍率 8，则下一个 CONVST x 上升沿采集各通道的第一个样点，一个内部产生的采样信号采集所有通道的其余 7 个样点，然后对这些样点求平均值，以改进 SNR 性能。开启过采样时，CONVST A 和 CONVST B 引脚必须连在一起驱动，转换过程中 BUSY 保持高电平的时间会延长。BUSY 保持高电平的实际时间取决于所选的过采样倍率，过采样倍率越高，则 BUSY 保持高电平的时间或总转换时间越长。

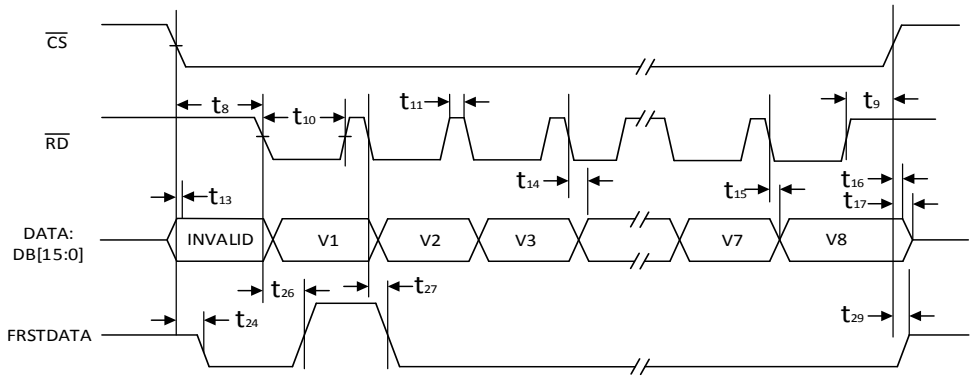


工作时序

AD7606 根据采样方式不同具有多种驱动时序，本次采用的为并行输出（即 8 个 16 位的数据通过 16 根并行线一个接着一个输出），转化后读取模式。其时序图有两部分组成：1.完成 AD 转化；2.读取 AD 数据。其中的时间可以参考 ADI 公司的手册。当 CONVST A 和 CONVST B 通道都变为上升沿时，BUSY 信号转变为高电平，代表转换开始，直到 BUSY 的下降沿到来，代表数据已经转换完成，正在锁存至输出数据寄存器中，当 $\overline{CS}$ 变为下降沿时，数据将会被输送到总线上。并行工作模式下，当 $\overline{CS}$ 和 $\overline{RD}$ 都为低电平时，会使能总线，将转换结果输出到并行数据总线上，当 V1 转换结果开始输出之后，FRSTDATA 会随后转变为高电平，表示输出数据总线可以提供 V1 的结果。并行模式下，每次数据的输出为 16 位，对应一个通道。



CONVST 时序——转换之后读取



并行模式，独立的 $\overline{CS}$ 和 $\overline{RD}$ 脉冲

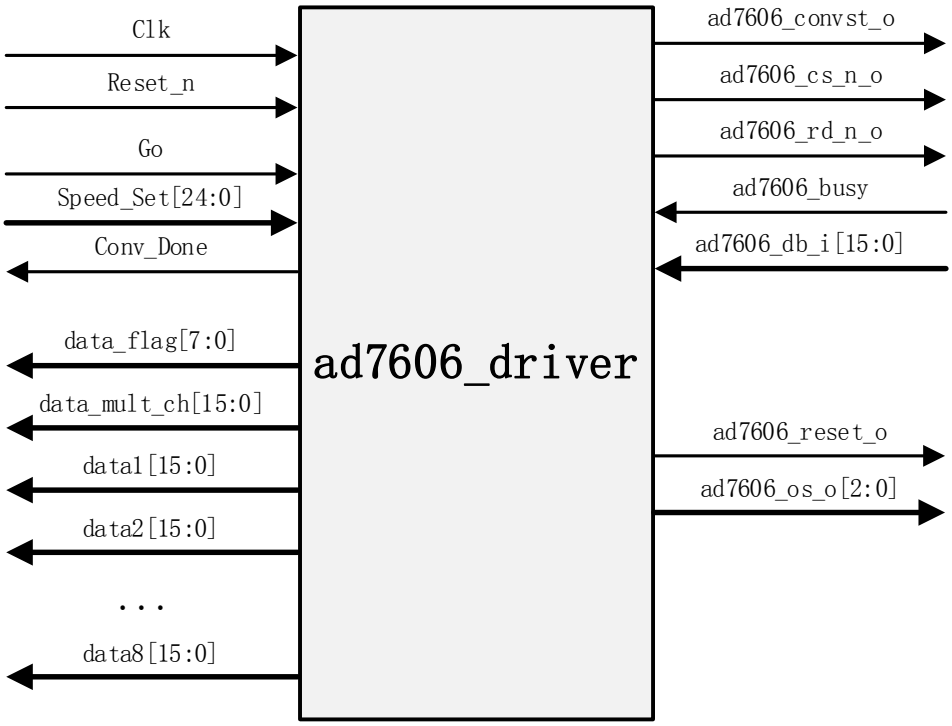
控制器说明

功能简介

该控制器实现了对 AD7606 型 8 通道 16 位 ADC 的数据转换控制并输出。使用该控制器时，用户无需关心 AD7606 的具体控制时序，一切都在控制器内部完成，用户只需要像使用并行 ADC 一样取用数据即可。我们还为该模块提供了基于 quartus II 的数据采集方案,有关基于 quartus II 的数据采集方案与本次设计的基于 vivado 的数据采集系统原理一致，只是应用的平台不同，用户可以参考我们提供的相关源工程。

接口说明

该控制器接口分为四个类，第一类为时序逻辑模工作所必须的时钟和复位信号（Clk、Reset_n），第二类是 AD7606 控制器与 AD7606 芯片管脚相连的各种功能控制和数据信号，第三类是设置控制器工作状态/工作数据的用户控制接口，第四类是控制器的结果输出接口。对于用户来说，只需要关注第三类和第四类接口的使用，即可快速高效的使用该控制器来控制 AD7606 芯片完成数据转换。

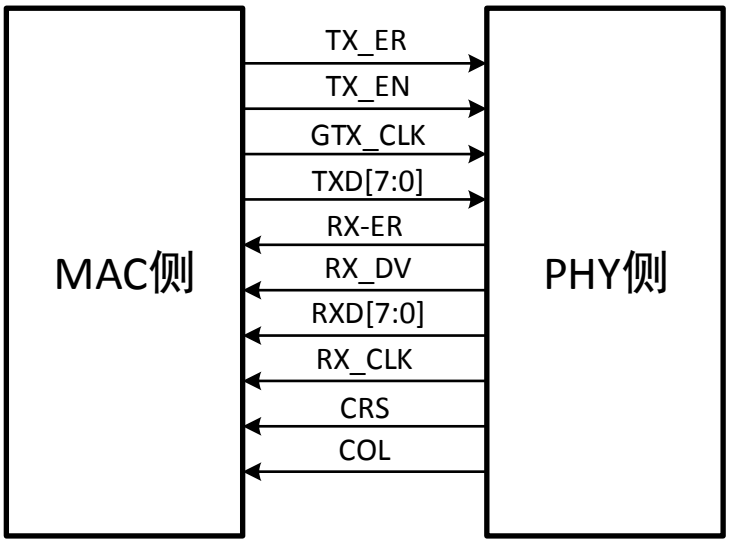


类	信号名	位	功能简介
	Clk	1	系统时钟，为了让采样速率准确，要求为 50MHz
	Reset_n	1	系统复位，低电平复位
控制信号	Go	1	采样使能信号，为高电平就使能采样，低电平则在已经开始的一轮采样结束后，停止下一次采样。
	Speed_Set	25	采样速率控制端口，Speed_Set = 1000000000/20/speed - 1
	Conv_Done	1	一次采样完成标志信号，单时钟周期脉冲信号。每次 8 个通道结果都输出后，产生一个高脉冲信号。
数据结果输	data_flag	8	转换结果有效标志信号，因为 AD7606 有 8 个通道，转换结果是依次输出，并非同时的，所以设置 8 个 Flag 信号，每个通道的结果就绪之后，就产生一个 Flag 信号，通知外部可以取用。另外，如果只关心其中的部分通道，则只需要关心 data_flag 中对应的位即可。

出端口和标志信号	data_mult_ch	16	多通道数据输出端口，该通道 16 位，在不同的时刻，输出不同通道的转换结果，使用时，与 data_flag 信号配合，data_flag 的哪一位出现高脉冲，则代表当前 data_mult_ch 的值为该通道的转换结果。该端口设计的目的是用于往 FIFO、RAM 等存储器中存储结果时使用。
	data1..8	16	8 个通道的采样结果输出端口，每个端口分别对应一个模拟通道的采样结果，使用时与 data_flag 信号配合，每当 data_flag 中的某一位为 1 时，则对应的通道上的 16 位采样结果已经就绪，可以使用。这些端口主要用于每个通道的数据需要分别应用的场合。
ADC 芯片控制信号	ad7606_cs_n_o	1	ad7606 芯片选中控制信号，可以从 AD7606 中读取转换结果时，需要使该信号为低电平。
	ad7606_rd_n_o	1	ad7606 转换结果读取信号，该信号的下降沿， AD7606 将特定通道的采样结果送到 16 位数据线上，供外部读取。外部可以在 rd_n 信号的上升沿读取 16 位数据线上的结果。
	ad7606_busy_i	1	ad7606 转换状态标志信号，为高电平则表明 ad7606 当前仍处于转换状态，结果没有更新，如果此时读取，读取的结果就还是前一次的采样转换结果。需要待该信号变为低电平之后，再读取 ad7606 中的数据。
	ad7606_db_i	16	ad7606 的 16 位数据线，读取时，输出对应通道的转换结果。
	ad7606_os_o	3	ad7606 过采样控制信号，使用过采样可以进一步提高 ad7606 的采样精度，使用过采样会降低 ad7606 的有效转换速率，关于过采样的功能和使用方法，可以参考 ad7606 的 datasheet，默认为 0，则表示不使用过采样。能够运行在最高的转换速率（200Ksps）
	ad7606_reset_o	1	ad7606 的复位信号，复位 ad7606 内部各个功能单元的工作状态。
	ad7606_convst_o	1	ad7606 转换开始信号，该信号的上升沿启动 ad7606 内部的采样转换逻辑开始新一轮的采样转换。

GMII 接口

GMII 接口信号连接关系及各信号的介绍如下。



信号	方向	位宽	说明
TX_ER	O	1	即 Transmit Error，发送数据错误提示信号，同步于 GTX_CLK，高电平有效，表示 TX_ER 有效期内传输的数据无效。
TX_EN	O	1	即 Transmit Enable，发送使能信号，只有在 TX_EN 有效期内传的数据才有效。
GTX_CLK	O	1	发送参考时钟，时钟频率为 125MHz。需要注意的是，GTX_CLK 时钟的方向是从 MAC 侧指向 PHY 侧的，此时钟是由 MAC 提供的，这里与 MII 接口有所差别的地方。
TXD	O	8	即 Transmit Data，数据发送信号，共 8 根信号线
RX_ER	I	1	即 Receive Error，接收数据错误提示信号，同步于 RX_CLK， 高电平有效，表示 RX_ER 有效期内传输的数据无效。
RX_DV	I	1	即 Reveive Data Valid，接收数据有效信号，作用类似于发送通道的 TX_EN。
RXD	I	8	即 ReceiveData，数据接收信号，共 8 根信号线。
RX_CLK	I	1	接收数据参考时钟，时钟频率为 125MHz。RX_CLK 是由

			PHY 侧提供的。
CRS	I	1	即 Carrier Sense，载波侦测信号，不需要同步于参考时钟，只要有数据传输，CRS 就有效。需要注意的是 CRS 只有 PHY 在半双工模式下有效。
COL	I	1	即 Collision Detectd，冲突检测信号，不需要同步于参考时钟。需要注意的是 CRS 只有 PHY 在半双工模式下有效。

（注：表格中的方向是站在 MAC 侧角度看的）

应用案例

系统模块功能介绍

AD7606 控制器驱动模块（ad7606_driver）

该控制器在工作时会根据主机的指令对采样频率进行修改，当信号转换完成后便对 ADC 写控制器发出 data_flag 信号，控制其将转换完成数据 data_mult_ch 写入 FIFO 或者 RAM 的同时，也指出了该信号来自哪个通道。通过这两个信号，我们可以实现让 ADC 以特定的采样速率多次采样一个或多个通道的数据。每当 data_flag 中任意一位为 1，则将 data_mult_ch 中的值写入 FIFO 或者 RAM 中。由于 FIFO 和 RAM 等存储器，只有一个数据输入接口，所以这个时候用一个 data_mult_ch 端口分时输出不同通道的采样结果，就比使用 data1~data8 这 8 个 16 位的数据端口分别输出各自通道的采样结果要方便。

例如， 将通道 1、2、5、8 的采样结果存入 FIFO。就可以使用下面的写法。

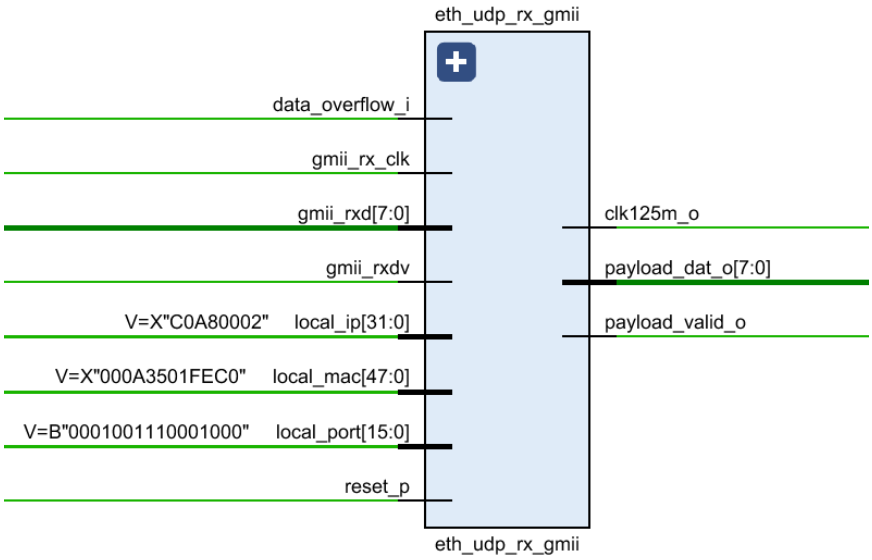
```
module adc_wr_fifo(
    input Clk,
    input Reset_n,
    input [7:0]data_flag;
    input [15:0]data_mult_ch;
    output reg fifo_wrreq,
    output reg [15:0]fifo_data
);
```

```
    always@(posedge Clk or negedge Reset_n)
    if(!Reset_n)begin
        fifo_wrreq <= 0;
        fifo_data <= 0;
    else if(data_flag == 8'b1001_0011)begin
        fifo_wrreq <= 1'd1;
        fifo_data <= data_mult_ch;
    end
    else begin
        fifo_wrreq <= 0;
        fifo_data <= fifo_data;
    end

endmodule
```

GMII 以太网接收模块（eth_udp_rx_gmii）

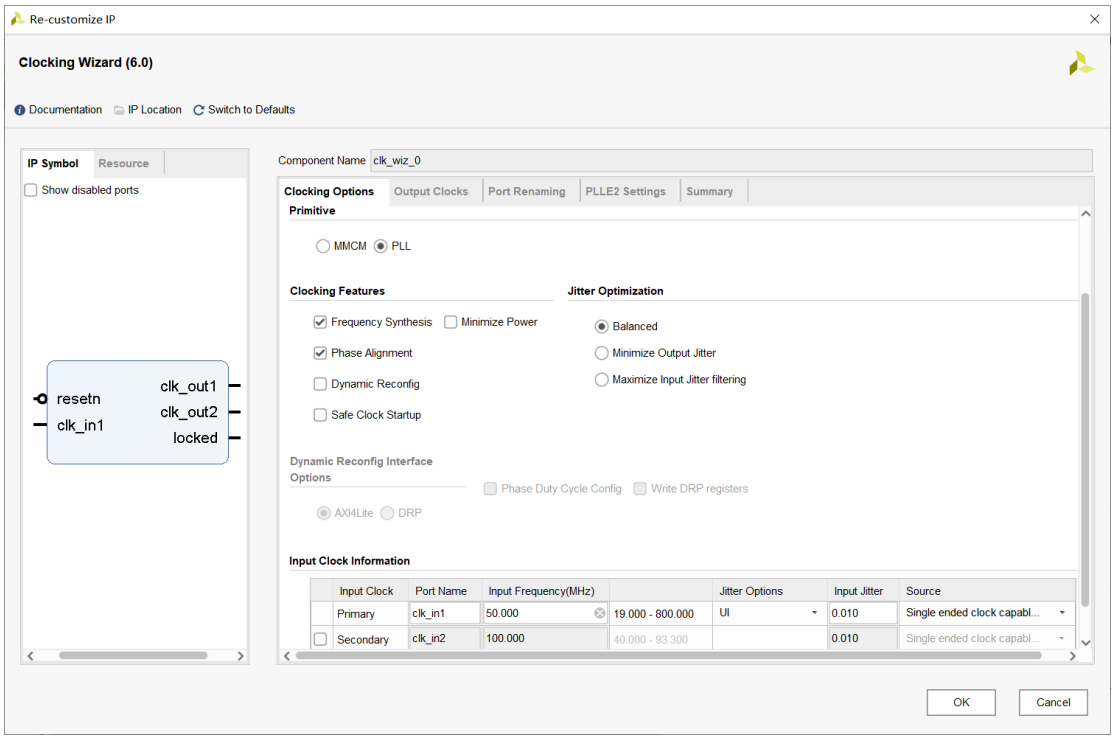
以太网接收模块可以接收用户在电脑上通过网络调试助手发送的包含指令数据的数据帧，并对接收的的数据帧进行判断解析。以太网接收模块的工作时钟 gmii_rx_clk 为 125MHz，由 PHY 芯片 RTL8211 提供，数据接收 gmii_rxd 信号为 8 位，每次接收一个字节，一秒接收的数据量为 1000M。gmii_rxdv 为接收数据有效信号，当 gmii_rxdv 有效 FPGA 接收数据。本地主机（PC）端为目的端，与之连接的 FPGA 端为源端，其中 local_mac、local_ip、local_port 分别为源 MAC 地址、源 IP 地址与源设备端口号，把设置的 MAC 地址转换为十进制表示为：00_10_53_01_254_192，源 IP 地址十进制为：192_168_00_02，源设备端口号十进制表示为：5000。以太网接收模块复位信号为高电平复位。

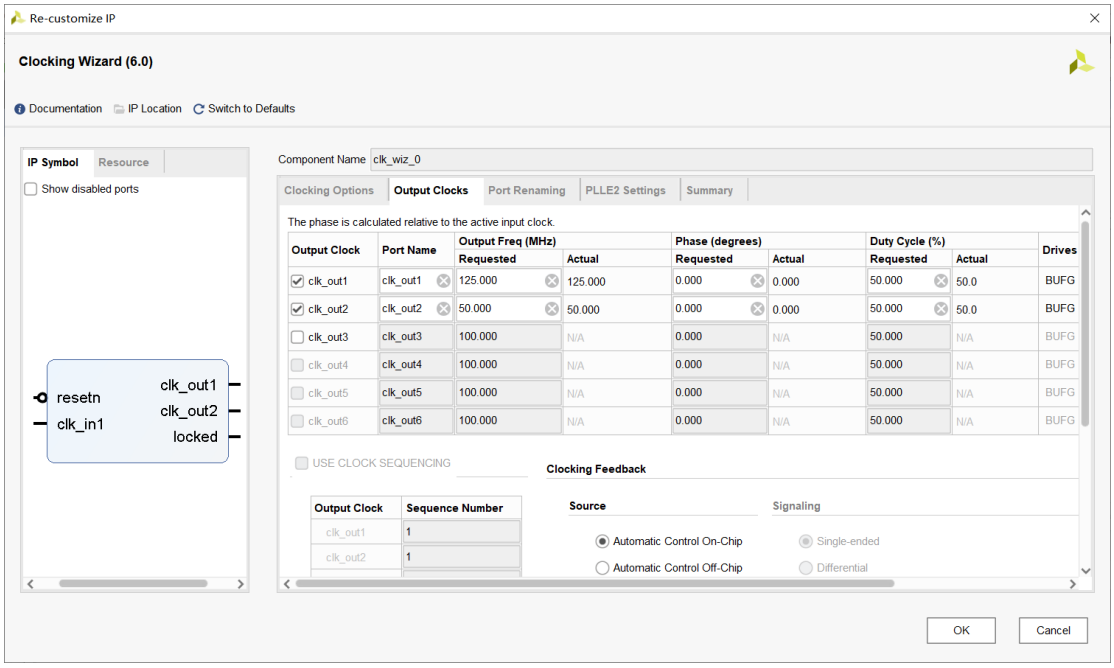
以太网接收模块对数据进行分析处理后可输出我们所需要的信号，根据本次实验需要，这里我们输出三个信号。Clk125m_0 作为时钟信号输出，payload_dat_o 则是从 PC 端接收的数据信息，后续需要对 payload_dat_o 输出的数据做进一步的处理得到所需要的控制指令数据。payload_valid_o 为 payload_dat_o 输出标志信号。想要进一步理解以太网功能原理可参考 ACX720 配套资料《小梅哥 Xilinx FPGA 自学教程》35 章。



时钟 IP 核 PLL(clk_wiz_0)

由于本次实验中所运用到的模块需要在不同的时钟下工作，例如千兆以太网的发送和接收模块的工作时钟为 125MHz，AD7606 控制器驱动模块工作时钟为 50MHz。本次实验运用时钟 IP 核，选择 PLL 模式。Clk_in1 输入系统时钟 50MHz，clk_out1、clk_out2 分别输出 125MHz 与 50MHz 时钟信号。locked 可作为复位信号使用，在 pll 输出时钟与输入时钟不处于锁定状态时 locked 为低电平，当锁定后 Locked 信号就变为高电平。下图为 PLL 的配置信息。

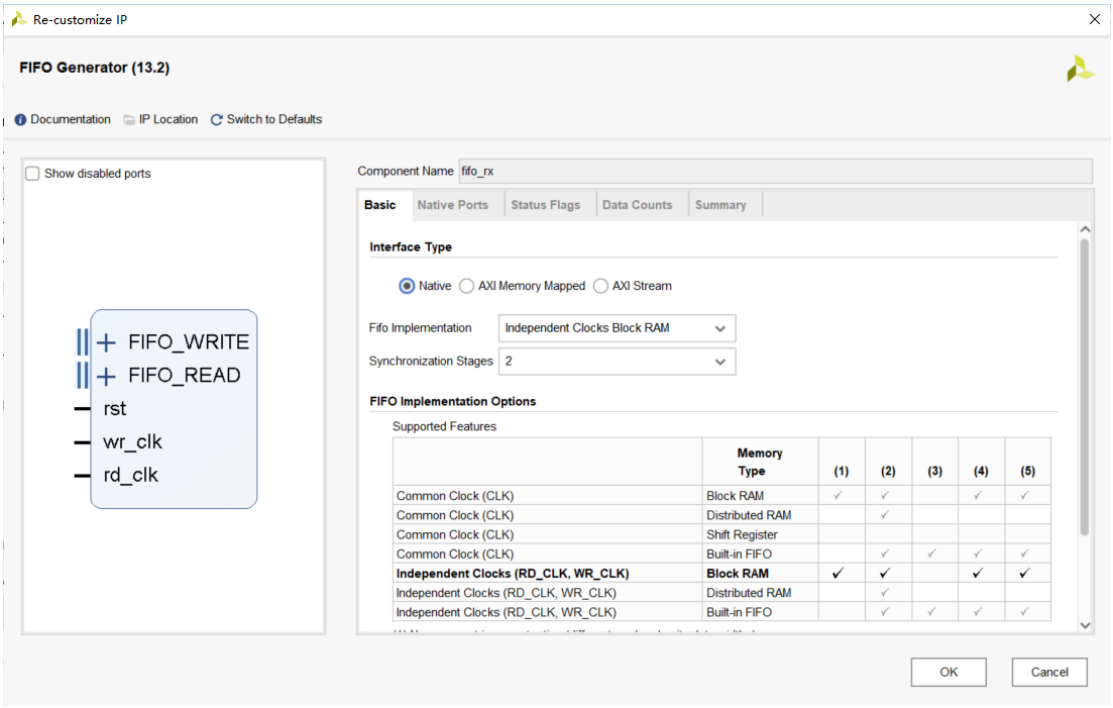


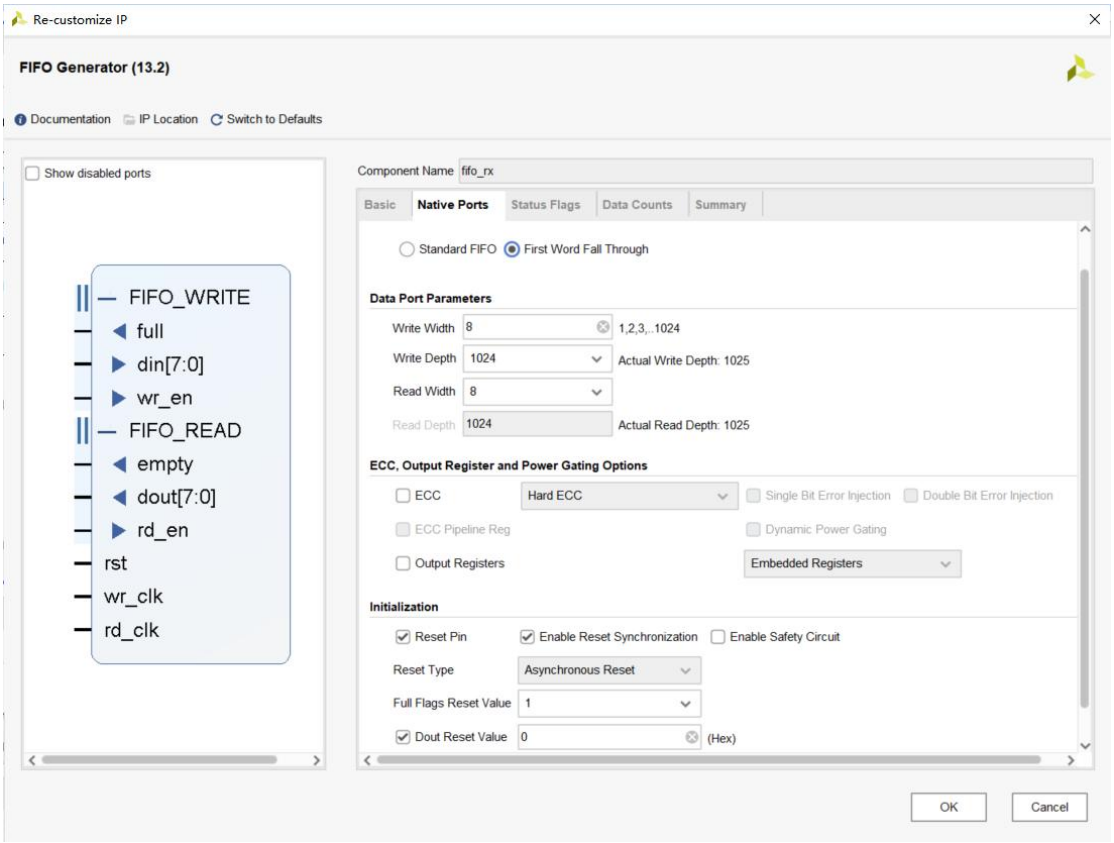


FIFO IP 核（fifo_rx）

FIFO（First In First Out），即先进先出存储器。千兆以太网接收模块工作时钟为 125MHz，而 AD7606 控制器驱动模块的工作时钟为 50MHz，不能使用 125MHz 的速率来直接采集 50MHz 速率的数据，因为两者数据速率不匹配，在采集过程中会出现包括亚稳态问题在内的一系列问题，所以这里使用一个具备双时钟结构的 FIFO 来进行异步数据的收发。

为实现数据的调用需要配置一个双时钟 FIFO，接口类型选择 Native,双时钟 FIFO 模式选择为 Independent Clock Block RAM。Read Mode 类型选择 First Word Fall Through 模式，使当前数据提前已经到数据读数据线上，在读使能到来后，下一个数据会到数据线上。这里 FIFO 读写位宽都设置的为 8 位，数据深度为 1024，其它设置保持默认。配置信息如下图所示。





wr_clk 连接 clk_out1 输入 125MHz 时钟信号，wr_en、full 与 din 信号同步于 wr_clk，wr_en 由有效时使 FIFO 通过 din 开始往里面写数据。rd_clk 输入 50MHz 时钟信号，同样的，empty、dout、rd_en 同步于 rd_clk，当 rd_en 有效时外部从 FIFO 读数据。

接收转命令模块（eth_cmd）

在讲解该模块之前，需要先对配置 AD7606 的寄存器进行说明，配置 AD7606 的寄存器一共有四个，他们的功能和地址分别如下：

名称	地址	位宽	功能简介
RestartReq	0	1	重新开始采集请求寄存器，向该寄存器写入任意值即可启动新一轮的采样存储传输。
ChannelSel	1	8	通道设置寄存器，共 8 位，对应了 8 个通道的数据存储开关，如果某通道对应的设置位为 1，则该通道的采样结果就会被存入 FIFO 并通过串口发送。
DataNum	2	16	数据个数寄存器，设定总共采集传输多少个数据。注意，该寄存器设置的是总共采集的数据个数，假设设置采集 100 个数据，而且设置了 ChannelSel 为 0000_0011b，则实际每个通道采样的次数就是 50，2 个通道的数据加起来是 100 个。假设设置采集 100 个数据，而且设置了 ChannelSel 为 0011_0011b，则实际每个通道采样的次数就是 25。4 个通道加起来采集 100 个数据
ADC_Speed_Set	3	32	ADC 采样速率设置寄存器。该寄存器用来设置 ADC 每多久执行一次转换。由于 ADC 的最大采样速率为 200Ksps，所以可以通过设置该寄存器的值来让 ADC 的采样速率在 1~200Ksps 范围内调整，以适应不同的应用场景。该寄存器的值与采样速率关系为：ADC_Speed_Set = 1000000000/20/speed – 1，其中 speed 就是实际要设置的采样速率。

网口发送一次命令数据内容为 32 个字节，为了实现通过网口修改这些寄存器的值，需要对发送一次的数据进行拆解才能实现，对于设计的数据帧，该帧一帧数据共 8 个字节，包含帧头、帧尾、地址段、数据段。帧格式如下所示：

数据	D0	D1	D2	D3	D4	D5	D6	D7
功能	帧头 0	帧头 1	地址	data[31:24]	data[23:16]	data[15:8]	data[7:0]	帧尾
值	0x55	0xA5	xx	xx	xx	xx	xx	0xF0

每帧数据共 8 个字节，分别用 D0~D7 表示，其中，D0 和 D1 两个数据作为帧头，其值分别为固定的 0x55、0xA5，D7 作为帧尾，其值为固定的 0xF0。D2 则为要操作的寄存器地

址，D3 为要写入寄存器的数据的 24~31 位，D4 为要写入寄存器的数据的 16~24 位，D5 为要写入寄存器的数据的 8~15 位，D6 为要写入寄存器的数据的 0~7 位。

而该模块的作用就是将网口接收到的数据拆解成该格式，将 D2 作为地址 address 输出，指定修改哪个寄存器，D3~D6 共 32 位作为数据 data 输出，控制 AD7606 进行相对应的配置。该模块在执行过程中会对数据进行判定，当数据符合 D0 为 8'd55,D1 为 8'dA5,D7 为 8'dF0，则代表该数据格式正确，会生成一个指令正确信号 cmdvalid 输出到指令转控制模块。

```
    if(fifo_rd_req)begin
     rddata_cnt <= rddata_cnt + 1'b1;
     case(rddata_cnt)
         0,8,16,24   : begin data_0[63:56] = fifodout;data_0_done <= 1'b0;end
         1,9,17,25   : data_0[55:48] = fifodout;
         2,10,18,26  : data_0[47:40] = fifodout;
         3,11,19,27  : data_0[39:32] = fifodout;
         4,12,20,28  : data_0 [31:24]= fifodout;
         5,13,21,29  : data_0 [23:16]= fifodout;
         6,14,22,30  : data_0[15:8] = fifodout;
         7,15,23   : begin data_0[7:0] = fifodout;data_0_done <= 1'b1;end
         32'd31 : begin
                     data_0[7:0] = fifodout;
                     data_0_done <= 1'b1;
                     rx_done <= 1'b01;
                  end
         default:;
     endcase
 if(data_0_done)begin
     f((data_0[63:56] == 8'h55) && (data_0[55:48] == 8'ha5)
 && (data_0[7:0] == 8'hf0))begin
         cmd_data[31:24] <= #1 data_0[39:32];
         cmd_data[23:16] <= #1 data_0[31:24];
         cmd_data[15:8] <= #1 data_0[23:16];
         cmd_data[7:0] <= #1 data_0[15:8];
         address <= #1  data_0 [47:40];
         cmdvalid <= 1'b1;
      end
 end
 if(rx_done)
     state <= 2;
```

```
    end
    else begin
        fifo_rd_req <= 1'b0;
        rx_done <= 1'b0;
        rddata_cnt <= 16'd0;
        address <= #1 0;
        cmd_data <= #1 32'd0;
        data_0_done <= 1'b0;
end
```

指令转控制模块（ad7606_instruct）

该模块的作用旨在将从接收转命令模块接收到的数据转换为相应的控制数据并分别输出到对应的模块，每当接收到 cmdvalid 信号时，该模块便通过地址信号 address 判断对那个寄存器执行操作，并从数据中提取对应的数据将其输出到相应的寄存器。

```
    always@(posedge Clk or negedge Reset_n)
        if(!Reset_n)begin
            ChannelSel <= 8'b1111_1111;
            DataNum <= 16'd32;
            ADC_Speed_Set <= 32'd9999;
            RestartReq <= 1'b0;
        end
        else if(cmdvalid)begin
            case(cmd_addr)
                0: RestartReq <= 1'b1;
                1: ChannelSel <= cmd_data[7:0];
                2: DataNum <= cmd_data[15:0];
                3: ADC_Speed_Set <= cmd_data;
                4:
                    begin
```

```
                    ChannelSel <= cmd_data[7:0];
                    DataNum <= cmd_data[23:8];
                    RestartReq <= 1'b1;
                end
            default:;
        endcase
    end
    else
        RestartReq <= 1'b0;
```

ADC 采集控制模块（adc_Write_ctrl）

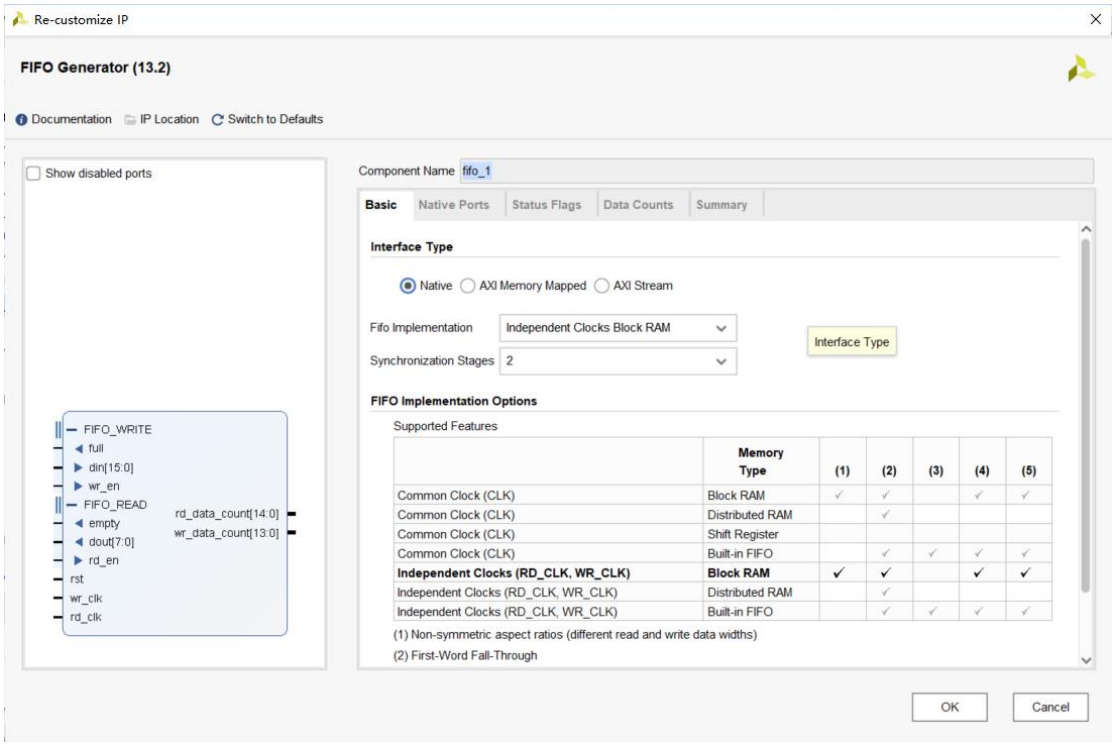
AD7606 在配置好采样频率后进行采样，采集的数据由 adc_data_mult_ch 输入 ADC 采集控制模块。同时，输入 adc_data_flag 采样标志信号，使不同通道采集数据有效时产生对应位的有效信号，ADC 采集控制模块根据 adc_data_flag & ChannelSel 有效输出 fifowrreq 信号用于读取所需通道的数据。采集的数据存入 FIFO，经由 FIFO 后通过以太网发送模块发送到 PC 端。AD7606 每次采集的数据为 16 位，两个字节，缓存到 FIFO 中时先存入高位数据。以太网发送模块数据接收信号共 8 根信号线，每次可从 FIFO 读取一字节数据，为保证以太网发送模块从 FIFO 读取数据的准确性，这里对写入 FIFO 的 fifowrdata 的值做高字节和低字节调换。

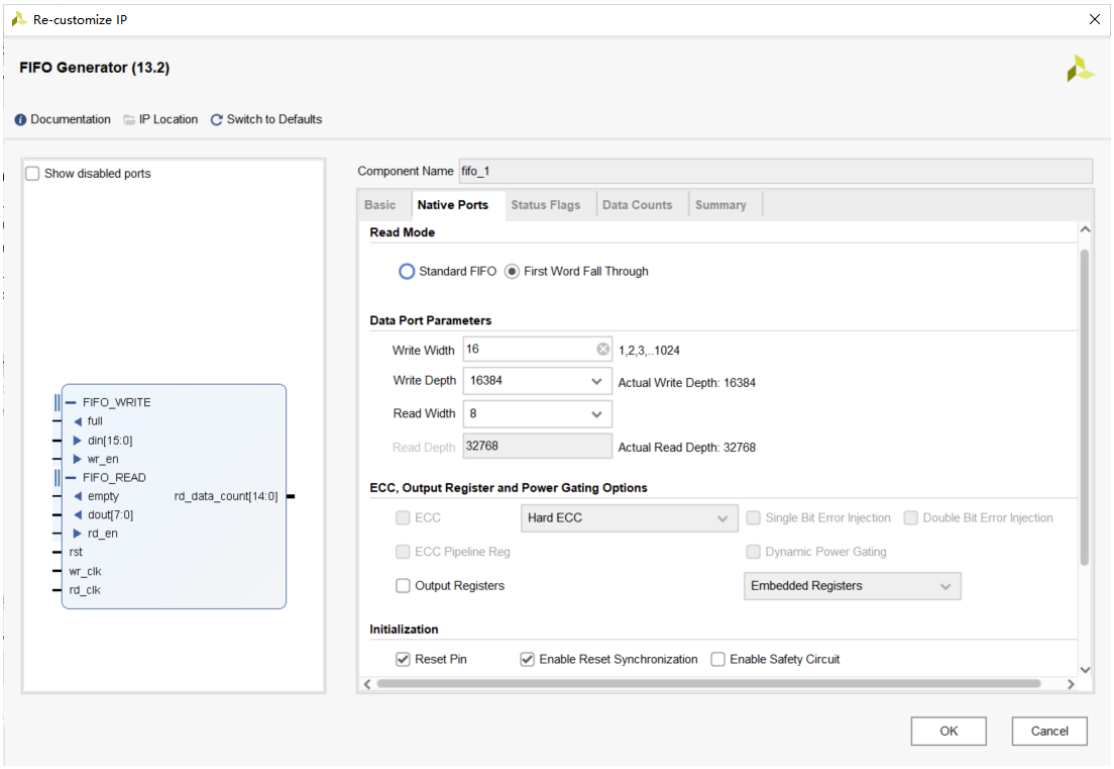
```
//采样个数计数器，在采样使能阶段，每个 flag 到来的时候计数器自加 1
    always@(posedge Clk or negedge Reset_n)
    if(!Reset_n)
        data_cnt <= #1  15'd0;
    else if(sample_en)begin
        if(adc_data_flag & ChannelSel)
            data_cnt <= #1  data_cnt + 1'd1;
        else
            data_cnt <= #1  data_cnt;
    end
    else
        data_cnt <= #1  15'd0;

    always@(posedge Clk)
        fifowrreq <= #1  (adc_data_flag & ChannelSel)  && sample_en;
```

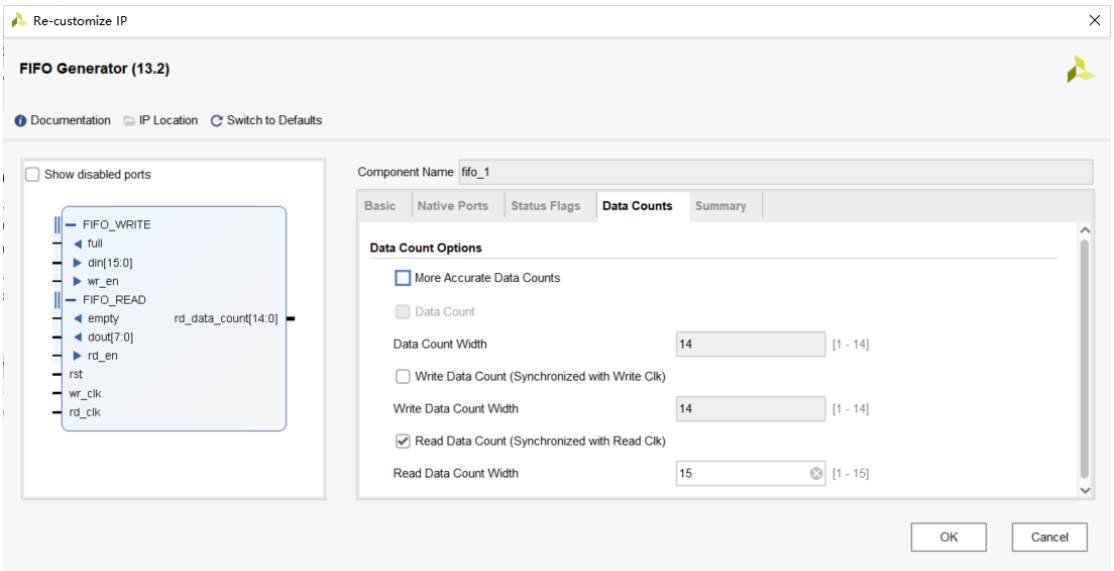
```
//fifo 先读高字节
always@(posedge Clk)
begin
    fifowrdata[15:8] <= #1  adc_data_mult_ch[7:0];
    fifowrdata[7:0] <= #1  adc_data_mult_ch[15:8];

end
```

FIFO IP 核（fifo_tx）

fifo_tx 模块需要接收从 ADC 采集控制模块输出的 16 位数据，数据经缓存后由以太网发送模块读取。ADC 采集控制模块的工作时钟为 50MHz，千兆以太网接收模块工作时钟为 125MHz，模块 fifo_tx 仍需配置为一个具备双时钟结构的 FIFO 进行异步数据的收发。接口类型选择 Native,双时钟 FIFO 模式选择为 Independent Clock Block RAM。Read Mode 类型选择 First Word Fall Through 模式。FIFO 读位宽设置的为 16 位，写位宽设置的为 8 位，数据深度设置为 16384。这里需引出 rd_data_count 接口，输出 FIFO 中可读取数据的数据量，用作 FIFO 读控制的参考信号。配置信息如下图所示。



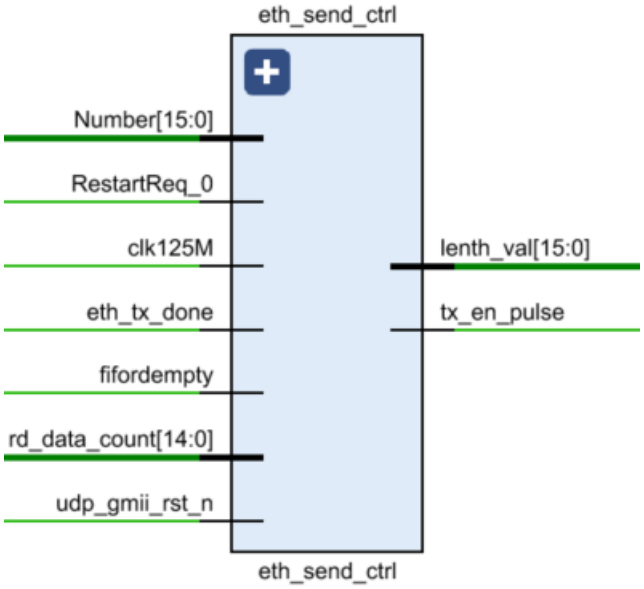




wr_clk 连接 Clk50m_0 输入 50MHz 时钟信号，wr_en、full 与 din 信号同步于 wr_clk，FIFO 写数据由 ADC 采集控制模块 fifowrreq 控制，命令设置通道数据有效时使通过 din 往 FIFO 写数据。rd_clk 输入 125MHz 时钟信号，empty、dout、rd_en、rd_data_count 同步于 rd_clk，当 rd_en 有效时外部从 FIFO 读数据，rd_data_count 信号输出到网口发送控制模块 eth_send_ctrl。

网口发送控制模块（eth_send_ctrl）

该模块主要负责配置控制网口发送模块的使能控制信号 tx_en_pulse，通过 lenth_val 信号对以太网数据帧数据长度进行控制。为保证以太网有效传输效率，以太网帧最大长度 1518 字节（数据段 1500 字节），其中数据段 1500 字节还包括 20 字节 IP 报文头部和 8 字节 UDP 报文头部，所以数据帧发送的 AD7606 采集的数据最大长度为 1472 字节。从 ad7606_instruct 模块获取采样请求信号 RestertReq、采集的数据总量 Number。RestertReq 信号有效，系统开始采样，同时，根据 Number 信号的值确定数据帧发送的 AD7606 采集数据的最大长度。数据帧发送的 AD7606 采集数据的最大长度作为判断条件，FIFO 计数信号 rd_data_count 的数值满足一帧数据帧发送采集数据长度，产生 tx_en_pulse 信号，以太网发送模块开始读取 FIFO 中的数据。

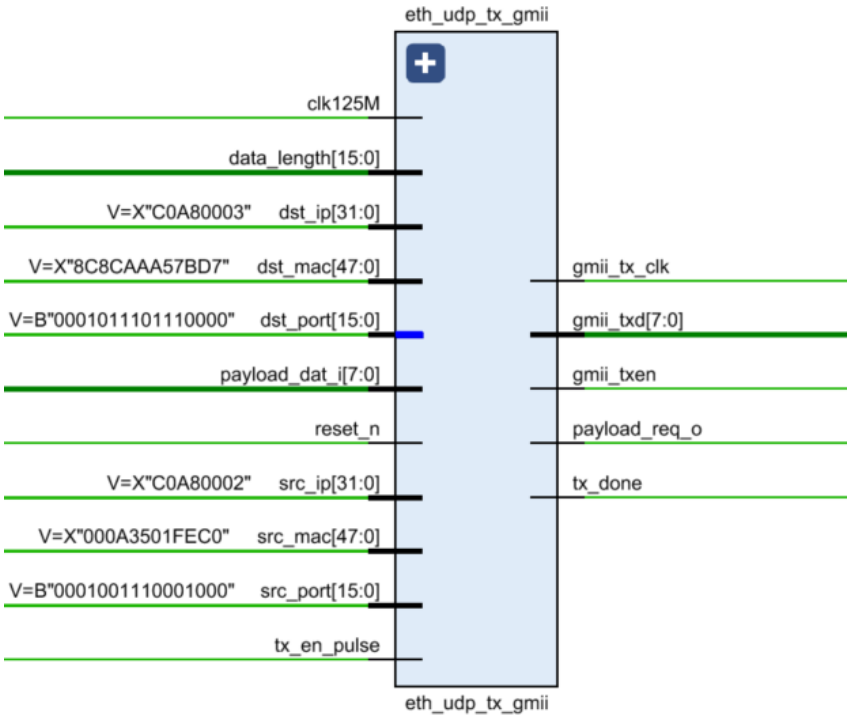


网口发送模块（eth_udp_tx_gmii）

网口发送模块在获取到采集数据并对数据进行打包，然后以以太网帧的形式发送到 PC 端。为配置以太网数据帧，网口发送模块需要获取目的 MAC 地址（dsc_mac）、目的 IP 地址(dst_ip)、目的端口号(dst_port)、源 MAC 地址(src_mac)、源 IP 地址(src_ip)、源端口号(src_port)。上板验证中目的 MAC 地址为电脑的实际 MAC 地址，需要根据根据实际情况进行更改，如果不知道自己电脑网卡的 MAC 地址，就在 DOS 命令窗口，用 ipconfig －all 命令查看。

```
.dst_mac        (48'h8C_8C_AA_A5_7B_D7),
.src_mac        (48'h00_0a_35_01_fe_c0),
.dst_ip         (32'hc0_a8_00_03),
.src_ip         (32'hc0_a8_00_02),
.dst_port       (16'd6000),
.src_port       (16'd5000),
```

网口发送模块接收到 tx_en_pulse 信号后，输出 payload_req_0 信号开始读取 fifo_tx 中的采集数据，每次采集一字节数据。GMII 接口中 Transmit Data 数据发送信号，共 8 根信号线，当 gmii_txen 有效，gmii_txd 开始向 PC 端发送数据，每次可发送一字节数据。从 FIFO 读取采集数据的数据量由 eth_send_ctrl 模块的 lenth_val 输出值决定，一次数据帧发送完成网口发送模块产生 tx_done 信号，等待下一次的发送。

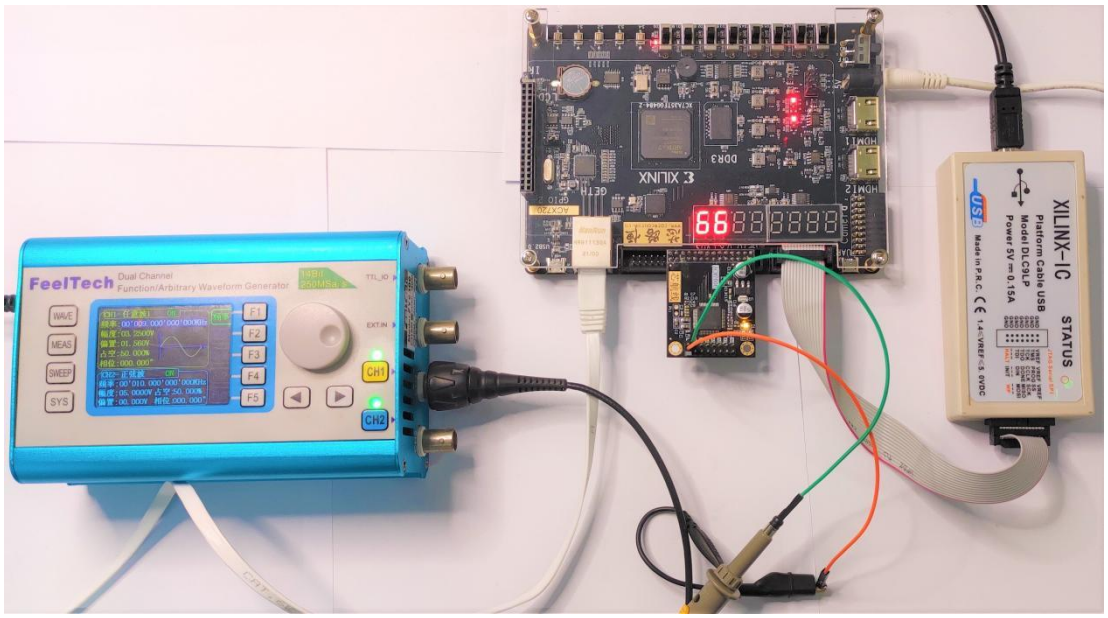


工程演示

硬件连接

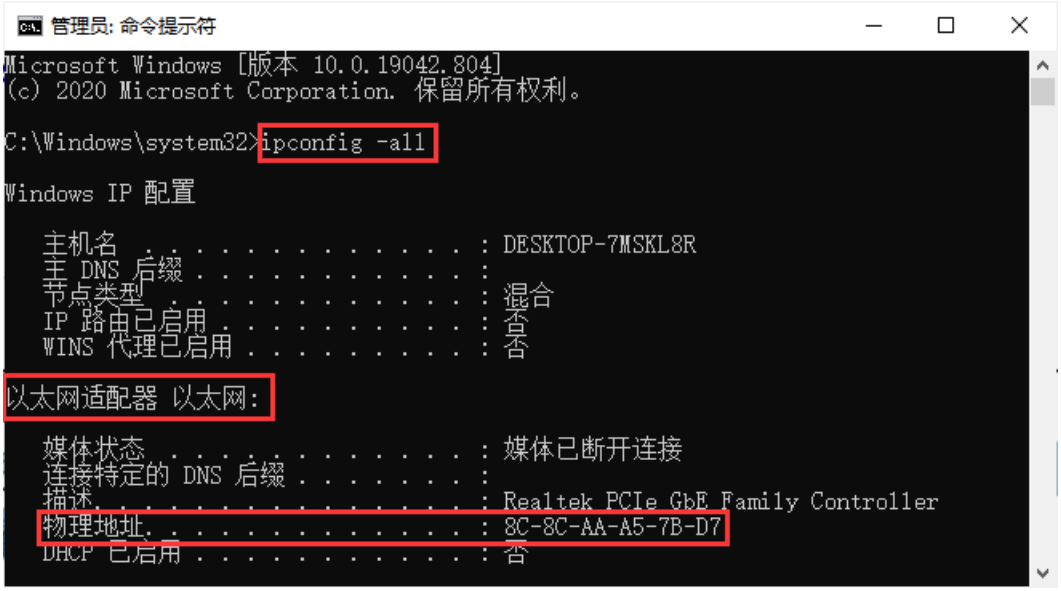
要验证本次实验，首先第一步就是对本次实验的整个系统硬件进行连接，本次工程硬件具体的连接如下图所示。这里我们用信号发生器，分别输出 200Hz 的正弦波、方波、三角波，用户可以直接使用信号发生器作为信号源。网线连接好开发板以太网接口和电脑的以太网接口，杜邦线连接 AD7606 模块 A1 端口，AD7606 模块在正确连接后右边会多出 6 组排针，由于视觉缘故，这里很容易连接错误，连接错误后网口调试助手上容易出现有发送而无接收的

情况，电源线与下载器连接如图中所示，在硬件连接好后打开开关就可以下载程序了，用户可以直接将我们提供的 bit 文件夹下的文件烧写进板中，也可打开工程一步步下载。



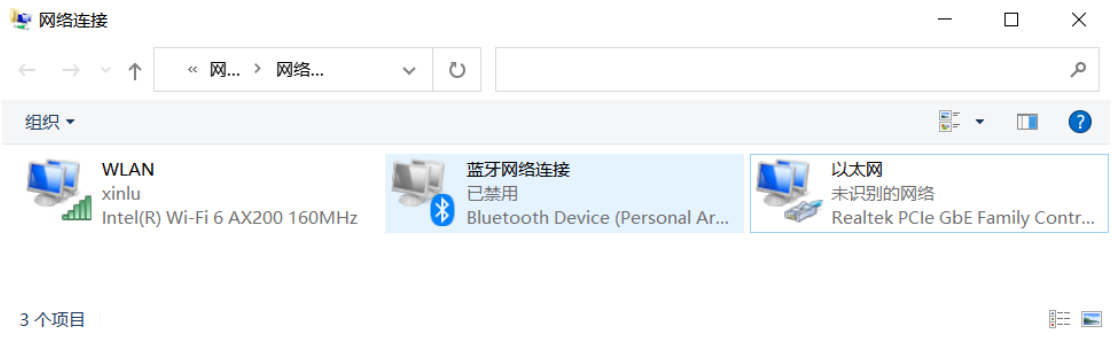
网络连接设置

硬件连接好之后我们就可以开始对以太网的连接设置。首先，我们需要查看自己电脑网卡的 MAC 地址，就在 DOS 命令窗口，用 ipconfig－all 命令查看。完成网口发送模块的设置。



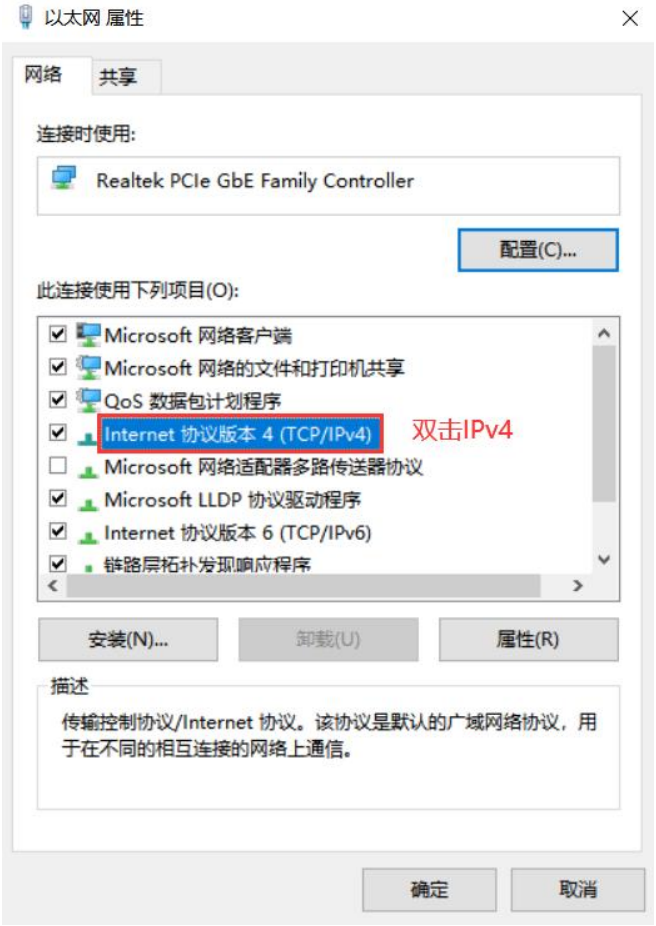
在电脑上进入【控制面板】->【网络连接】->【以太网】->【更改适配器选项】，查看网络连接状态。需要看到在活动网络中有本地连接存在，才表明开发板和电脑的网络才已经连通。此时如果重新下载 sof 文件到开发板中，会发现此本地连接会先消失，然后再重新出现。至于显示的无法连接到网络选项，意思是指无法连接到互联网获取网络上的数据，这是

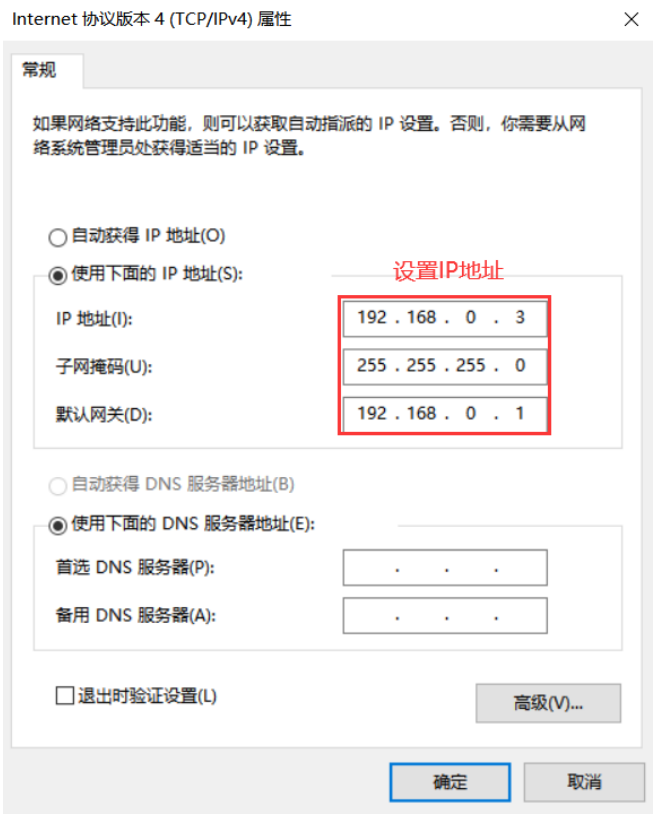
正常的，无需在意。



点击“本地连接”文字，以查看该网络状态，确认当前连接速度为千兆速率（1000.0 Mbps）

上板验证的目的 IP 固定设置的 192.168.0.3，端口号固定设置为 6000。这些可根据实际情况更改，需与 FPGA 网口相连接的电脑网口的配置保持一致。在上述本地连接状态中，点击属性，并在弹出的属性对话框中双击【Internet 协议版本 4 （TCP/Ipv4）】选项，然后再弹出的属性对话框中设置静态 IP 地址。如下图所示。


以太网 属性
网络
共享
连接时使用:
Realtek PCIe GbE Family Controller
配置(C)...
此连接使用下列项目(O):
Microsoft 网络客户端
Microsoft 网络的文件和打印机共享
QoS 数据包计划程序
Internet 协议版本 4 (TCP/IPv4)
双击IPv4
Microsoft 网络适配器多路传送器协议
Microsoft LLDP 协议驱动程序
Internet 协议版本 6 (TCP/IPv6)
链路层拓扑发现响应程序
安装(N)...
卸载(U)
属性(R)
描述
传输控制协议/Internet 协议。该协议是默认的广域网络协议，用于在不同的相互连接的网络上通信。
确定
取消

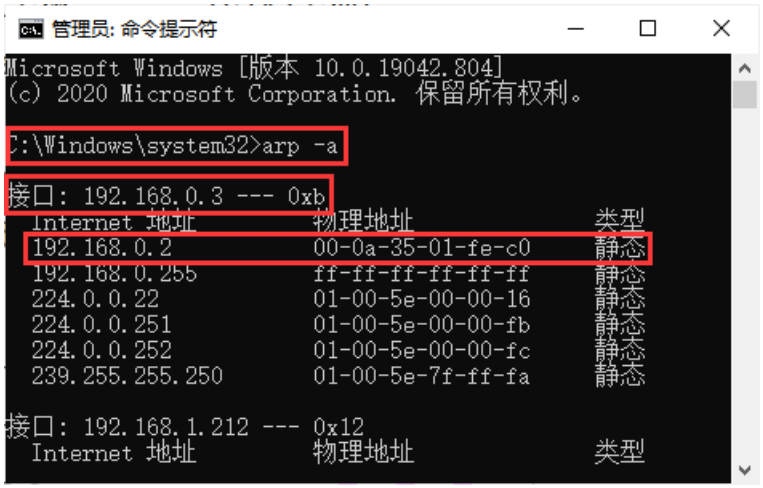

Internet 协议版本 4 (TCP/IPv4) 属性
常规
如果网络支持此功能，则可以获取自动指派的 IP 设置。否则，你需要从网络系统管理员处获得适当的 IP 设置。
自动获得 IP 地址(O)
使用下面的 IP 地址(S):
设置IP地址
IP 地址(I):
192 . 168 . 0 . 3
子网掩码(U):
255 . 255 . 255 . 0
默认网关(D):
192 . 168 . 0 . 1
自动获得 DNS 服务器地址(B)
使用下面的 DNS 服务器地址(E):
首选 DNS 服务器(P):
备用 DNS 服务器(A):
退出时验证设置(L)
高级(V)...
确定
取消

由于本测试工程不支持 ARP 协议，因此只能通过静态绑定的方式来强制将开发板的 IP 地址和 MAC 地址关联在一起。这样，当 PC 发送给 192.168.0.2 的数据包的时候，目标 MAC 地址自动为开发板的 MAC 地址。

操作时先以管理员身份运行 cmd.exe 程序（该文件在 C:\Windows\System32 路径下），也就是我们常说的命令行窗口。由于有用户反应在使用时无法成功绑定 arp，经过分析就是操作权限不够，所以这里强调要以管理员身份运行 cmd.exe。然后在窗口中输入下述命令：

```
arp    -s    192.168.0.2    00-0a-35-01-fe-c0
```

绑定后我们可以用 arp　-a 命令来查看 PC 上绑定的结果，如下图所示：

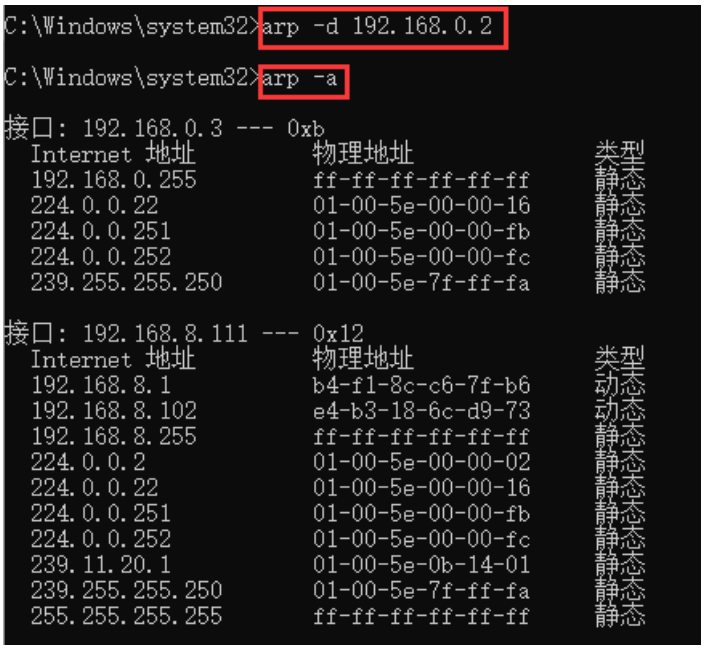


IP 绑定错误

例如将 IP192.168.0.2 绑定到错误的接口（WiFi）。

为了更正回来，我们可以先用 arp-d 命令将该 IP 删除，命令如下：

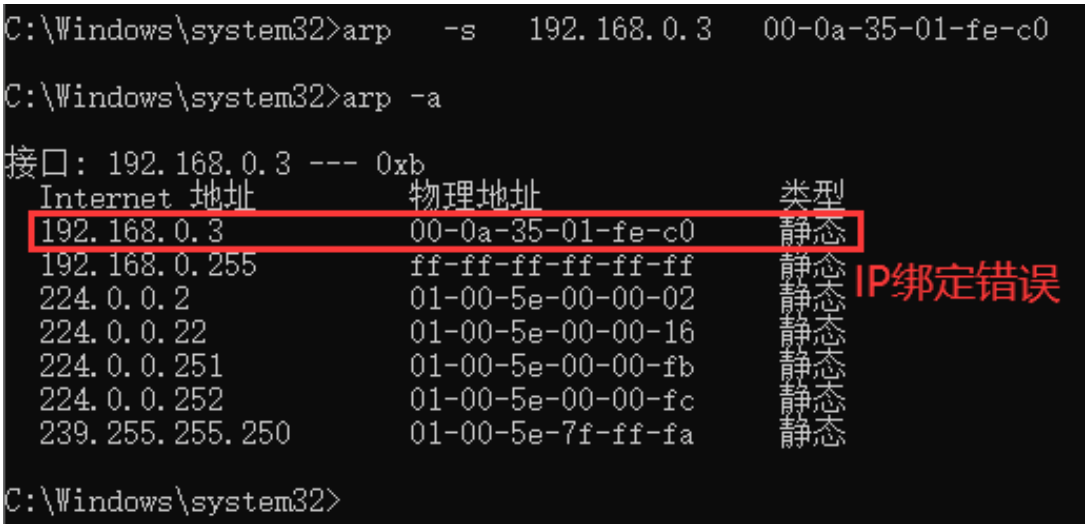
```
arp -d 192.168.0.2
```



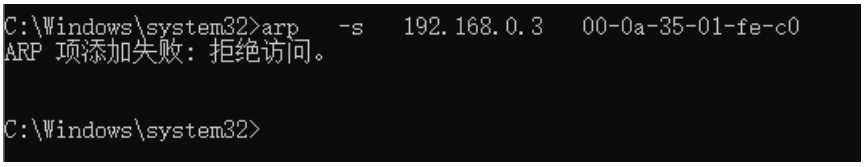
绑定时可暂时禁用其它网络接口，“网络连接\状态\更改适配器选项\网络禁用”，这样可确保 IP 绑定到以太网接口。

又如这里，我们把误把 IP192.168.0.3 绑定到了开发板的 MAC 地址 00-0a-35-01-fe-c0 上，为了更正回来，我们可以先用 arp-d 命令将该 IP 删除，命令如下：

```
arp -d 192.168.0.3
```

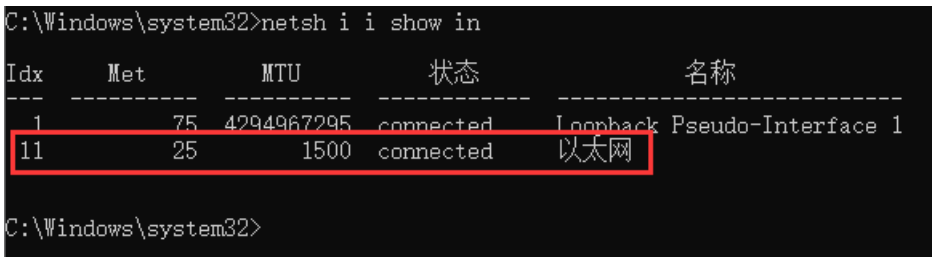


随后便尝试使用 arp -s 指令再次添加，发现被拒绝访问。



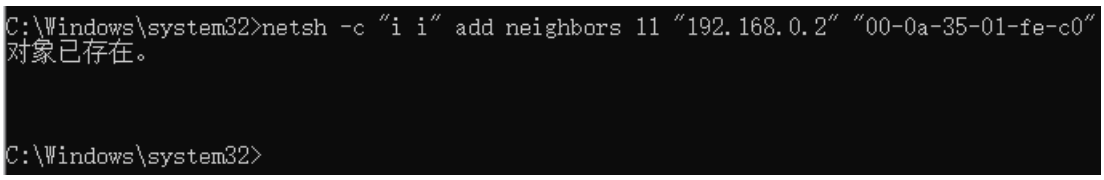
接下来使用 netsh 指令查看本地连接以太网的 Idx 值，命令如下：

```
netsh i i show in
```

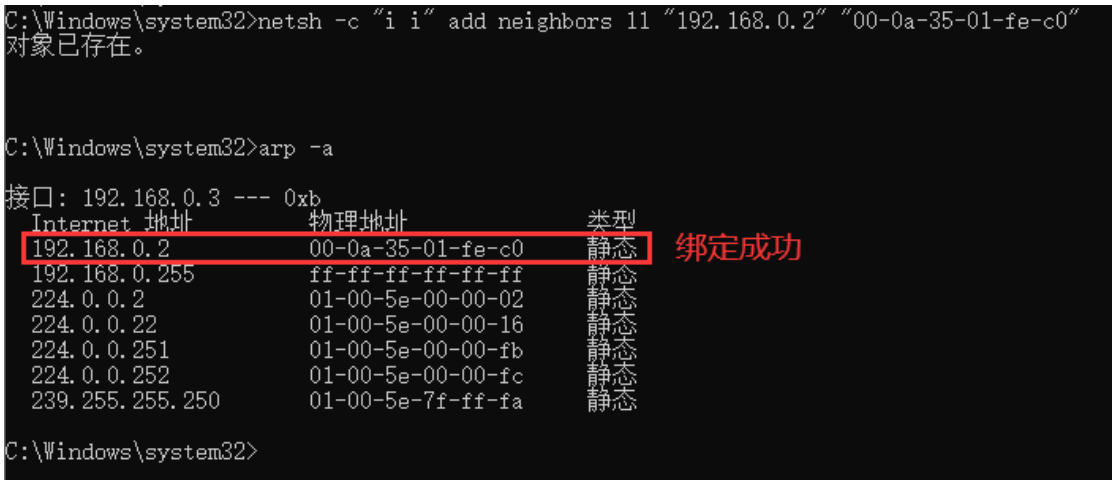


可以看到这里笔者电脑以太网的 Idx 为 11，接着我们根据这个值来绑定 arp，命令如下：

```
netsh -c "i i" add neighbors 11 "192.168.0.2" "00-0a-35-01-fe-c0"
```



接下来我们使用 arp -a 命令查看，即可看到 IP 地址已经绑定成功。



更多问题可上 www.corecourse.cn 搜索“以太网实验常见问题”关键词搜索相关帖子。

配置 ADC

程序下载完成后开发板上会有两个红灯亮起，接着便可以对 AD7606 进行配置了，首先打开网络调试助手（NetAssist.exe）并按照如下所述设置各项参数。

①选择协议类型为 UDP。

②设置本地 IP 地址为 192.168.0.3。

③设置本地端口号为 6000。

④点击【连接】按钮以创建连接，连接上后该按钮为红色“断开”字样。

⑤连接上后，设置目标主机为 192.168.0.2，目标端口为 5000。

⑥在文本框中输入希望发送的文本内容。

以上面接收转命令模块中介绍到的数据帧格式对 AD7606 的四个寄存器进行配置。

例如，PC 端要设置采样数据个数(DataNum 寄存器，地址为 2)为 16000（0x3E80）个，则发送的数据帧内容为：0x55 0xA5 0x02 0x00 0x00 0x3E 0x80 0xF0。

例如，PC 端要设置采样速率(ADC_Speed_Set 寄存器，地址为 3)为 100K，则对应的 ADC_Speed_Set 值为 1000000000/20/100000 – 1 = 499（0x000001F3）个，则发送的数据帧内容为：0x55 0xA5 0x03 0x00 0x00 0x01 0xF3 0xF0。

当上述设置都设置完成后，就可以向 0 号寄存器写入任意值，来开始一次采样传输了。数据帧内容可以为 0x55 0xA5 0x00 0x00 0x00 0x00 0x00 0xF0，这里需要注意的是 0 号寄存器必须放在最后，因为 0 号寄存器负责启动 ADC，ADC 在未配置完全的情况下开始启动，数据很容易输出错误值。就能实现以 200K 的采样速率，对 1 个通道进行采样，共采集 16384 个数据。四个寄存器对应的代码如下：

DataNum：　　55 A5 02 00 00 40 00 F0

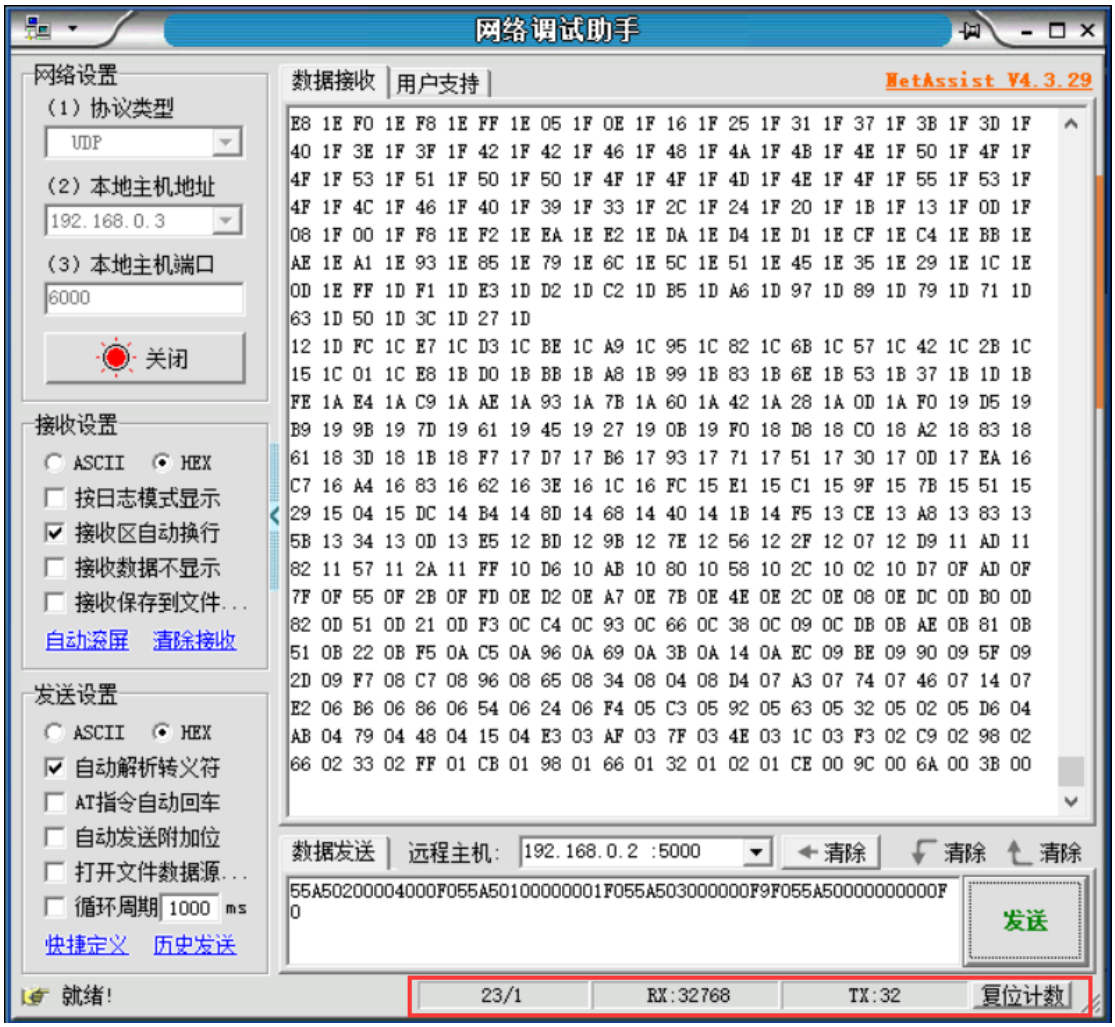
ChannelSel：　55 A5 01 00 00 00 01 F0

ADC_Speed_Set：　55 A5 03 00 00 00 F9 F0

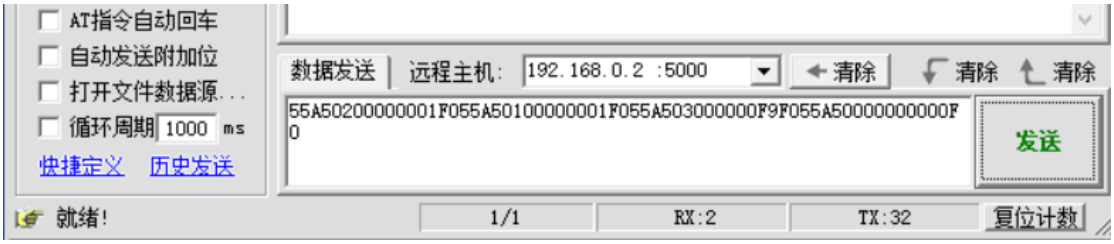
RestartReq：　55 A5 00 00 00 00 00 F0

配置成网络调试助手发送的数据格式为：

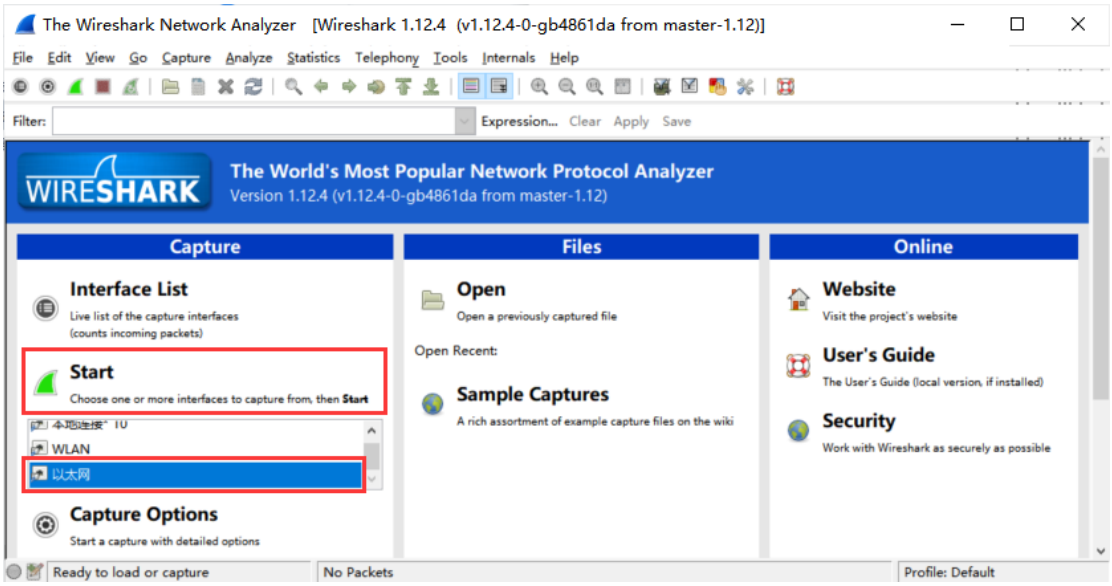
55A50200004000F055A50100000001F055A503000000F9F055A50000000000F0



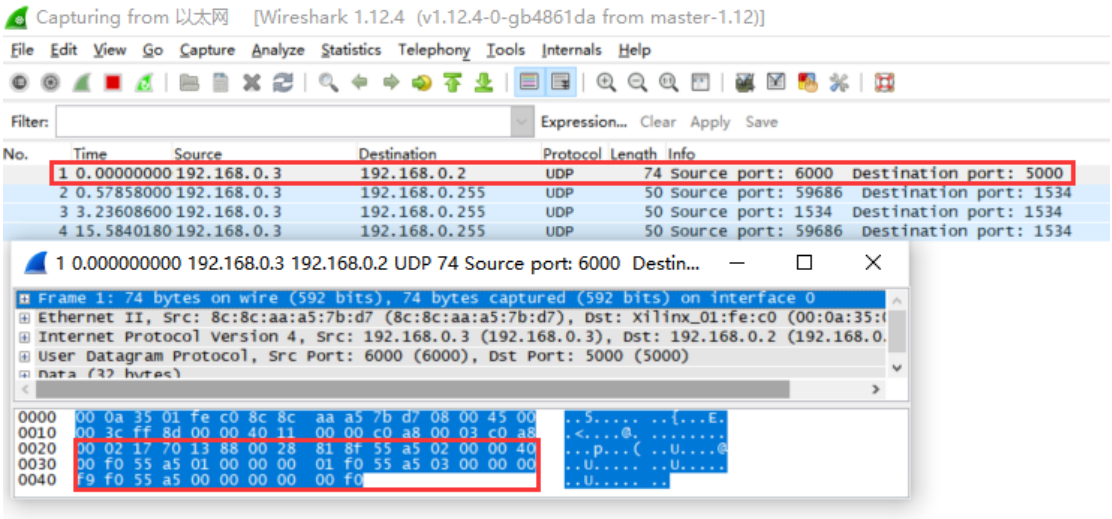
共采集 16384 个数据，通过观察接收数据量为 32768，与预期一致，也可以设置不同的采样数据量来简单验证设计是否达到预期效果。比如采集一个数据观察网络调试助手收到的数据量是否正确。



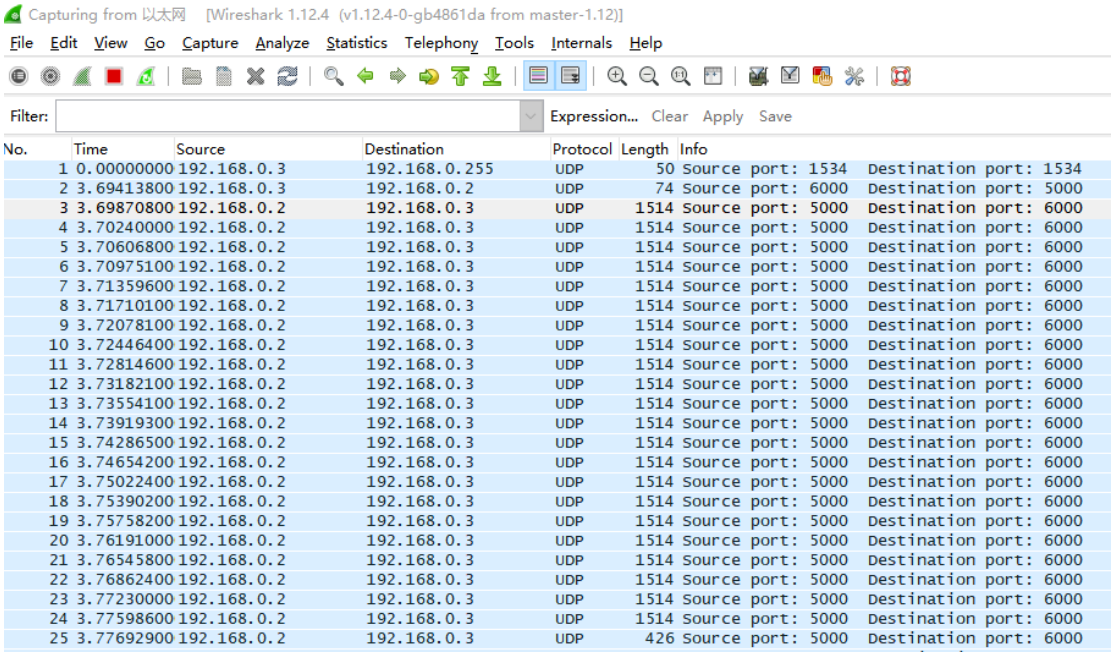
同时也可以安装网络抓包工具 Wireshark，我们在实验的时候可以用这工具来查看 PC 网口发送的数据和接收到的数据。打开安装好的 wireshark 抓包工具。在软件界面选择您 PC 的有线网卡，按开始按钮开始抓包。（不同的版本界面有所差异，我们提供的是 Wireshark_win64_V1.12.4_setup.1427187922.exe）。



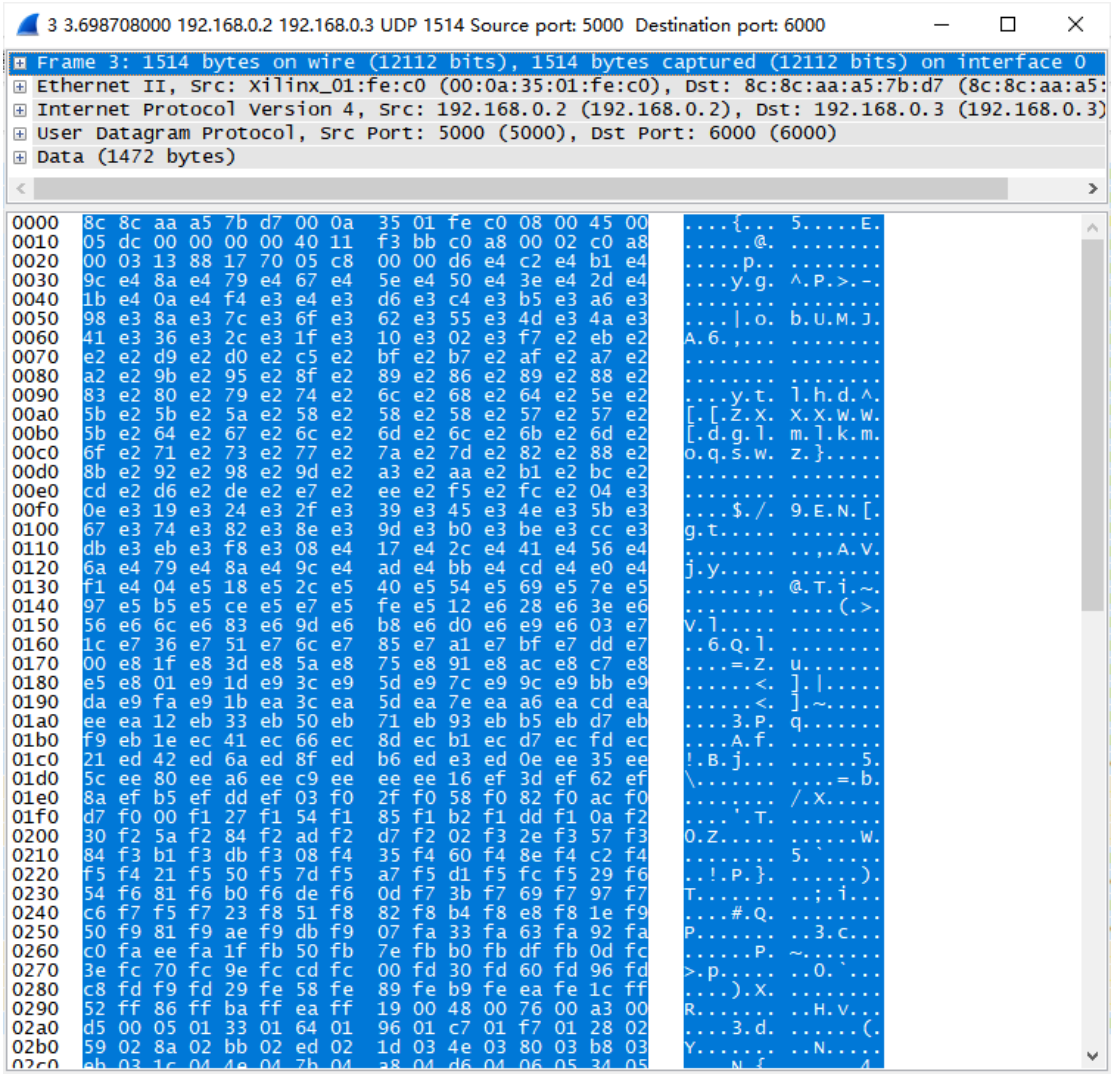
从 wireshark 抓到的数据包可以到与网络调试助手发送的数据一致。



同时也可以对采集到的数据进行查看

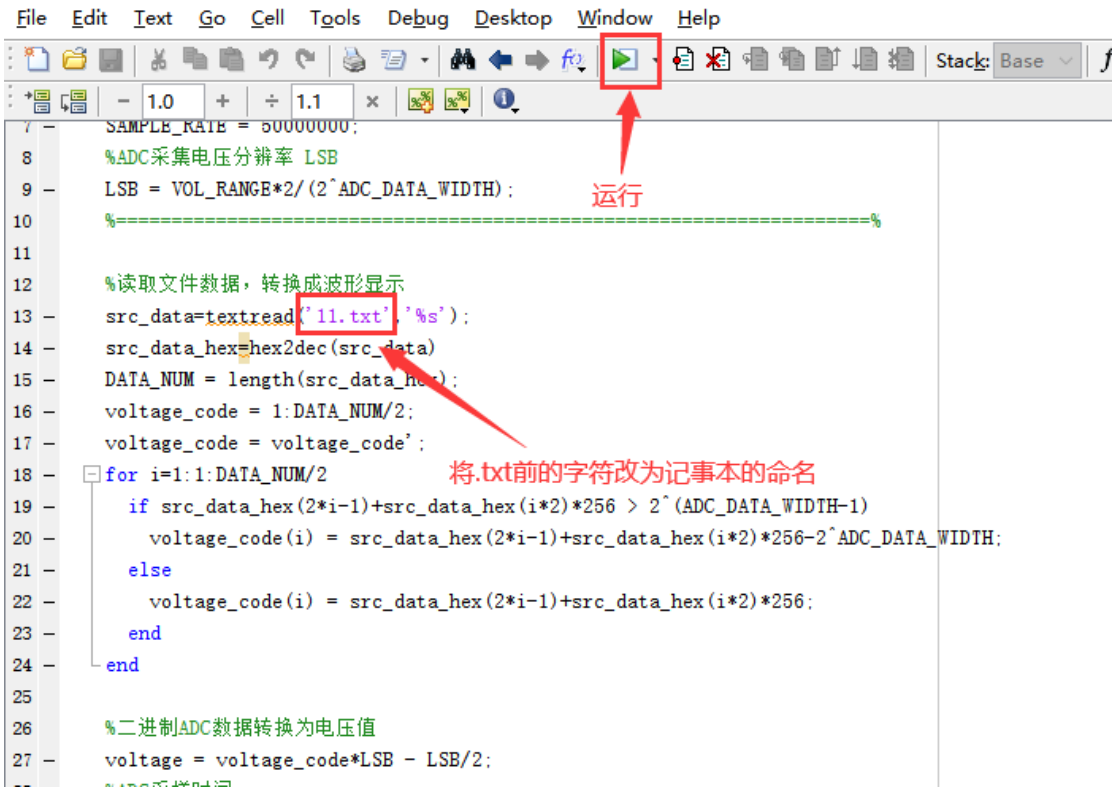


No.	Time	Source	Destination	Protocol	Length	Info
1	0.00000000	192.168.0.3	192.168.0.255	UDP	50	Source port: 1534 Destination port: 1534
2	3.69413800	192.168.0.3	192.168.0.2	UDP	74	Source port: 6000 Destination port: 5000
3	3.69870800	192.168.0.2	192.168.0.3	UDP	1514	Source port: 5000 Destination port: 6000
4	3.70240000	192.168.0.2	192.168.0.3	UDP	1514	Source port: 5000 Destination port: 6000
5	3.70606800	192.168.0.2	192.168.0.3	UDP	1514	Source port: 5000 Destination port: 6000
6	3.70975100	192.168.0.2	192.168.0.3	UDP	1514	Source port: 5000 Destination port: 6000
7	3.71359600	192.168.0.2	192.168.0.3	UDP	1514	Source port: 5000 Destination port: 6000
8	3.71710100	192.168.0.2	192.168.0.3	UDP	1514	Source port: 5000 Destination port: 6000
9	3.72078100	192.168.0.2	192.168.0.3	UDP	1514	Source port: 5000 Destination port: 6000
10	3.72446400	192.168.0.2	192.168.0.3	UDP	1514	Source port: 5000 Destination port: 6000
11	3.72814600	192.168.0.2	192.168.0.3	UDP	1514	Source port: 5000 Destination port: 6000
12	3.73182100	192.168.0.2	192.168.0.3	UDP	1514	Source port: 5000 Destination port: 6000
13	3.73554100	192.168.0.2	192.168.0.3	UDP	1514	Source port: 5000 Destination port: 6000
14	3.73919300	192.168.0.2	192.168.0.3	UDP	1514	Source port: 5000 Destination port: 6000
15	3.74286500	192.168.0.2	192.168.0.3	UDP	1514	Source port: 5000 Destination port: 6000
16	3.74654200	192.168.0.2	192.168.0.3	UDP	1514	Source port: 5000 Destination port: 6000
17	3.75022400	192.168.0.2	192.168.0.3	UDP	1514	Source port: 5000 Destination port: 6000
18	3.75390200	192.168.0.2	192.168.0.3	UDP	1514	Source port: 5000 Destination port: 6000
19	3.75758200	192.168.0.2	192.168.0.3	UDP	1514	Source port: 5000 Destination port: 6000
20	3.76191000	192.168.0.2	192.168.0.3	UDP	1514	Source port: 5000 Destination port: 6000
21	3.76545800	192.168.0.2	192.168.0.3	UDP	1514	Source port: 5000 Destination port: 6000
22	3.76862400	192.168.0.2	192.168.0.3	UDP	1514	Source port: 5000 Destination port: 6000
23	3.77230000	192.168.0.2	192.168.0.3	UDP	1514	Source port: 5000 Destination port: 6000
24	3.77598600	192.168.0.2	192.168.0.3	UDP	1514	Source port: 5000 Destination port: 6000
25	3.77692900	192.168.0.2	192.168.0.3	UDP	426	Source port: 5000 Destination port: 6000



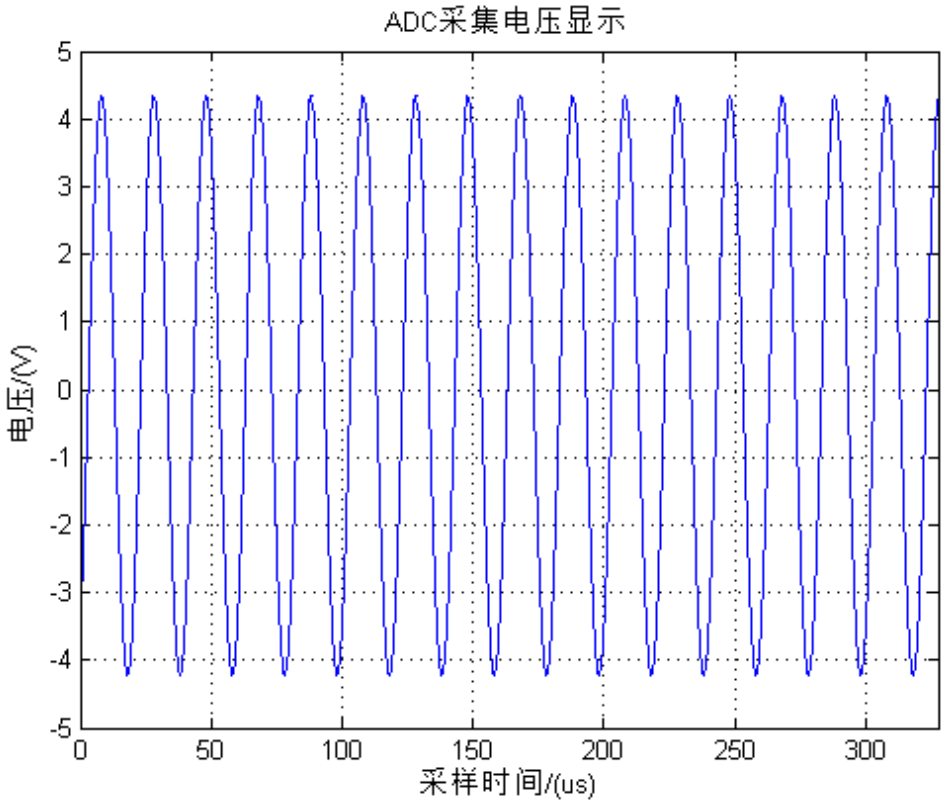
MATLAB 图像绘制

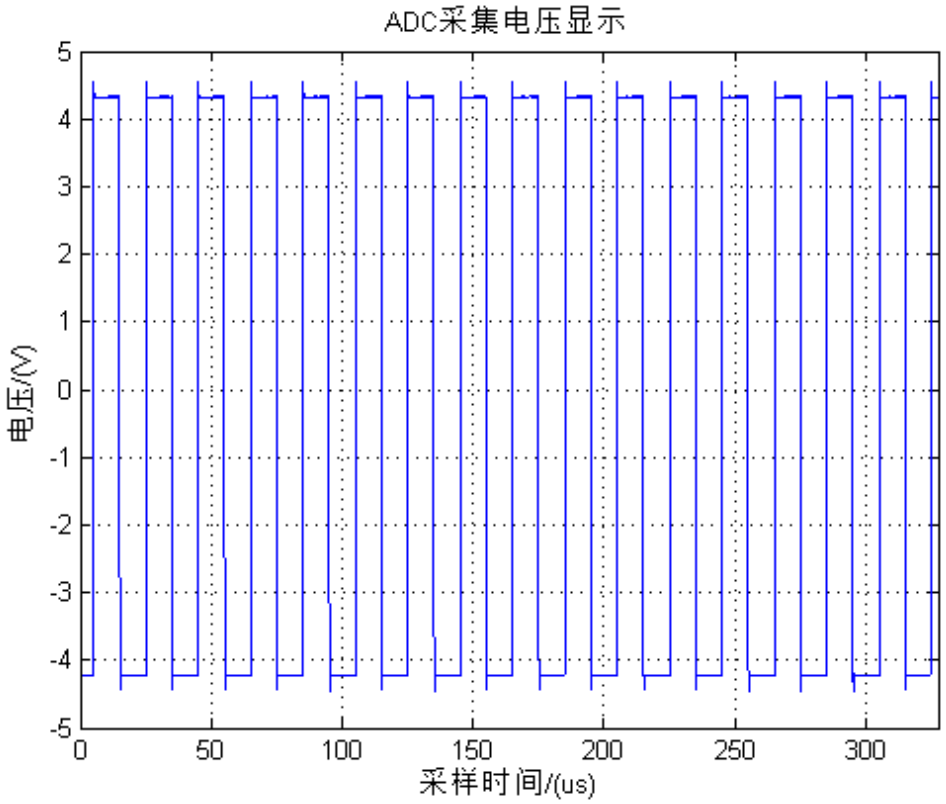
将拷贝好的记事本移动到 MATLAB 解压文件夹中（记得必须是纯英文路径），双击我们提供的 ADCdata_to_wave_v1_1.m 文件，在打开方式里选择以 MATLAB 打开，将方框中.txt 前的字符修改为你对记事本的命名，随后运行便可以直观的看到数据是否正确。

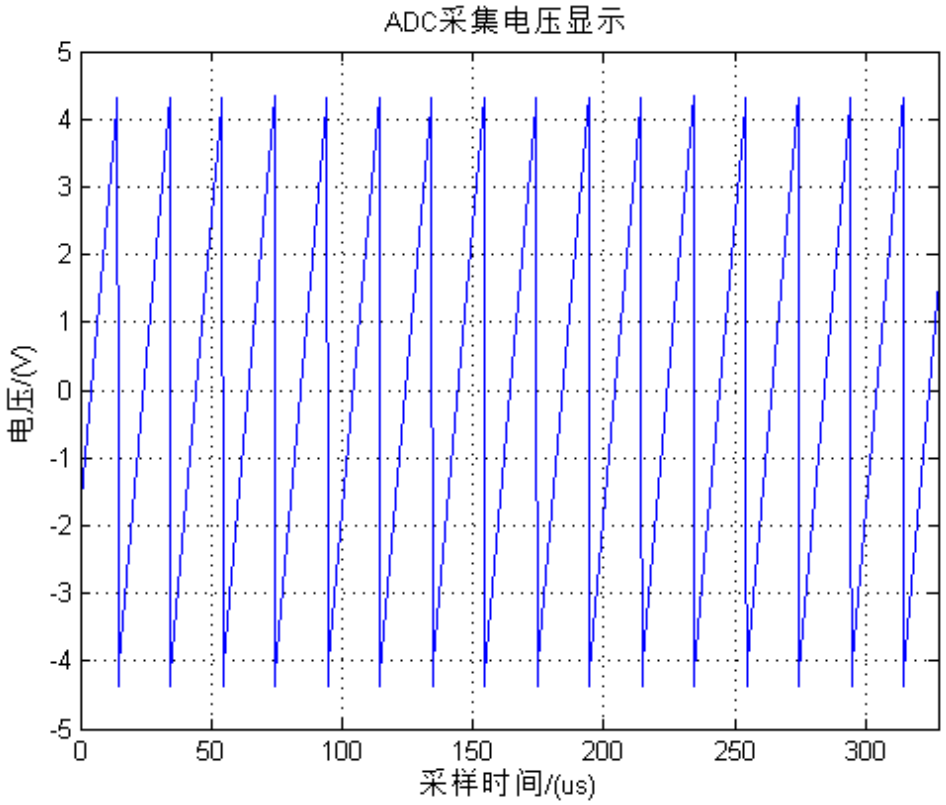



```
SAMPLE_RATE = 50000000;
%ADC采集电压分辨率 LSB
LSB = VOL_RANGE*2/(2^ADC_DATA_WIDTH);
%=========================================================================%

%读取文件数据，转换成波形显示
src_data=textread('11.txt','%s');
src_data_hex=hex2dec(src_data)
DATA_NUM = length(src_data_hex);
voltage_code = 1:DATA_NUM/2;
voltage_code = voltage_code';
for i=1:1:DATA_NUM/2
  if src_data_hex(2*i-1)+src_data_hex(i*2)*256 > 2^(ADC_DATA_WIDTH-1)
    voltage_code(i) = src_data_hex(2*i-1)+src_data_hex(i*2)*256-2^ADC_DATA_WIDTH;
  else
    voltage_code(i) = src_data_hex(2*i-1)+src_data_hex(i*2)*256;
  end
end

%二进制ADC数据转换为电压值
voltage = voltage_code*LSB - LSB/2;
```

下面附上分别对正弦波，方波，三角波进行采样绘图最后得到的图形，相应的数据存放在本次实验提供的压缩包的 data 文件夹下。


ADC采集电压显示
电压/(V)
采样时间/(us)
5
4
3
2
1
0
-1
-2
-3
-4
-5
0
50
100
150
200
250
300


ADC采集电压显示
电压/(V)
采样时间/(us)
5
4
3
2
1
0
-1
-2
-3
-4
-5
0
50
100
150
200
250
300



总结

本次实验介绍了基于 AD7606 的千兆以太网收发，用户通过网口调试助手以太网帧向开发板发送指令数据配置 AD7606 的四个寄存器，控制 ADC 进行采样，并将数据缓存后再组成以太网帧，经由网口发送至电脑，借由网口调试工具对数据进行查看。注意在配置寄存器的过程中要考虑到 FIFO 的写深度，一次采样所能采集的数据应该小于或等于 FIFO 的写深度，同时负责启动 ADC 的 0 号寄存器应该放在代码指令的最后进行配置。本次实验涉及时钟域的转换，以及采集数据位宽的转换，完成这些转换需要对 FIFO 有一定的了解。本次实验也要求读者对以太网协议有足够的认识，想要进一步理解以太网功能原理可参考 ACX720 配套资料《小梅哥 Xilinx FPGA 自学教程》35 章。

修订记录

V1.0	2021/04/9	首次发布