版本信息：
版本
REV2020
时间
10/20/2020

07 米联客 2020 版 FPGA 教程(FT60X 篇)

电子版自学资料

常州一二三电子科技有限公司
溧阳米联电子科技有限公司

米联客(MSXBO)04QQ　群：516869816
米联客(MSXBO)03QQ　群：543731097
米联客(MSXBO)02QQ　群：86730608
米联客(MSXBO)01QQ　群：34215299

版本	时间	描述
Rev2019	2019-09-17	首次更新，6 个例子
Rev2020	2020-10-20	升级至 2020 版教程

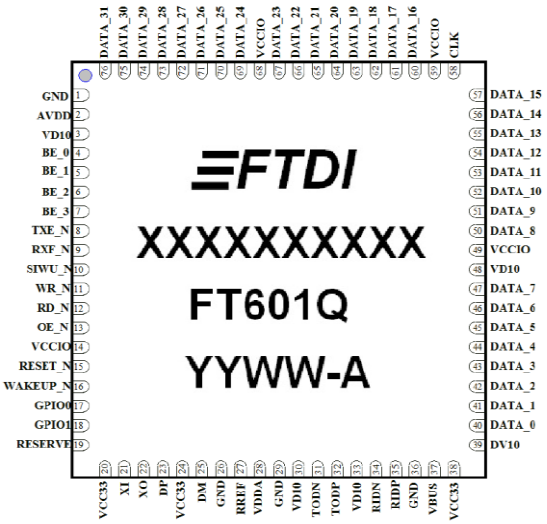
目录

CH01 初探 USB3.0 极简方案 FT601Q 芯片方案

软件版本：VIVADO2019.2
操作系统：WIN10 64bit
硬件平台：适用XILINX 7系列FPGA(没有USB3.0接口的开发板需要购买FEP USB3.0子卡)
米联客(MSXBO)技术社区： www.uisrc.com 欢迎大家给我提问！！

1.1 概述

A703_35T 的 IO 没有引出至底板 USB3.0 接口，而使用相同底板的 A703_100T 的 IO 已经引出至底板 USB3.0 接口。本 A703_35T 的例子中使用的是 FEP 接口外接 USB601Q 子卡。



说来惭愧，从米联客开发板硬件上开始支持 USB3.0 芯片，FT600/601Q 芯片方案已经快 1 年了，但是一直没给出非常详细的使用 demo。市面上的开发板目前大都采用 CY3014 方案，那么米联客为什么一定要选择 FT600/601Q 这款开发板方面用的不多的芯片来做 USB3.0 方案呢？为什么不买个别人的开发板来 copy 下方案呢？为什么要耗费这么大精力呢？如果从商业经济角度来说，拿现成的方案来做自然是最省事，技术风险也最小，但是米联客作为开发板设计生产厂家，以及原创资料的倡导者，希望能给整个开发板生态做出一份自己的贡献，所以我们做的事情，要能给广大开发者提供一定的帮助，而不是 copy。

1.1.1 FT600/601Q 的技术参数:

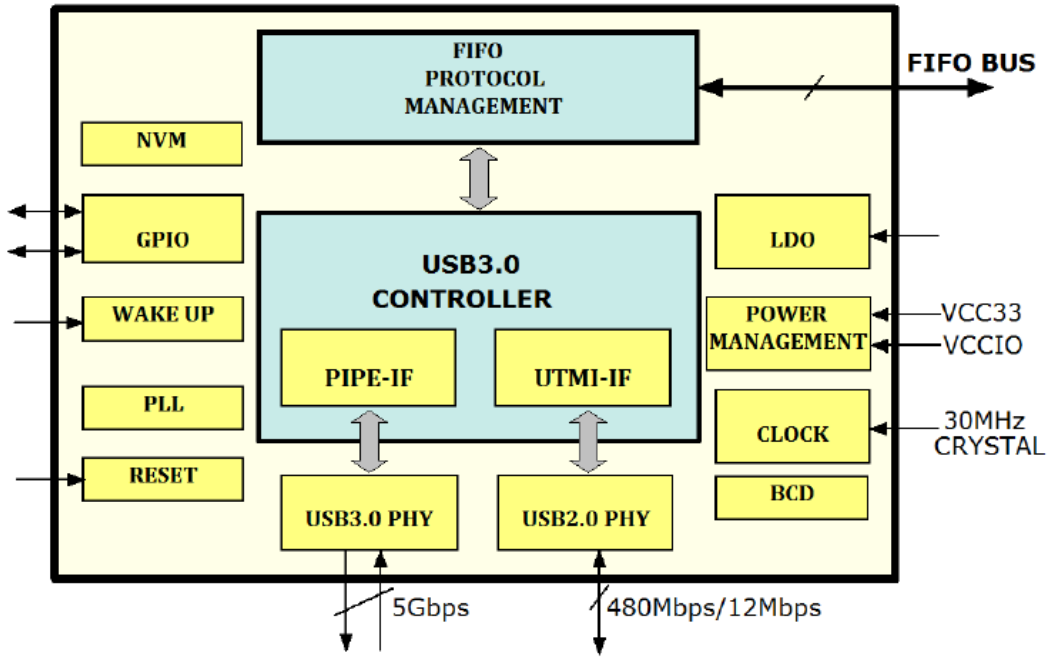
>>FT600&601Q 芯片是 FT 最新推出的 USB3.0 to FIFO interface IC，实现 USB3.0 与 16/32bit 并行 IO 接口之间的数据传输。
>>整个 USB 通信协议全部由芯片驱动自行完成，开发者无须考虑 USB 底层固件的编程。
>>兼容支持 USB3.0（5Gbps）,向下兼容 USB2.0（480Mbps and 12Mbps）传输。
>>高达 8 个可配置 Endpoint.
>>支持 2 种 FIFO 传输协议，最大传输可达 400MB/s。
>>芯片内部有 16K 字节的缓冲区，可以进行数据的大吞吐量操作。
>>支持远程唤醒功能。

>>芯片支持多种 IO 电压：1.8V,2.5V.3.3V。>>通过 16bit D[O：15]或 32bit D[0:31]并行数据线和读写状态 / 控制线 RXF、TXE、RD、WR，加上时钟 CLK，使能 OE 信号线就可实现与 CPU/FPGA 的数据交换。

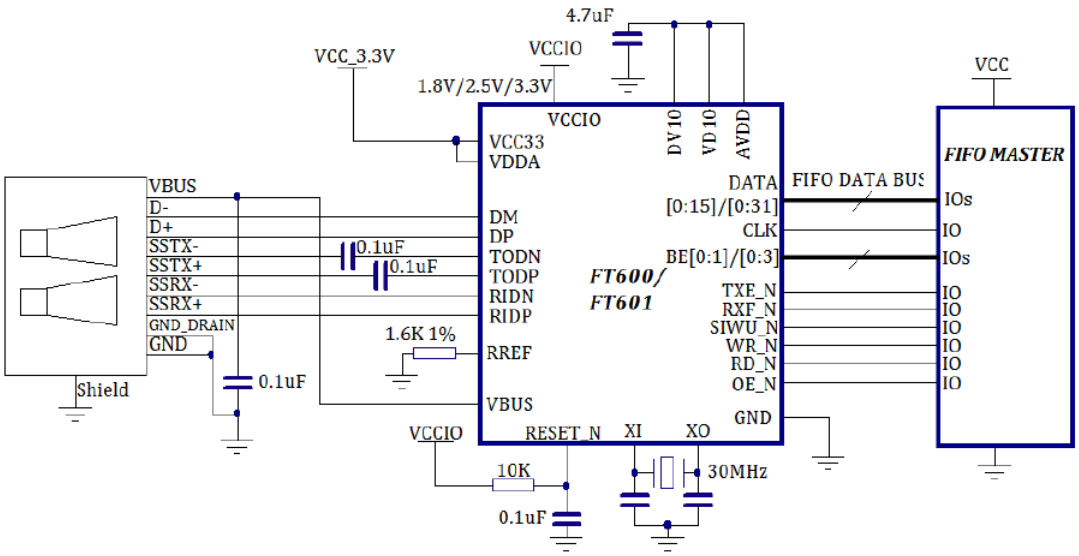
>>该芯片内部集成 1.0V LDO,可提供给芯片核心部分使用。

>>工业级芯片，工作温度范围-40 to 85℃。

1.1.2 芯片构架



1.1.3 极简外围设计



综上所述，可以看出 FT600/601Q 方案绝对是 USB3.0 传输中非常好用的方案。

1.1.4 时序及控制信号

相比官方的 FPGA 测试程序，自己写的测试程序更加简单，阅读方面，可以直接兼容官方的测试软件进行测试。

FT600/601Q 支持的传输模式如下说明，其中 245 Synchronous FIFO 模式和 Multi-Channel FIFO 模式是我们最常用的模式。我们这里介绍的 demo 以 245 Synchronous FIFO 模式为例

GPIO (General purpose input and output) pins

GPOI[1:0] are multifunctional pins. The functions are configured by the chip configuration data. The default chip configuration sets the GPIO pins as FIFO mode configuration input. At the power up, FT600 or FT601 sets the chip to 245 synchronous FIFO mode or multi-channel FIFO modes depending on the GPIO[1:0] input, details in the table below:

GPIO1	GPIO0	Chip mode
0	0	1 channel, 245 Synchronous FIFO mode
0	1	1 channel, Multi-Channel FIFO mode
1	0	2 channel, Multi-Channel FIFO mode
1	1	4 channel, Multi-Channel FIFO mode

To enable the GPIO function, the chip configuration must be updated to set the GPIO function.

To enable the BCD mode, the chip configuration must be updated to set the BCD mode.

When the FT600 or FT601 is configured to support BCD, the GPIO pins are set to output, output changes according to BCD detection result.

GPIO1	GPIO0	Chip mode
0	0	No USB connection or BCD detection on going.
0	1	SDP(standard downstream port) detected
1	0	CDP(Charging downstream port) detected
1	1	DCP(Dedicated charging port) detected

245 Synchronous FIFO 模式读时序

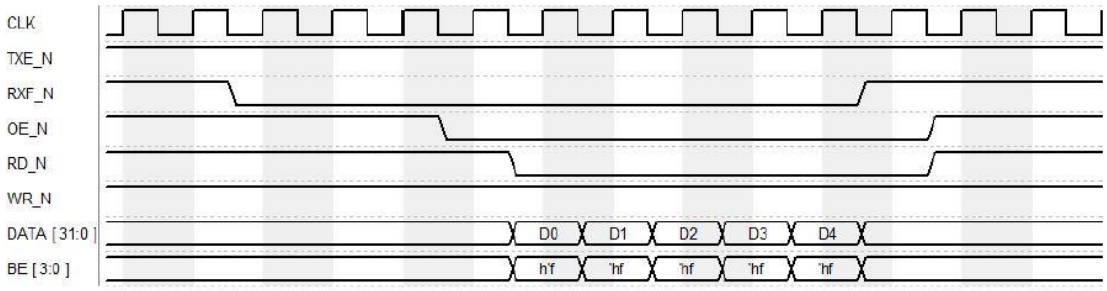


Figure 4.6 245 Synchronous FIFO mode bus master read cycle

245 Synchronous FIFO 模式写时序

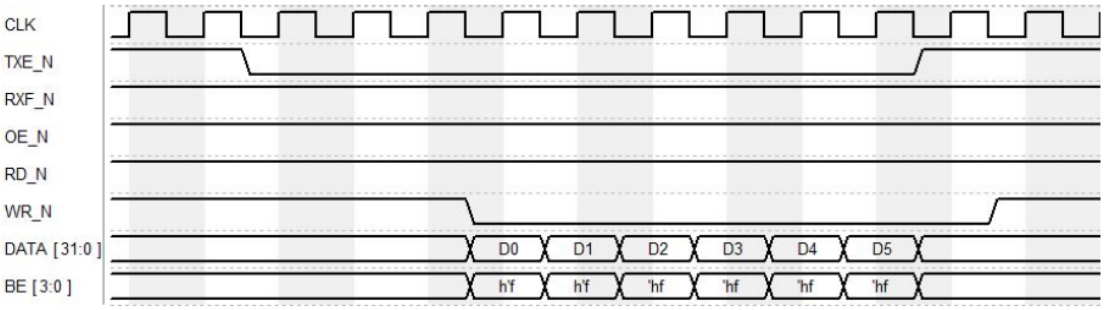


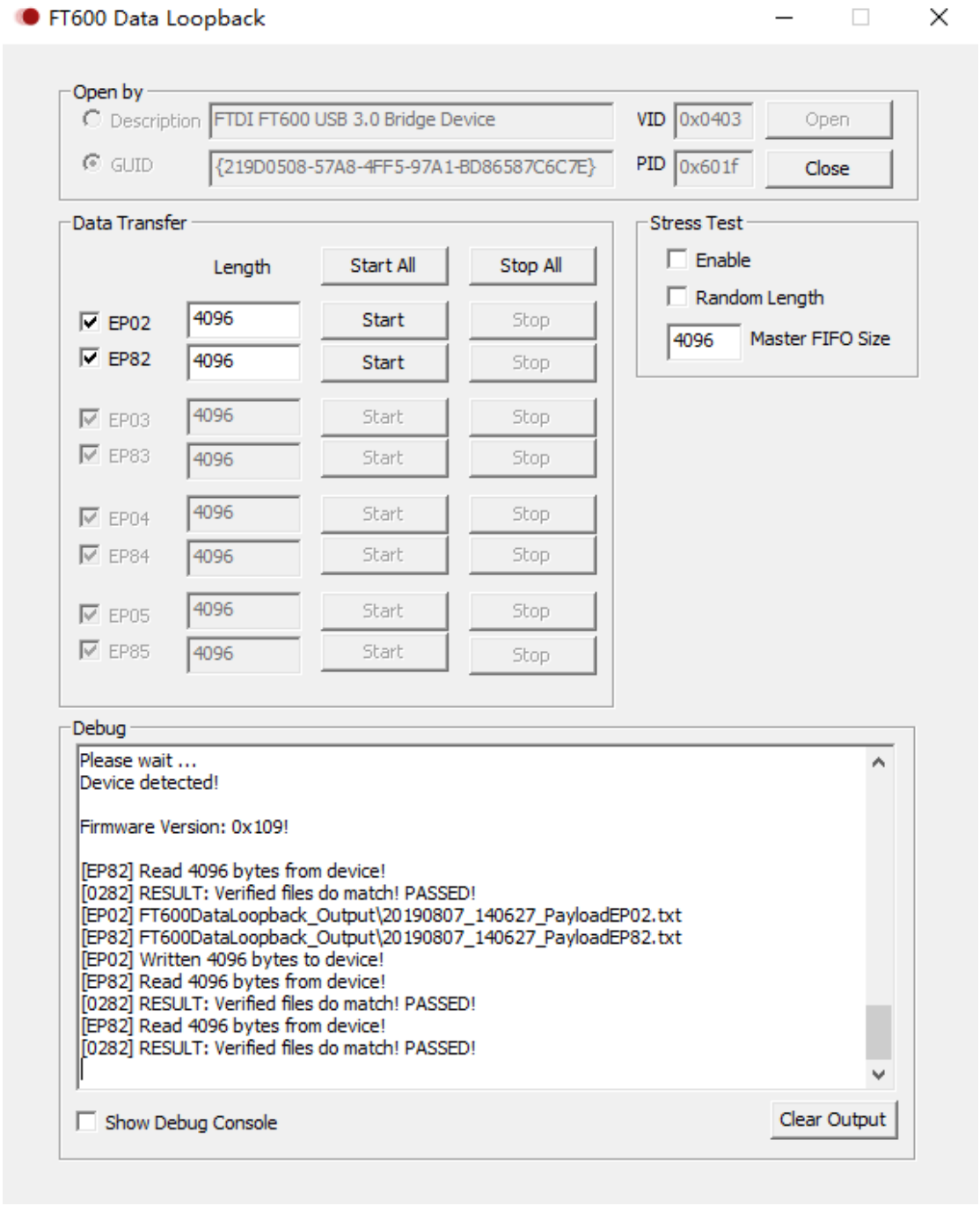
Figure 4.7 245 Synchronous FIFO mode bus master write cycle

1.2 LoopBack 测试

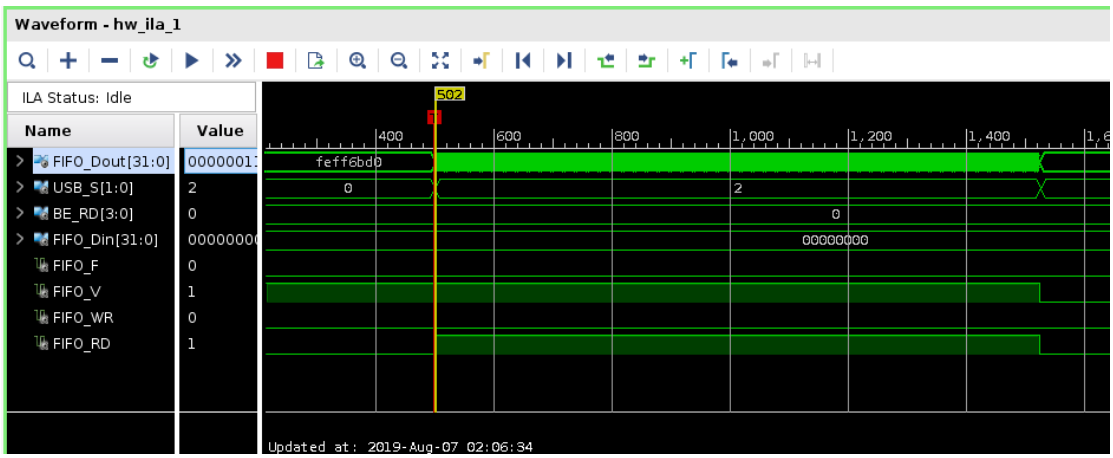
1.2.1 软件介绍

由于本文是初探 FT601Q 芯片方案，并未涉及上位机软件的编写，当然我们后面肯定会涉及上位机软件编写。这里用官方的测试程序进行测试，主要用到 2 个测试程序，分别是 FT600 Data Loopback.exe 和 FT600 Data Streamer Application.exe

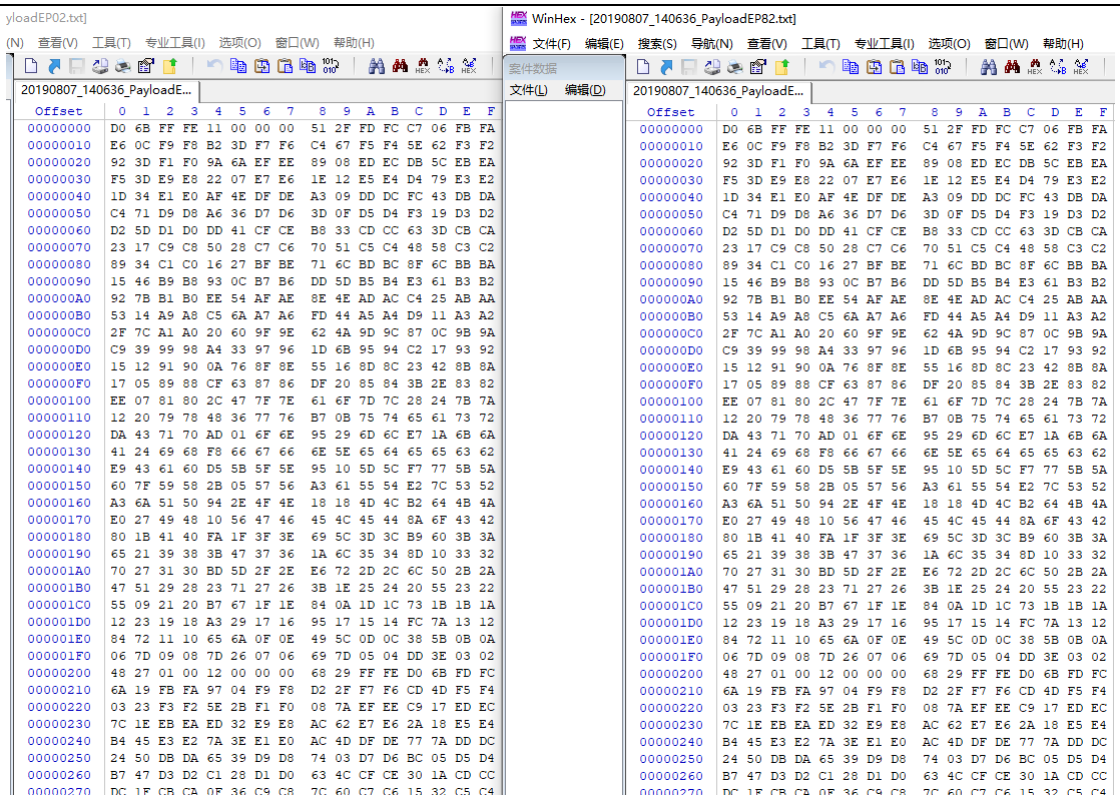
首先介绍 Loopback 方式的测试



通过在线逻辑分析仪查看数据



用 winhex 查看数据



1.2.2 245 模式 loopback 的 FPGA 代码

```
module ft60x_top(
// system control
input                   Rstn_i,//fpga reset
output                  USBSS_EN,//power enable
// FIFO interface
input                   CLK_i,
inout [31:0]            DATA_io,
inout [3:0]             BE_io,
input                   RXF_N_i,    // ACK_N
input                   TXE_N_i,
output reg              OE_N_o,
output reg              WR_N_o,     // REQ_N
output                  SIWU_N_o,
output reg              RD_N_o,
output                  WAKEUP_o,
output [1:0]            GPIO_o

);

assign USBSS_EN = 1'b1;
assign WAKEUP_o = 1'b1;
assign GPIO_o   = 2'b00;
assign SIWU_N_o = 1'b0;
```

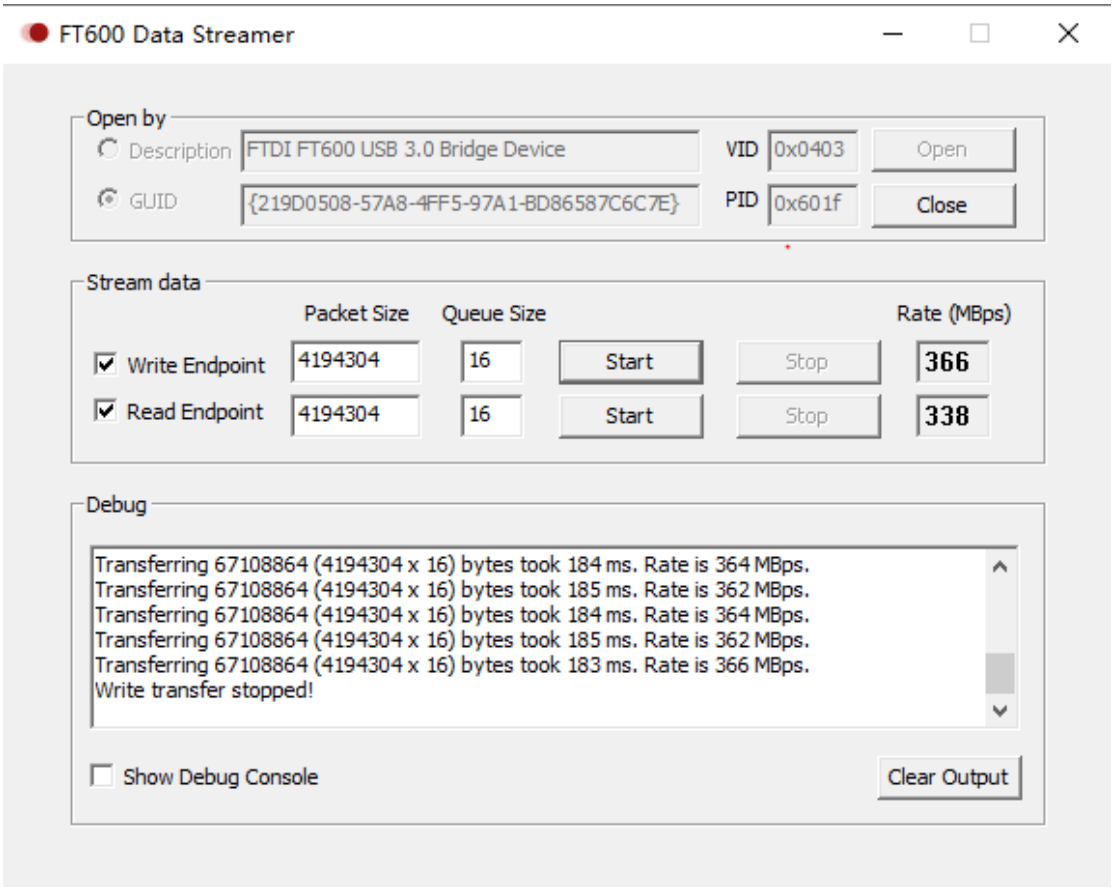
```
wire rstn;

(*mark_debug = "true"*) wire [31:0] FIFO_Din;
(*mark_debug = "true"*) wire [31:0] FIFO_Dout;
(*mark_debug = "true"*) (* KEEP = "TRUE" *) wire [3 :0] BE_RD;
(*mark_debug = "true"*) wire [ 3:0] BE_WR;
(*mark_debug = "true"*) wire FIFO_F,FIFO_V;
(*mark_debug = "true"*) reg [1:0] USB_S;
(*mark_debug = "true"*) wire FIFO_WR, FIFO_RD;

//read or write flag
assign FIFO_Din =    (USB_S==2'd1) ? DATA_io      : 32'd0;//read data dir
assign DATA_io    =    (USB_S==2'd2) ? FIFO_Dout : 32'bz;// write data dir
assign BE_RD       =    (USB_S==2'd1) ? BE_io      : 4'd0;
assign BE_io       =    (USB_S==2'd2) ? BE_WR      : 4'bz;// write data dir
assign BE_WR       =    4'b1111;

assign FIFO_WR       = (!RD_N_o)&&(!RXF_N_i);
assign FIFO_RD       = (!WR_N_o)&&(!TXE_N_i);


always @(posedge CLK_i)begin
      if(!rstn)begin
            USB_S <= 2'd0;
            OE_N_o <= 1'b1;
            RD_N_o <= 1'b1;
            WR_N_o <= 1'b1;
      end
      else begin
            case(USB_S)
            0:begin
                  OE_N_o <= 1'b1;
                  RD_N_o <= 1'b1;
                  WR_N_o <= 1'b1;
                  if((!RXF_N_i)) begin
                        USB_S    <= 2'd1;
                        OE_N_o <= 1'b0;
                  end
                  else if(!TXE_N_i)begin
                        USB_S    <= 2'd2;
                  end
            end
            1:begin
                  RD_N_o <= 1'b0;
                  if(RXF_N_i) begin
                        USB_S    <= 2'd0;
                        RD_N_o <= 1'b1;
```

```
                    OE_N_o <= 1'b1;
                end
            end
            2:begin
                WR_N_o <= 1'b0;
                if(TXE_N_i) begin
                    USB_S    <= 2'd0;
                    WR_N_o <= 1'b1;
                 end
            end
            3:begin
                USB_S <= 2'd0;
            end
            endcase
        end
end


// fifo master
fifo_generator_0 fifo_inst (
   .clk(CLK_i),           // input wire clk
   .srst(!rstn),       // input wire srst
   .din(FIFO_Din),           // input wire [15 : 0] din
   .wr_en(FIFO_WR),    // input wire wr_en
   .rd_en(FIFO_RD),    // input wire rd_en
   .dout(FIFO_Dout),       // output wire [15 : 0] dout
   .full(FIFO_F),       // output wire full
  // .empty(FIFO_E),    // output wire empty
   .valid(FIFO_V)    // output wire valid
);


Delay_rst #(
      .num(20'hffff0)
)
Delay_rst_inst
(
      .clk_i(CLK_i),
      .rstn_i(Rstn_i),
      .rst_o(rstn)
  );

endmodule
```

1.3 Stream 测试

1.3.1 软件介绍

再说 stream 模式，FT600 Data Streamer Application.exe 程序可以完成 stream 流测试，可以看到写的速度达到了 366MB/S，而读速度达到了 338MB/S 这个传输速度还是非常不错的。



1.3.2 245 模式 Stream 模式 FPGA 代码

```
module ft60x_top(
// system control
input                   Rstn_i,//fpga reset
output                  USBSS_EN,//power enable
// FIFO interface
input                   CLK_i,
inout [31:0]            DATA_io,
inout [3:0]             BE_io,
input                   RXF_N_i,    // ACK_N
input                   TXE_N_i,
output reg              OE_N_o,
output reg              WR_N_o,     // REQ_N
output                  SIWU_N_o,
output reg              RD_N_o,
output                  WAKEUP_o,
```

```
output [1:0]                GPIO_o

);

assign USBSS_EN = 1'b1;
assign WAKEUP_o = 1'b1;
assign GPIO_o     = 2'b00;
assign SIWU_N_o = 1'b0;

wire rstn;

(*mark_debug = "true"*) wire [31:0] rd_data;
(*mark_debug = "true"*) wire [31:0] wr_data;
(*mark_debug = "true"*) (* KEEP = "TRUE" *) wire [3 :0] BE_RD;
(*mark_debug = "true"*) wire [ 3:0] BE_WR;
(*mark_debug = "true"*) reg [1:0] USB_S;

//read or write flag
assign rd_data    =    (USB_S==2'd1) ? DATA_io : 32'd0;//read data dir
assign DATA_io    =    (USB_S==2'd2) ? wr_data : 32'bz;// write data dir
assign BE_RD      =    (USB_S==2'd1) ? BE_io     : 4'd0;
assign BE_io      =    (USB_S==2'd2) ? BE_WR     : 4'bz;// write data dir
assign BE_WR      =    4'b1111;


reg [7:0]wr_cnt;

assign wr_data = {wr_cnt,wr_cnt,wr_cnt,wr_cnt};

always @(posedge CLK_i)begin
      if(!rstn)begin
            wr_cnt <= 8'd0;
      end
      else if(WR_N_o) begin
            wr_cnt <= wr_cnt + 1'b1;
      end
end

always @(posedge CLK_i)begin
      if(!rstn)begin
            USB_S <= 2'd0;
            OE_N_o <= 1'b1;
            RD_N_o <= 1'b1;
            WR_N_o <= 1'b1;
      end
      else begin
            case(USB_S)
            0:begin
```

```
                OE_N_o <= 1'b1;
                RD_N_o <= 1'b1;
                WR_N_o <= 1'b1;
                if((!RXF_N_i)) begin
                    USB_S    <= 2'd1;
                    OE_N_o <= 1'b0;
                end
                else if(!TXE_N_i)begin
                    USB_S    <= 2'd2;
                end
            end
            1:begin
                RD_N_o <= 1'b0;
                if(RXF_N_i) begin
                    USB_S    <= 2'd0;
                    RD_N_o <= 1'b1;
                    OE_N_o <= 1'b1;
                end
            end
            2:begin
                WR_N_o <= 1'b0;
                if(TXE_N_i) begin
                    USB_S    <= 2'd0;
                    WR_N_o <= 1'b1;
                 end
            end
            3:begin
                USB_S <= 2'd0;
            end
            endcase
        end
end


Delay_rst #(
    .num(20'hfffff0)
)
Delay_rst_inst
(
    .clk_i(CLK_i),
    .rstn_i(Rstn_i),
    .rst_o(rstn)
 );


endmodule
```

CH02 USB3.0 通信入门 QT 上位机简单读写 FPGA

软件版本：VIVADO2019.2
操作系统：WIN10 64bit
硬件平台：适用XILINX 7系列FPGA(没有USB3.0接口的开发板需要购买FEP USB3.0子卡)
米联客(MSXBO)技术社区： www.uisrc.com 欢迎大家给我提问！！

2.1 概述

在我们完成“初探 USB3.0 极简方案 FT601Q 芯片方案”后，我们需要编写自己的上位机软件，实现对 FPGA 的访问。本文中，笔者编写了一个简单的应用程序，可以实现对开发板上 LED 的控制，以及读取开发板按键值。

A703_35T 的 IO 没有引出至底板 USB3.0 接口，而使用相同底板的 A703_100T 的 IO 已经引出至底板 USB3.0 接口。

本 A703_35T 的例子中使用的是 FEP 接口外接 USB601Q 子卡。

2.2 软件环境搭建

除了安装必要的 VIVADO 等开发 FPGA 的软件，我们开发上位机还需要安装 VS2015 和 QT 软件。

2.2.1 安装 VS2015

首先，安装 VS2015,如果你的电脑已经安装 VS2010 或者其他版本的 VS 软件，建议先卸载，卸载的时候不要用 360 等软件去卸载，用 VS 自带的卸载工具卸载，否则可能程序安装出问题，导致使用的时候不正常。

这里一定要选择自定义，默认情况下是不安装 VC++语言的，不知道 VS 是怎么搞的。

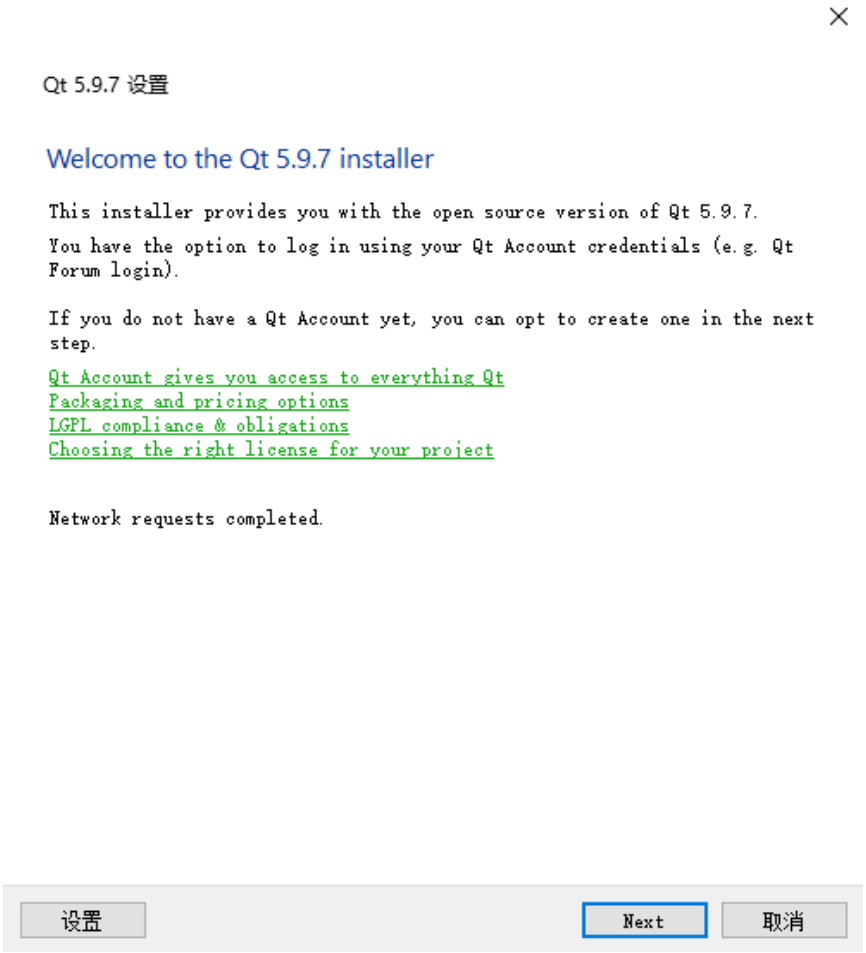

Visual Studio
Enterprise 2015
选择安装位置
C:\Program Files (x86)\Microsoft Visual Studio 14.0
...
安装程序在所有驱动器中最多需要 10 GB 空间。
选择安装类型
默认值(D)
Includes C#/VB Web and Desktop features
自定义(U)
允许自定义安装功能
安装后，你可通过“控制面板”中的“程序和功能”随时添加或删除其他功能。
单击"下一步"按钮，即表示我同意接受许可条款和隐私声明。
取消(C)
下一步(N)

注意，在编程语言中勾选 Visual C++ 另外把 Visual Studio 2015 更新 3 也勾选上。

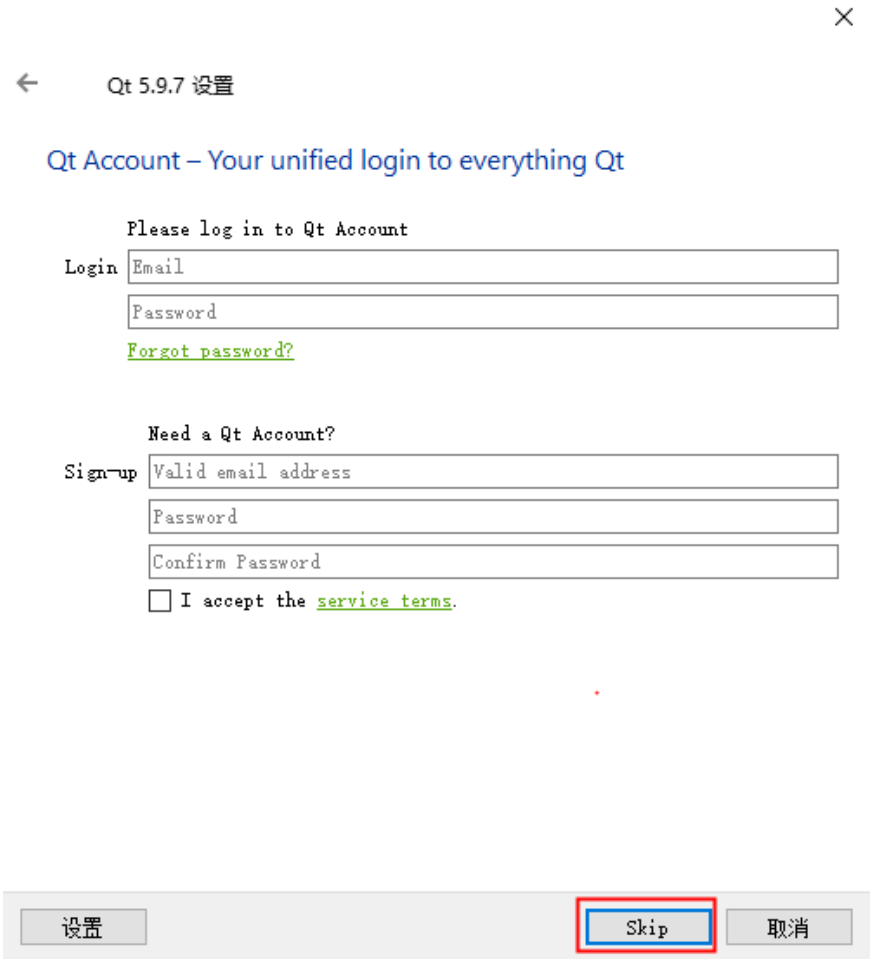
2.2.2 安装 QT

qt-opensource-windows-x86-5.9.7.exe

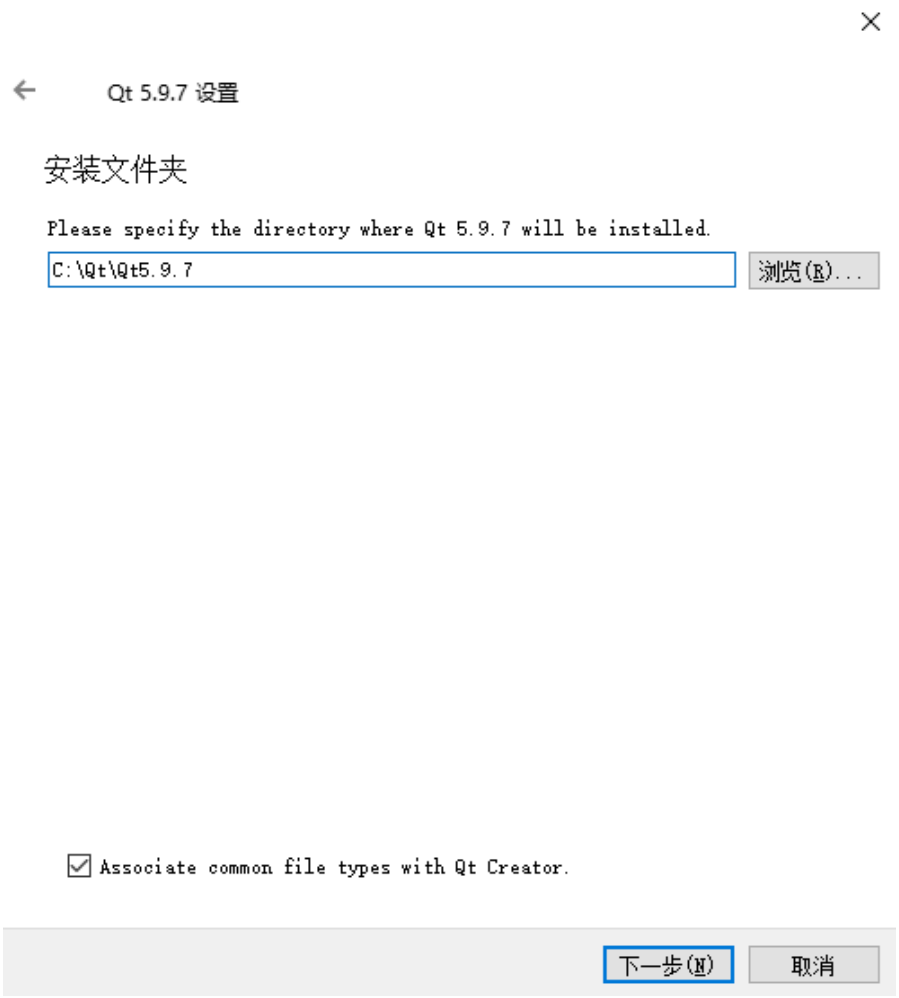
双击程序开始安装


Qt 5.9.7 设置
Welcome to the Qt 5.9.7 installer
This installer provides you with the open source version of Qt 5.9.7.
You have the option to log in using your Qt Account credentials (e.g. Qt Forum login).
If you do not have a Qt Account yet, you can opt to create one in the next step.
Qt Account gives you access to everything Qt
Packaging and pricing options
LGPL compliance & obligations
Choosing the right license for your project
Network requests completed.
设置
Next
取消

NEXT 继续安装

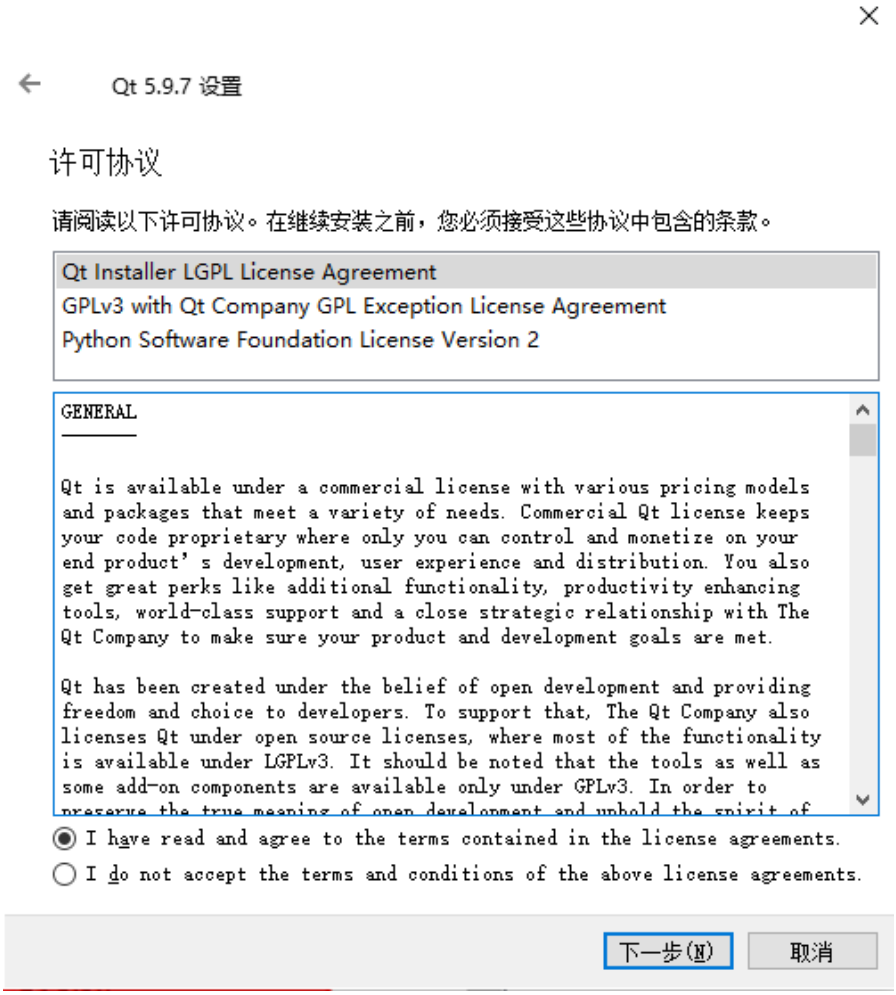

Qt 5.9.7 设置
Qt Account – Your unified login to everything Qt
Please log in to Qt Account
Login
Email
Password
Forgot password?
Need a Qt Account?
Sign-up
Valid email address
Password
Confirm Password
I accept the service terms.
设置
Skip
取消

这个地方可以直接 Skip 跳过



选择安装路径，一般默认 C 盘

选择 MSVC2015 32-bit 和 MSVC2015 64-bit 和我们安装的 VS2015 匹配



继续点击下一步，直到安装完成

打开我们提供的测试程序，需要重新指定路径，如下图

2.3 FPGA 程序设计

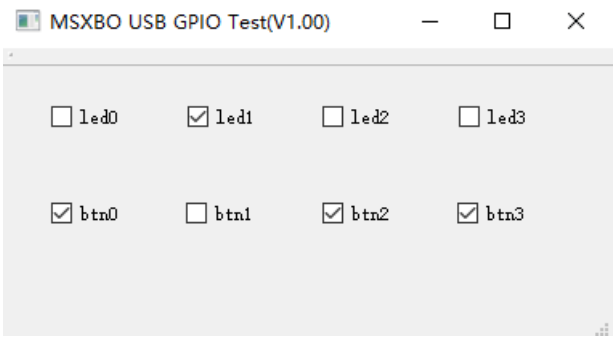
USB3.0 FPGA 程序设计,在以下程序中，我们实现上位机发送的数据控制 LED,以及上位机读取按键数据。

```
module ft60x_top(
// system control
input                   Rstn_i,//fpga reset
output                  USBSS_EN,//power enable
// FIFO interface
input                   CLK_i,
inout [31:0]            DATA_io,
inout [3:0]             BE_io,
input                   RXF_N_i,    // ACK_N
input                   TXE_N_i,
output                  OE_N_o,
output                  WR_N_o,     // REQ_N
output                  SIWU_N_o,
output                  RD_N_o,
output                  WAKEUP_o,
output [1:0]            GPIO_o,
// led
output reg              [3:0]LED,
input                   [3:0]BTN
);

assign USBSS_EN = 1'b1;
assign WAKEUP_o = 1'b1;
assign GPIO_o   = 2'b00;
assign SIWU_N_o = 1'b0;

wire rstn;
(*mark_debug = "true"*) wire             RXF_N_i;    // ACK_N
(*mark_debug = "true"*) wire             TXE_N_i;
(*mark_debug = "true"*) reg              OE_N_o;
(*mark_debug = "true"*) reg              WR_N_o;     // REQ_N
(*mark_debug = "true"*) reg              RD_N_o;

(*mark_debug = "true"*) (* KEEP = "TRUE" *)wire [31:0] rd_data;
(*mark_debug = "true"*) (* KEEP = "TRUE" *)wire [31:0] wr_data;
(*mark_debug = "true"*) wire [3 :0] BE_RD;
(*mark_debug = "true"*) wire [3 :0] BE_WR;
(*mark_debug = "true"*) reg [1:0] USB_S;

wire data_rd_valid,data_wr_valid;
assign data_rd_valid = (RD_N_o==1'b0)&&(RXF_N_i==1'b0);
assign data_wr_valid = (WR_N_o==1'b0)&&(TXE_N_i==1'b0);
//read or write flag
assign rd_data  =   data_rd_valid ? DATA_io : 32'd0;//read data dir
```

```
assign DATA_io    =    data_wr_valid ? wr_data : 32'bz;// write data dir
assign BE_RD      =    data_rd_valid ? BE_io     : 4'd0; //read data dir
assign BE_io      =    data_wr_valid ? BE_WR     : 4'bz;// write data dir
assign BE_WR      =    4'b1111;

assign wr_data = {8'h00,8'haa,8'hff,4'h0,BTN};

(*mark_debug = "true"*) (* KEEP = "TRUE" *) reg [15:0]rd_cnt;

always @(posedge CLK_i)begin
      if(!rstn)begin
            rd_cnt <= 16'd0;
      end
      else if(data_rd_valid) begin
            rd_cnt <= rd_cnt + 1'b1;
      end
      else begin
          rd_cnt    <= 16'd0;
      end
end


always @(posedge CLK_i)begin
      if(!rstn)begin
            LED <= 4'b0000;
      end
      else if(data_rd_valid) begin
            LED <= rd_data[3:0];
      end
end


always @(posedge CLK_i)begin
      if(!rstn)begin
            USB_S <= 2'd0;
            OE_N_o <= 1'b1;
            RD_N_o <= 1'b1;
            WR_N_o <= 1'b1;
      end
      else begin
            case(USB_S)
            0:begin
                  OE_N_o <= 1'b1;
                  RD_N_o <= 1'b1;
                  WR_N_o <= 1'b1;
                  if((!RXF_N_i)) begin
                        USB_S    <= 2'd1;
                        OE_N_o <= 1'b0;
```

```
                end
                else if(!TXE_N_i)begin
                    USB_S    <= 2'd2;
                end
            end
            1:begin
                RD_N_o <= 1'b0;
                if(RXF_N_i) begin
                    USB_S    <= 2'd0;
                    RD_N_o <= 1'b1;
                    OE_N_o <= 1'b1;
                end
            end
            2:begin
                WR_N_o <= 1'b0;
                if(TXE_N_i) begin
                    USB_S    <= 2'd0;
                    WR_N_o <= 1'b1;
                 end
            end
            3:begin
                USB_S <= 2'd0;
            end
            endcase
        end
end


Delay_rst #(
    .num(20'hffff0)
)
Delay_rst_inst
(
    .clk_i(CLK_i),
    .rstn_i(Rstn_i),
    .rst_o(rstn)
 );
```

2.4 人机界面



2.5 上位机分析

2.5.1 Mainwindow.c

此程序中关键就打开 USB 设备，以及发送写数据操作和读数据操作

1)、UsbManager::getInstance();中会调用并且打开 USB 设备

2)、由于 GPIO 控制不需要非常高的速度，因此定时每 100ms 用定时器更新一次数据

3)、在定时器 read_one_time 函数中首先把需要设置的 LED 值发送给 USB 设备，FPGA 会根据读取到的数据设置 LED 的状态；然后再读取按键的状态，之后更新上位机按键的显示状态

```
#include "mainwindow.h"
#include "ui_mainwindow.h"
#include <QMessageBox>
#include "usb_fun.h"

uint32_t m_io_val;

MainWindow::MainWindow(QWidget *parent) :
    QMainWindow(parent),
    ui(new Ui::MainWindow)
{

    ui->setupUi(this);
    this->setWindowTitle(tr("MSXBO USB3.0 GPIO Test(V1.00)"));

    if(usb_init()<0)
    {
        QMessageBox::information(this,"ERROR","usb init error");
    }
    m_io_val=0;
    m_QTimer_btn = new QTimer(this);
    connect(m_QTimer_btn,SIGNAL(timeout()),this,SLOT(read_one_time()));
    m_QTimer_btn->start(100);
}
```

```
MainWindow::~MainWindow()
{
    usb_deinit();
    delete ui;
}

void MainWindow::read_one_time()
{
    uint32_t val;
    read_control(&val);

    if(val&0x1) ui->ch_btn_0->setCheckState(Qt::Unchecked);
      else ui->ch_btn_0->setCheckState(Qt::Checked);

    if(val&0x2) ui->ch_btn_1->setCheckState(Qt::Unchecked);
      else ui->ch_btn_1->setCheckState(Qt::Checked);

    if(val&0x4) ui->ch_btn_2->setCheckState(Qt::Unchecked);
      else ui->ch_btn_2->setCheckState(Qt::Checked);

    if(val&0x8) ui->ch_btn_3->setCheckState(Qt::Unchecked);
      else ui->ch_btn_3->setCheckState(Qt::Checked);

}

void MainWindow::on_ch_led_0_stateChanged(int arg1)
{
    if(arg1==Qt::Checked) m_io_val|=0x1;
    else m_io_val&=0xe;
    write_control(m_io_val);
}

void MainWindow::on_ch_led_1_stateChanged(int arg1)
{
    if(arg1==Qt::Checked) m_io_val|=0x2;
    else m_io_val&=0xd;
    write_control(m_io_val);

}

void MainWindow::on_ch_led_2_stateChanged(int arg1)
{
    if(arg1==Qt::Checked) m_io_val|=0x4;
    else m_io_val&=0xb;
    write_control(m_io_val);
}

void MainWindow::on_ch_led_3_stateChanged(int arg1)
```

```
{
    if(arg1==Qt::Checked) m_io_val|=0x8;
    else m_io_val&=0x7;
    write_control(m_io_val);
}
```

2.5.2 usb_fun.c

在此程序中实现了对 USB 设备的打开，读写，这里注意驱动 inf 文件里面的 gui 需要和程序里面的 GUID 一致，否作无法正常打开设备。

```
#ifdef __cplusplus
  extern "C" {
  #endif
#include <Windows.h>
#include <assert.h>
#include <stdlib.h>
#include <stdio.h>
#include <strsafe.h>
#include <stdint.h>
#include <SetupAPI.h>
#include <INITGUID.H>
#include <WinIoCtl.h>
//#include <AtlBase.h>
#include <io.h>
#include "FTD3XX.h"

//#pragma comment(lib, "../pcie_speed/FTD3XX.lib")

DEFINE_GUID(GUID_DEVINTERFACE_FOR_FT_DEVICES,
        0x36fc9e60, 0xc465, 0x11cf, 0x80, 0x56, 0x44, 0x45, 0x53, 0x54, 0x00, 0x00);

static unsigned char *h2c_align_mem_tmp;
static unsigned char *c2h_align_mem_tmp;

static int cnt1=0;
static int cnt2=0;

static FT_HANDLE ftHandle;
static FT_STATUS ftStatus;



static int verbose_msg(const char* const fmt, ...) {

    int ret = 0;
    va_list args;
    if (1) {
```

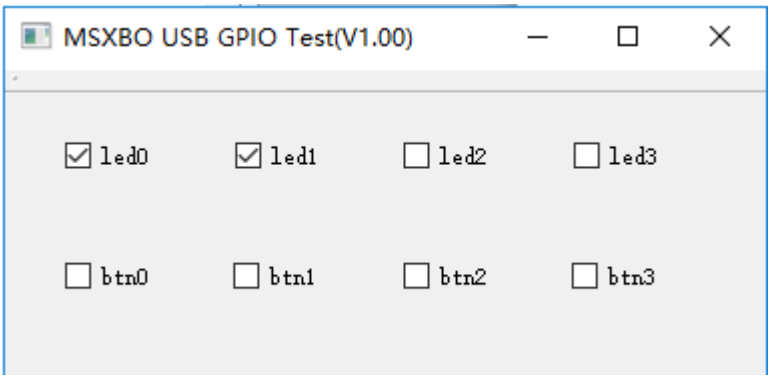
```
            va_start(args, fmt);
            ret = vprintf(fmt, args);
            va_end(args);
      }
      return ret;

}
static BYTE* allocate_buffer(size_t size, size_t alignment) {

      if(size == 0) {
            size = 4;
      }

      if (alignment == 0) {
            SYSTEM_INFO sys_info;
            GetSystemInfo(&sys_info);
            alignment = sys_info.dwPageSize;
            //printf("alignment = %d\n",alignment);
      }
      verbose_msg("Allocating host-side buffer of size %d, aligned to %d bytes\n", size, alignment);
      return (BYTE*)_aligned_malloc(size, alignment);

}

static void open_devices( int dev )
{
      //FT_INVALID_HANDLE
            USHORT uwVID = 0;
            USHORT uwPID = 0;

            GUID DeviceGUID[2] = { 0 };
            memcpy(&DeviceGUID[0], &GUID_DEVINTERFACE_FOR_FT_DEVICES, sizeof(GUID));
            ftStatus = FT_Create(&DeviceGUID[0], FT_OPEN_BY_GUID, &ftHandle);

            if (FT_FAILED(ftStatus))
            {
                  verbose_msg("open fail");
                  return ;
            }

            ftStatus = FT_GetVIDPID(ftHandle, &uwVID, &uwPID);
            if (FT_FAILED(ftStatus))
            {
                  FT_Close(ftHandle);
            }
}

static BOOL read_device( UCHAR* buff, ULONG *length )
```

```
{
	if (ftStatus != FT_OK)
	{
		return FALSE;
	}
	unsigned long readLen;
	ftStatus = FT_ReadPipe(ftHandle, 0x82, buff, *length, &readLen, NULL);

	BOOL result = TRUE;
	if (FT_FAILED(ftStatus) || readLen != *length)
	{
		result = FALSE;
	}
	*length = readLen;
	return result;
}

static BOOL write_device( UCHAR *data, ULONG *length )
{
	if (ftStatus != FT_OK)
	{
		return FALSE;
	}
	unsigned long writenLen;

	ftStatus = FT_WritePipe(ftHandle, 0x02, data, *length, &writenLen, NULL);

	BOOL result = TRUE;
	if (FT_FAILED(ftStatus) || writenLen != *length)
	{
		result = FALSE;
	}
	return result;
}

void read_control(uint32_t *val)
{
	ULONG len = 4;
	read_device((UCHAR *)val,&len);
}

void write_control(uint32_t val)
{
	ULONG len = 4;
	write_device((UCHAR *)&val,&len);
}
```

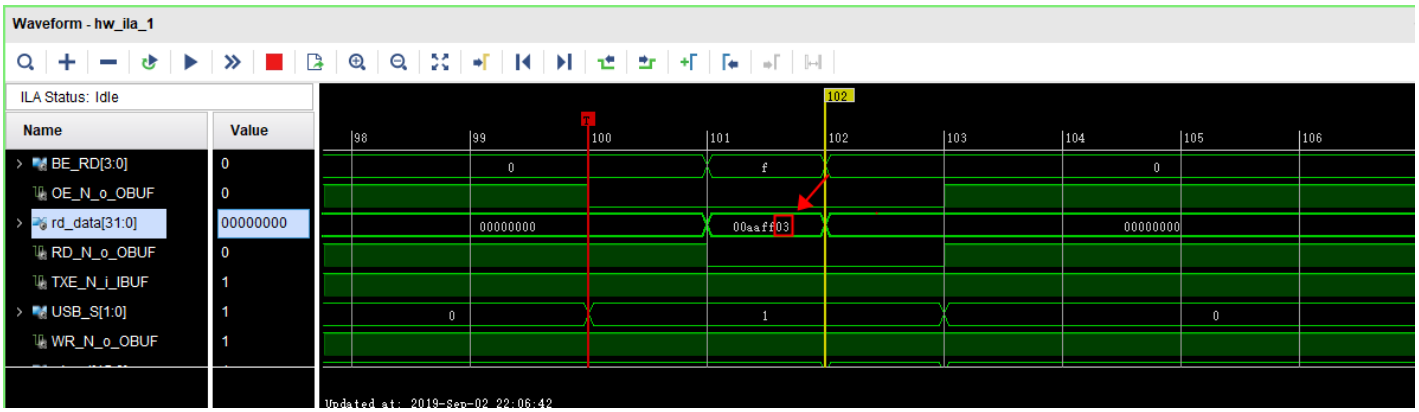
```
int usb_init()
{
	int res = 1;
	open_devices(0);
	h2c_align_mem_tmp = allocate_buffer(0x800000,4096);
	c2h_align_mem_tmp = allocate_buffer(0x800000,4096);
	if(NULL == h2c_align_mem_tmp || NULL == c2h_align_mem_tmp)
		return 0;
	return res;
}
void usb_deinit()
{
	FT_Close(ftHandle);
}


#ifdef __cplusplus
}
#endif
```

2.6 测试结果

通过 FPGA 内嵌的逻辑分析仪查看数据，点亮 LED0 LED1



在线逻辑分析抓到的数据



CH03 USB3.0 方案 FT601Q 测速

软件版本：VIVADO2019.2
操作系统：WIN10 64bit
硬件平台：适用XILINX 7系列FPGA(没有USB3.0接口的开发板需要购买FEP USB3.0子卡)
米联客(MSXBO)技术社区： www.uisrc.com 欢迎大家给我提问！！

3.1 概述

Setp by Step FT601Q 的驱动支持同步模式 245 模式，异步模式 245 模式，以及同步模式 Stream 方式，和异步模式 Stream 方式，我们这里以比较简单的同步模式 245 模式来实现对 USB3.0 FT601Q 方案测速，当然这种测速的速度肯定比不上 Stream 模式的正确。如果有时间我们也会给出 Stream 的模式方案。对于 FPGA 工程师上位机要写的比较好确实比较难，但是软件工程师，这个都是小意思。

这个例子中我们还简化了 USB 接口函数的设计，更加方便使用。

A703_35T 的 IO 没有引出至底板 USB3.0 接口，而使用相同底板的 A703_100T 的 IO 已经引出至底板 USB3.0 接口。

本 A703_35T 的例子中使用的是 FEP 接口外接 USB601Q 子卡。

3.2FPGA FT601Q 程序设计

USB3.0 FPGA 程序设计还是用上一篇文章的就可以了

```
module ft60x_top(
// system control
input                   Rstn_i,//fpga reset
output                  USBSS_EN,//power enable
// FIFO interface
input                   CLK_i,
inout [31:0]            DATA_io,
inout [3:0]             BE_io,
input                   RXF_N_i,    // ACK_N
input                   TXE_N_i,
output                  OE_N_o,
output                  WR_N_o,     // REQ_N
output                  SIWU_N_o,
output                  RD_N_o,
output                  WAKEUP_o,
output [1:0]            GPIO_o,
// led
output reg              [3:0]LED,
input                   [3:0]BTN
);

assign USBSS_EN = 1'b1;
assign WAKEUP_o = 1'b1;
assign GPIO_o   = 2'b00;
assign SIWU_N_o = 1'b0;
```

```
wire rstn;
(*mark_debug = "true"*) wire                RXF_N_i;     // ACK_N
(*mark_debug = "true"*) wire                TXE_N_i;
(*mark_debug = "true"*) reg                 OE_N_o;
(*mark_debug = "true"*) reg                 WR_N_o;      // REQ_N
(*mark_debug = "true"*) reg                 RD_N_o;

(*mark_debug = "true"*) (* KEEP = "TRUE" *)wire [31:0] rd_data;
(*mark_debug = "true"*) (* KEEP = "TRUE" *)wire [31:0] wr_data;
(*mark_debug = "true"*) wire [3 :0] BE_RD;
(*mark_debug = "true"*) wire [3 :0] BE_WR;
(*mark_debug = "true"*) reg [1:0] USB_S;

wire data_rd_valid,data_wr_valid;
assign data_rd_valid = (RD_N_o==1'b0)&&(RXF_N_i==1'b0);
assign data_wr_valid = (WR_N_o==1'b0)&&(TXE_N_i==1'b0);
//read or write flag
assign rd_data    =    data_rd_valid ? DATA_io : 32'd0;//read data dir
assign DATA_io    =    data_wr_valid ? wr_data : 32'bz;// write data dir
assign BE_RD      =    data_rd_valid ? BE_io     : 4'd0; //read data dir
assign BE_io      =    data_wr_valid ? BE_WR     : 4'bz;// write data dir
assign BE_WR      =    4'b1111;

assign wr_data = {8'h00,8'haa,8'hff,4'h0,BTN};

(*mark_debug = "true"*) (* KEEP = "TRUE" *) reg [15:0]rd_cnt;

always @(posedge CLK_i)begin
      if(!rstn)begin
            rd_cnt <= 16'd0;
      end
      else if(data_rd_valid) begin
            rd_cnt <= rd_cnt + 1'b1;
      end
      else begin
          rd_cnt    <= 16'd0;
      end
end


always @(posedge CLK_i)begin
      if(!rstn)begin
            LED <= 4'b0000;
      end
      else if(data_rd_valid) begin
            LED <= rd_data[3:0];
      end
```

```
end


always @(posedge CLK_i)begin
    if(!rstn)begin
        USB_S <= 2'd0;
        OE_N_o <= 1'b1;
        RD_N_o <= 1'b1;
        WR_N_o <= 1'b1;
    end
    else begin
        case(USB_S)
        0:begin
            OE_N_o <= 1'b1;
            RD_N_o <= 1'b1;
            WR_N_o <= 1'b1;
            if((!RXF_N_i)) begin
                USB_S   <= 2'd1;
                OE_N_o <= 1'b0;
            end
            else if(!TXE_N_i)begin
                USB_S   <= 2'd2;
            end
        end
        1:begin
            RD_N_o <= 1'b0;
            if(RXF_N_i) begin
                USB_S   <= 2'd0;
                RD_N_o <= 1'b1;
                OE_N_o <= 1'b1;
            end
        end
        2:begin
            WR_N_o <= 1'b0;
            if(TXE_N_i) begin
                USB_S   <= 2'd0;
                WR_N_o <= 1'b1;
             end
        end
        3:begin
            USB_S <= 2'd0;
        end
        endcase
    end
end


Delay_rst #(
```

```
        .num(20'hfffff0)
)
Delay_rst_inst
(
        .clk_i(CLK_i),
        .rstn_i(Rstn_i),
        .rst_o(rstn)
  );
```

3.3 上位机分析

3.3.1 USB 设备接口设计

和前面的教程使用的方法不一样，我们现在重新编写了更加简易的函数调用接口。而且大家可以看到这个函数和我们之前学习的 PCIE 课程中的函数非常类似，这样我们只要稍加修改就可以把我们之前写的 PCIE 上位机应用到 USB 上位机中了。

Usb_fun.c

```
#ifdef __cplusplus
  extern "C" {
  #endif
#include <Windows.h>
#include <assert.h>
#include <stdlib.h>
#include <stdio.h>
#include <strsafe.h>
#include <stdint.h>
#include <SetupAPI.h>
#include <INITGUID.H>
#include <WinIoCtl.h>
//#include <AtlBase.h>
#include <io.h>
#include "FTD3XX.h"

//#pragma comment(lib, "../pcie_speed/FTD3XX.lib")

DEFINE_GUID(GUID_DEVINTERFACE_FOR_FT_DEVICES,
        0x36fc9e60, 0xc465, 0x11cf, 0x80, 0x56, 0x44, 0x45, 0x53, 0x54, 0x00, 0x00);

static unsigned char *h2c_align_mem_tmp;
static unsigned char *c2h_align_mem_tmp;

static int cnt1=0;
static int cnt2=0;

static FT_HANDLE ftHandle;
static FT_STATUS ftStatus;
```

```
static int verbose_msg(const char* const fmt, ...) {

	int ret = 0;
	va_list args;
	if (1) {
		va_start(args, fmt);
		ret = vprintf(fmt, args);
		va_end(args);
	}
	return ret;

}
static BYTE* allocate_buffer(size_t size, size_t alignment) {

	if(size == 0) {
		size = 4;
	}

	if (alignment == 0) {
		SYSTEM_INFO sys_info;
		GetSystemInfo(&sys_info);
		alignment = sys_info.dwPageSize;
		//printf("alignment = %d\n",alignment);
	}
	verbose_msg("Allocating host-side buffer of size %d, aligned to %d bytes\n", size, alignment);
	return (BYTE*)_aligned_malloc(size, alignment);

}

static void open_devices( int dev )
{
	//FT_INVALID_HANDLE
		USHORT uwVID = 0;
		USHORT uwPID = 0;

		GUID DeviceGUID[2] = { 0 };
		memcpy(&DeviceGUID[0], &GUID_DEVINTERFACE_FOR_FT_DEVICES, sizeof(GUID));
		ftStatus = FT_Create(&DeviceGUID[0], FT_OPEN_BY_GUID, &ftHandle);

		if (FT_FAILED(ftStatus))
		{
			verbose_msg("open fail");
			return ;
		}

		ftStatus = FT_GetVIDPID(ftHandle, &uwVID, &uwPID);
```

```
        if (FT_FAILED(ftStatus))
        {
            FT_Close(ftHandle);
        }
}

static BOOL read_device( UCHAR* buff, ULONG *length )
{
    if (ftStatus != FT_OK)
    {
        return FALSE;
    }
    unsigned long readLen;
    ftStatus = FT_ReadPipe(ftHandle, 0x82, buff, *length, &readLen, NULL);

    BOOL result = TRUE;
    if (FT_FAILED(ftStatus) || readLen != *length)
    {
        result = FALSE;
    }
    *length = readLen;
    return result;
}

static BOOL write_device( UCHAR *data, ULONG *length )
{
    if (ftStatus != FT_OK)
    {
        return FALSE;
    }
    unsigned long writenLen;

    ftStatus = FT_WritePipe(ftHandle, 0x02, data, *length, &writenLen, NULL);

    BOOL result = TRUE;
    if (FT_FAILED(ftStatus) || writenLen != *length)
    {
        result = FALSE;
    }
    return result;
}


int usb_init()
{
    int res = 1;
    open_devices(0);
    h2c_align_mem_tmp = allocate_buffer(0x800000,4096);
```

```
	c2h_align_mem_tmp = allocate_buffer(0x800000,4096);
	if(NULL == h2c_align_mem_tmp || NULL == c2h_align_mem_tmp)
		return 0;
	return res;
}
void usb_deinit()
{
	FT_Close(ftHandle);
}
unsigned int h2c_transfer(unsigned int size)
{
	double bd=0;
	double time_sec;
	LARGE_INTEGER start;
	LARGE_INTEGER stop;
	LARGE_INTEGER freq;

	QueryPerformanceFrequency(&freq);
	QueryPerformanceCounter(&start);

	write_device(h2c_align_mem_tmp,&size);
	QueryPerformanceCounter(&stop);
	time_sec = (unsigned long long)(stop.QuadPart - start.QuadPart) / (double)freq.QuadPart;
	bd = ((double)size)/(double)(time_sec)/1024.0/1024.0;

	return (unsigned int)bd;
}
unsigned int c2h_transfer(unsigned int size)
{
	double bd=0;
	double time_sec;
	LARGE_INTEGER start;
	LARGE_INTEGER stop;
	LARGE_INTEGER freq;

	QueryPerformanceFrequency(&freq);
	QueryPerformanceCounter(&start);
	read_device(c2h_align_mem_tmp,&size);
	QueryPerformanceCounter(&stop);
	time_sec = (unsigned long long)(stop.QuadPart - start.QuadPart) / (double)freq.QuadPart;
	bd = ((double)size)/(double)(time_sec)/1024.0/1024.0;

	return (unsigned int)bd;
}


#ifdef __cplusplus
}
```

```
#endif
```

3.3.2 测试码表程序

为了设计绚丽的测试码表控件我们用到了以下代码，这个码表测速也是我们 PCIE 中用到的程序

myspeed.c

```
#include "myspeed.h"
#include <QPainter>
#include <qmath.h>
mySpeed::mySpeed(QWidget *parent): QWidget(parent)
{
    m_background = Qt::black;
    m_foreground = Qt::green;

    m_startAngle=60;
    m_endAngle=60;
    m_scaleMajor=10;
    m_minValue=0;
    m_maxValue=1000;
    m_scaleMajor = 10;//分度
    m_scaleMinor = 10;
    m_units = "MB/S";
    m_title = "Speed USB";
    m_precision = 2;
    m_value=0;

    m_updateTimer = new QTimer(this);
    m_updateTimer->setInterval(50);//
    connect(m_updateTimer,SIGNAL(timeout()),this,SLOT(UpdateAngle()));
      m_updateTimer->start();//

     //setWindowFlags(Qt::FramelessWindowHint);//
     //setAttribute(Qt::WA_TranslucentBackground);//
     resize(400,400);
}

mySpeed::~mySpeed()
{

}

void mySpeed::drawCrown(QPainter *painter)
{
    painter->save();
    int radius =100;
    QLinearGradient lg1(0,-radius,0,radius);
```

```
    lg1.setColorAt(0.2,Qt::white);
    lg1.setColorAt(1,Qt::black);
    painter->setBrush(lg1);
    painter->drawEllipse(-100, -100, 200 ,200);
    painter->restore();
}

void mySpeed::drawBackground(QPainter *painter)
{
    painter->save();
    painter->setBrush(m_background);
    painter->drawEllipse(-90, -90, 180 ,180);
    painter->restore();
}

void mySpeed::drawScaleNum(QPainter *painter)
{
    painter->save();
    painter->setPen(m_foreground);
    double startRad = (360 - m_startAngle - 90) * (3.14 / 180);
    double deltaRad = (360 - m_startAngle - m_endAngle) * (3.14 / 180) / m_scaleMajor;
    double sina,cosa;
    int x, y;
    QFontMetricsF fm(this->font());
    double w, h, tmpVal;
    QString str;

    for (int i = 0; i <= m_scaleMajor; i++)
    {
        sina = sin(startRad - i * deltaRad);
        cosa = cos(startRad - i * deltaRad);

       tmpVal = 1.0 * i *((m_maxValue - m_minValue) / m_scaleMajor) + m_minValue;

        str = QString( "%1" ).arg(tmpVal);
        w = fm.size(Qt::TextSingleLine,str).width();
        h = fm.size(Qt::TextSingleLine,str).height();
        x = 82 * cosa - w / 2;
        y = -82 * sina + h / 4;
        painter->drawText(x, y, str);
    }
    painter->restore();
}

void mySpeed::drawScale(QPainter *painter)
{
    painter->save();
    painter->rotate(m_startAngle);
```

```
    int steps = (m_scaleMajor * m_scaleMinor);
    double angleStep = (360.0 - m_startAngle - m_endAngle) / steps;
    painter->setPen(m_foreground);
    QPen pen = painter->pen();
    for (int i = 0; i <= steps; i++)
    {
        if (i % m_scaleMinor == 0)
        {
            pen.setWidth(1);
            painter->setPen(pen);
            painter->drawLine(0, 62, 0, 72);
        }
        else
        {
            pen.setWidth(0);
            painter->setPen(pen);
            painter->drawLine(0, 67, 0, 72);
        }
        painter->rotate(angleStep);
    }
    painter->restore();
}

void mySpeed::drawTitle(QPainter *painter)
{
    painter->save();
    painter->setPen(m_foreground);
    //painter->setBrush(m_foreground);
    QString str(m_title);
    QFontMetricsF fm(this->font());
    double w = fm.size(Qt::TextSingleLine,str).width();
    painter->drawText(-w / 2, -30, str);
    painter->restore();
}
void mySpeed::drawNumericValue(QPainter *painter)
{
    QString str    =    QString("%1 %2").arg(m_value, 0, 'f', m_precision).arg(m_units);
    QFontMetricsF fm(font());
    double w = fm.size(Qt::TextSingleLine,str).width();
    painter->setPen(m_foreground);
    painter->drawText(-w / 2, 42, str);
}

void mySpeed::drawIndicator(QPainter *painter)
{
    painter->save();
    QPolygon pts;
    pts.setPoints(3, -2,0, 2,0, 0,60);     /* (-2,0)/(2,0)/(0,60) *///
```

```
    painter->rotate(m_startAngle);
    double degRotate =    (360.0 - m_startAngle - m_endAngle)/(m_maxValue - m_minValue)*(m_value - m_minValue);


    painter->rotate(degRotate);    //
    QRadialGradient haloGradient(0, 0, 60, 0, 0);    //
    haloGradient.setColorAt(0, QColor(60,60,60));
    haloGradient.setColorAt(1, QColor(160,160,160)); //
    painter->setPen(Qt::white); //
    painter->setBrush(haloGradient);//
    painter->drawConvexPolygon(pts); //
    painter->restore();

    //画中心点
    QColor niceBlue(150, 150, 200);
    QConicalGradient coneGradient(0, 0, -90.0);    //
    coneGradient.setColorAt(0.0, Qt::darkGray);
    coneGradient.setColorAt(0.2, niceBlue);
    coneGradient.setColorAt(0.5, Qt::white);
    coneGradient.setColorAt(1.0, Qt::darkGray);
    painter->setPen(Qt::NoPen);    //
    painter->setBrush(coneGradient);
    painter->drawEllipse(-5, -5, 10, 10);
}


void mySpeed::paintEvent(QPaintEvent *)
{
    QPainter painter (this);
    painter.setRenderHint(QPainter::Antialiasing);
    painter.translate(width()/2,height()/2);
    int side = qMin(width(),height());
    painter.scale(side /200.0,side /200.0);
    drawCrown(&painter);
    drawBackground(&painter);
    drawScaleNum(&painter);

    drawScale(&painter);
    drawTitle(&painter);

    drawNumericValue(&painter);
    drawIndicator(&painter);

}

void mySpeed::UpdateAngle()
{
```

```
    //m_value += 1;
    //if(m_value > 1000)
    //{
    //      m_value = 0;
    //}

    //m_angle = ((m_angle + 1) % 360);
    update();//
}
```

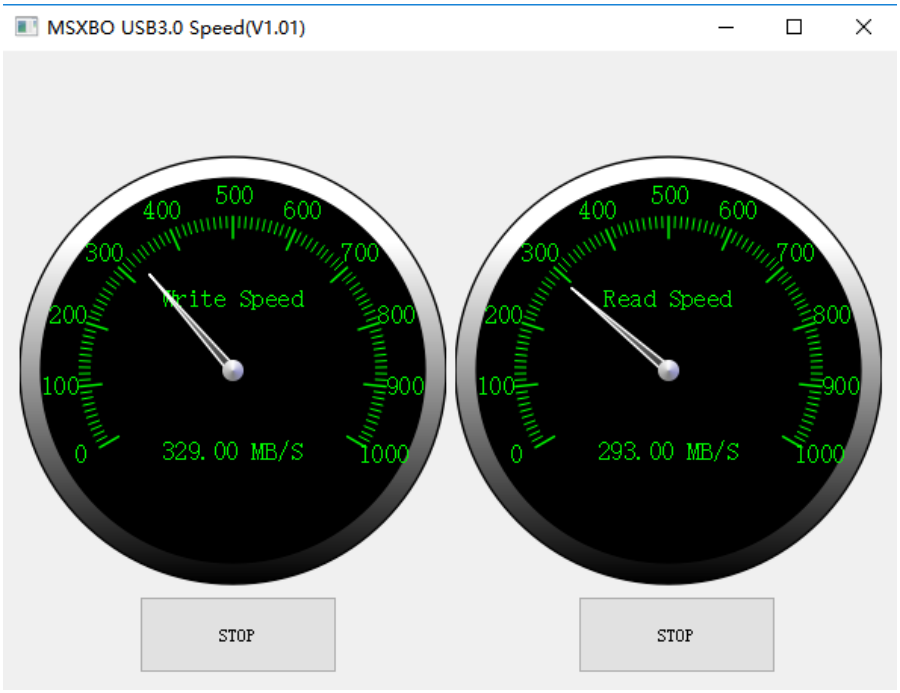
主程序中，我们会定时调用测试函数进行测速，并且把测试的值通过码表程序显示

```
#include "mainwindow.h"
#include "ui_mainwindow.h"
#include "usb_fun.h"

MainWindow::MainWindow(QWidget *parent) :
        QWidget(parent),
        ui(new Ui::MainWindow)
{
        //init ui
        QGridLayout *pUserLayout = new QGridLayout(this);
        for(int i=0;i<2;i++)
        {
                ppmySpeed[i] = new mySpeed;
                i ? ppmySpeed[i]->m_title = "Read Speed" : ppmySpeed[i]->m_title = "Write Speed" ;
                pUserLayout->addWidget(ppmySpeed[i], 0, i);
         }
        ui->setupUi(this);
        this->setWindowTitle(tr("MSXBO USB3.0 Speed(V1.01)"));
        //init usb
        if(usb_init()<0)
        {
                QMessageBox::information(this,"ERROR","usb init error");
        }
        isc2h_start = false;
        ish2c_start = false;

        c2h_transfer_size = 0x80000;
        h2c_transfer_size = 0x80000;

        m_QTimer_c2h = new QTimer(this);
        m_QTimer_h2c = new QTimer(this);
        connect(m_QTimer_c2h,SIGNAL(timeout()),this,SLOT(c2h_transfer_one_time()));
        connect(m_QTimer_h2c,SIGNAL(timeout()),this,SLOT(h2c_transfer_one_time()));

}
```

```
MainWindow::~MainWindow()
{

    usb_deinit();

    delete ui;
}
void MainWindow::c2h_transfer_one_time()
{
    unsigned int bd=0;
    bd = c2h_transfer(c2h_transfer_size);
    ppmySpeed[1]->m_value=bd;
    //QString str = QString("%1").arg(bd);
    //str += "MB/s";
    //ui->lineEdit_c2h->setText(str);
}
void MainWindow::h2c_transfer_one_time()
{
    //transfer h2c
    unsigned int bd=0;
    bd = h2c_transfer(h2c_transfer_size);
    ppmySpeed[0]->m_value=bd;
    //QString str = QString("%1").arg(bd);
    //str += "MB/s";
    //ui->lineEdit_h2c->setText(str);
}

void MainWindow::on_pushButton_h2c_start_clicked()
{
    if(false==ish2c_start)
    {
        ish2c_start = true;
        m_QTimer_h2c->start(100);
        ui->pushButton_h2c_start->setText("STOP");
    }
    else
    {
        ish2c_start = false;
        m_QTimer_h2c->stop();
        ui->pushButton_h2c_start->setText("START");
    }
}

void MainWindow::on_pushButton_c2h_start_clicked()
{
    if(false==isc2h_start)
    {
```

```
            isc2h_start = true;
            m_QTimer_c2h->start(100);
            ui->pushButton_c2h_start->setText("STOP");
        }
        else
        {
            isc2h_start = false;
            m_QTimer_c2h->stop();
            ui->pushButton_c2h_start->setText("START");
        }
}
```

3.4 测速结果



3.5 思考

USB3.0 和 PCIE2.0 的串行传输方式非常类似，速度也非常接近，但是 FT601Q 没有同时双向处理的能力，我们的程序为什么可以同时进行读写呢？显然不可能同时读写的，实际上你看到的感觉读和写同时在进行实际上，对于 FT601Q 和 FPGA 的通信来说，他们一个时刻只能是写或者读，所以如果同时读写，速度至少要降低一半的。我们这里测速只对单次读，和单次写进行了测速。具体的可以去看看代码，理解代码才是王道。

CH04 USB3.0 摄像头方案之 FT601Q

软件版本：VIVADO2019.2
操作系统：WIN10 64bit
硬件平台：适用XILINX 7系列FPGA(没有USB3.0接口的开发板需要购买FEP USB3.0子卡)
米联客(MSXBO)技术社区： www.uisrc.com 欢迎大家给我提问！！

4.1 概述

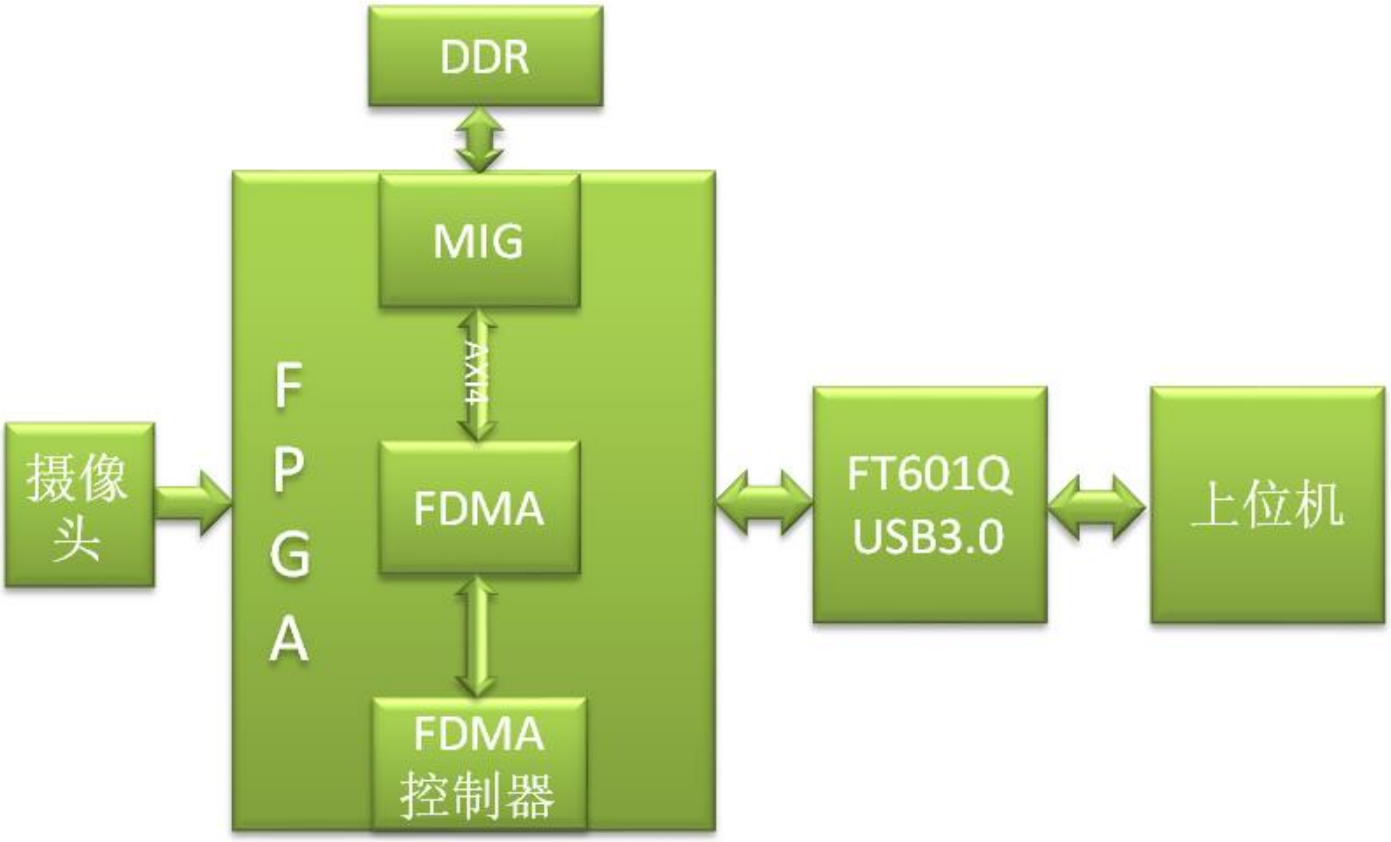
本文取名为“USB3.0 摄像头方案之 FT601Q”是因为我们还有一套“USB3.0 UVC 摄像头方案之 FT602Q”后续会发布。在完成了前面的点灯，测速后，我们主要对 FT601Q 官方的库函数使用有了深刻的了解。现在我们做一个比较实用的方案，实现摄像头的采集。当然我们本身的方案主要是用 FPGA 去采集视频，放到 DDR 缓存后，上位机读取缓存里面的视频数据，在电脑上显示。打通了 FPGA 视频数据到上位机后，一些做产品方案的客户可以加入自己的独家算法，实现 FPGA 的图像处理加速，然后通过上位机人机界面显示，或者做进一步的处理。

A703_35T 的 IO 没有引出至底板 USB3.0 接口，而使用相同底板的 A703_100T 的 IO 已经引出至底板 USB3.0 接口。

本 A703_35T 的例子中使用的是 FEP 接口外接 USB601Q 子卡。

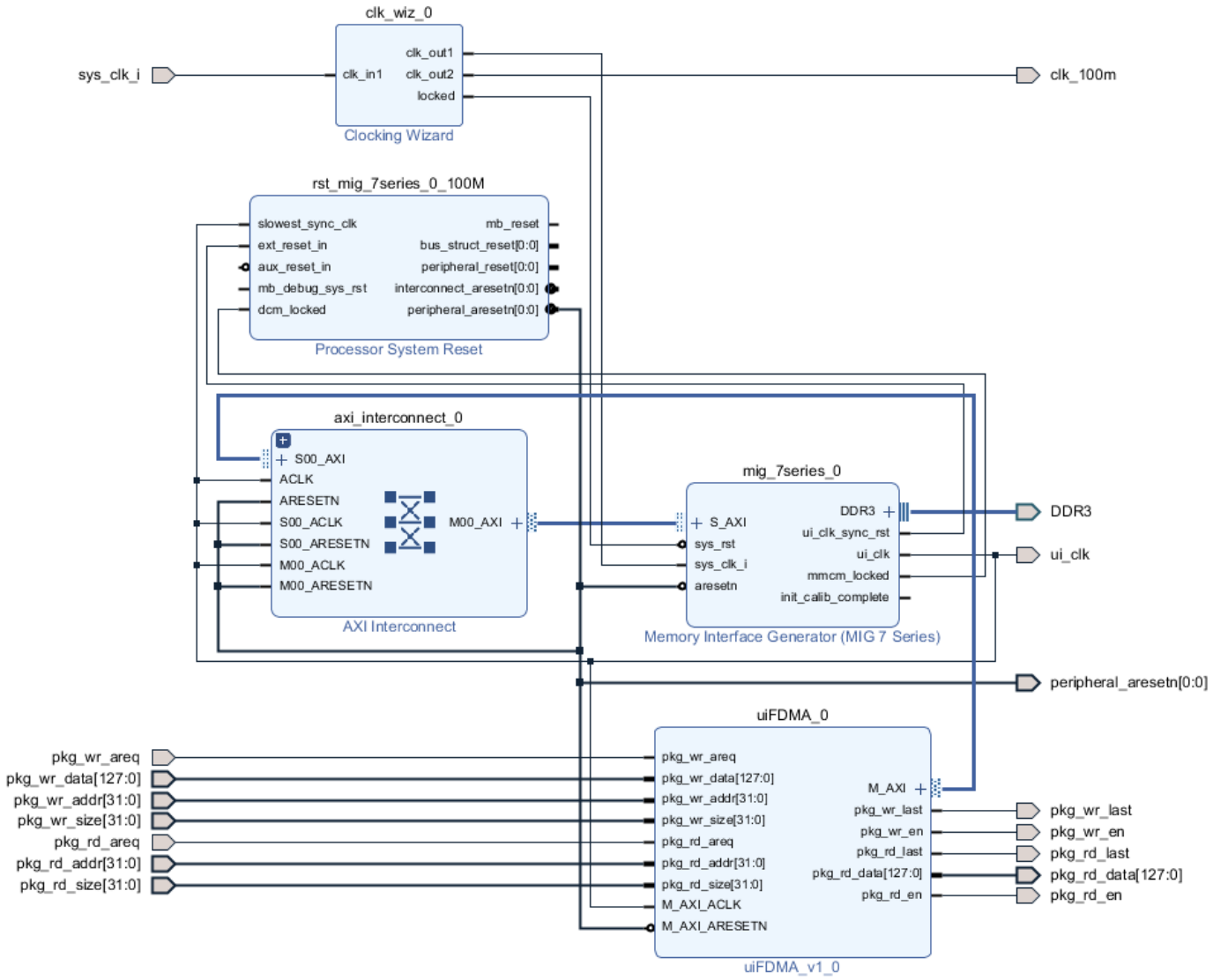
4.2 构架设计

如下图所示，对于 FDMA 是我们米联客(MSXBO 英文注册商标)自定的 AXI4 总线 IP 非常适合于图像数据的缓存应用，我们设计的 FDMA 控制器构架可以很方便的应用于视频的数据缓存。如果你用过米联客的开发板并且之前已经掌握了 FMDA IP 的使用，那么这部分内容就很简单了。因为对于 FT601Q 的 FPGA 代码非常简单。我们对 FT601Q 的 FPGA 代码做了一个简单的视频信号转换后接到了 FDMA 控制器，之前已经开发好的的代码几乎不用修改就可以使用了。这也是我一直强调的构架设计，设计一个优秀的构架，一个可重复利用的构架，可以提高我们的编程效率，以及降低代码的 bug.



4.3FPGA BD 设计

下图的 BD 设计没啥好好说的，对于第一次使用 FMDA 的人来说，主要是了解 FDMA 这个 IP 的使用，那就去阅读我们专门讲解 FMDA 的章节吧，因为这个 BD 工程和专门讲解 FMDA IP 章节的一样，这就是优秀代码构架的好处，不管你用于什么工程，基本上只要做很小的代码修改。



4.4 USB 接口代码

USB 接口代码依然非常简单

```
module ft60x(
// system control
input                   Rstn_i,//fpga reset
output                  USBSS_EN,//power enable
// FIFO interface
input                   CLK_i,
inout [31:0]            DATA_io,
inout [3:0]             BE_io,
input                   RXF_N_i,    // ACK_N
input                   TXE_N_i,
output                  OE_N_o,
output                  WR_N_o,     // REQ_N
output                  SIWU_N_o,
output                  RD_N_o,
output                  WAKEUP_o,
```

```
output [1:0]                GPIO_o,
output                      usb_rd_en,
output                      usb_frame,
input    [31:0]             usb_data
);

assign USBSS_EN = 1'b1;
assign WAKEUP_o = 1'b1;
assign GPIO_o     = 2'b00;
assign SIWU_N_o = 1'b0;

wire rstn;

(*mark_debug = "true"*) wire                usb_rd_en;
(*mark_debug = "true"*) wire                usb_frame;

(*mark_debug = "true"*) wire                RXF_N_i;     // ACK_N
(*mark_debug = "true"*) wire                TXE_N_i;
(*mark_debug = "true"*) reg                 OE_N_o;
(*mark_debug = "true"*) reg                 WR_N_o;      // REQ_N
(*mark_debug = "true"*) reg                 RD_N_o;

(*mark_debug = "true"*) (* KEEP = "TRUE" *)wire [31:0] rd_data;
(*mark_debug = "true"*) (* KEEP = "TRUE" *)wire [31:0] wr_data;
(*mark_debug = "true"*) wire [3 :0] BE_RD;
(*mark_debug = "true"*) wire [3 :0] BE_WR;
(*mark_debug = "true"*) reg [1:0] USB_S;

wire data_rd_valid,data_wr_valid;
assign data_rd_valid = (RD_N_o==1'b0)&&(RXF_N_i==1'b0);
assign data_wr_valid = (WR_N_o==1'b0)&&(TXE_N_i==1'b0);
//read or write flag
assign rd_data    =    (USB_S==2'd1) ? DATA_io : 32'd0;//read data dir
assign BE_RD      =    (USB_S==2'd1) ? BE_io     : 4'd0;
assign DATA_io    =    (USB_S==2'd2) ? wr_data : 32'bz;// write data dir
assign BE_io      =    (USB_S==2'd2) ? BE_WR     : 4'bz;// write data dir
assign BE_WR      =    4'b1111;

assign wr_data = usb_data;

(*mark_debug = "true"*) (* KEEP = "TRUE" *) reg [15:0]rd_cnt;

(*mark_debug = "true"*) (* KEEP = "TRUE" *) reg [31:0] fram_cmd;

always @(posedge CLK_i)begin
      if(!rstn)begin
            fram_cmd <= 32'd0;
      end
```

```
    else if(data_rd_valid) begin
        fram_cmd <= rd_data;
    end
    else begin
        if(fram_cmd> 32'd0)begin
            fram_cmd <= fram_cmd - 1'b1;
        end
    end
end

assign usb_frame = (fram_cmd> 32'd2000);
assign usb_rd_en = data_wr_valid;


always @(posedge CLK_i)begin
    if(!rstn)begin
        USB_S <= 2'd0;
        OE_N_o <= 1'b1;
        RD_N_o <= 1'b1;
        WR_N_o <= 1'b1;
    end
    else begin
        case(USB_S)
        0:begin
            OE_N_o <= 1'b1;
            RD_N_o <= 1'b1;
            WR_N_o <= 1'b1;
            if((!RXF_N_i)) begin
                USB_S   <= 2'd1;
                OE_N_o <= 1'b0;
            end
            else if((!TXE_N_i))begin
                USB_S   <= 2'd2;
            end
        end
        1:begin
            RD_N_o <= 1'b0;
            if(RXF_N_i) begin
                USB_S   <= 2'd0;
                RD_N_o <= 1'b1;
                OE_N_o <= 1'b1;
            end
        end
        2:begin
            WR_N_o <= 1'b0;
            if(TXE_N_i) begin
                USB_S   <= 2'd0;
                WR_N_o <= 1'b1;
```

```
                end
            end
            3:begin
                USB_S <= 2'd0;
            end
            endcase
        end
end


Delay_rst_0 Delay_rst_inst
(
    .clk_i(CLK_i),
    .rstn_i(Rstn_i),
    .rst_o(rstn)
 );


endmodule
```

请注意这三个信号

usb_rd_en 此信号接到 FDMA 读接口的 R0_rden_i 代表读数据使能

usb_frame 此信号接到 FDMA 读接口的 R0_FS_i 代表读数据帧信号

usb_data 此信号接到 FDMA 读接口的 R0_data_o 代表读到的有效数据

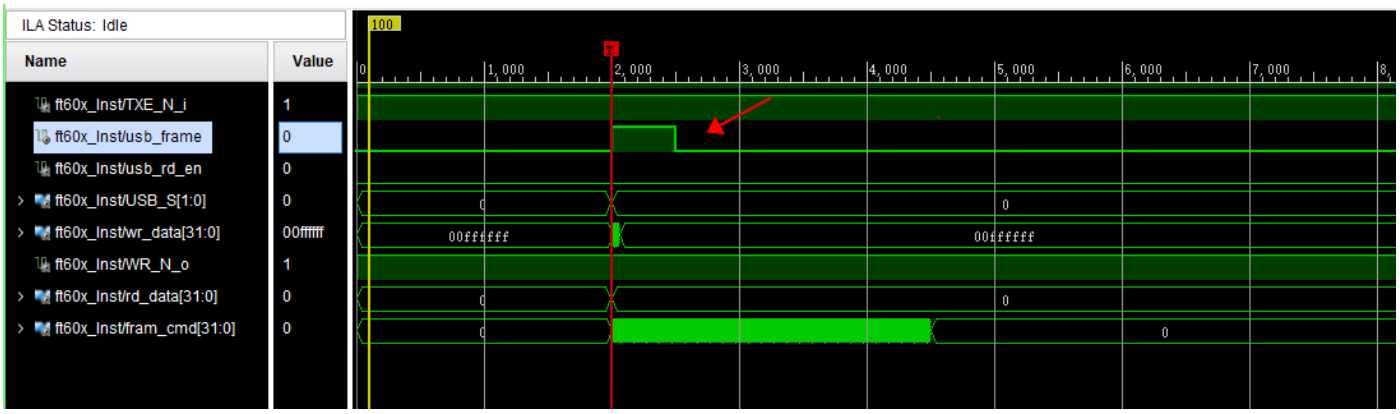
```
always @(posedge CLK_i)begin
    if(!rstn)begin
        fram_cmd <= 32'd0;
    end
    else if(data_rd_valid) begin
        fram_cmd <= rd_data;
    end
    else begin
        if(fram_cmd> 32'd0)begin
            fram_cmd <= fram_cmd - 1'b1;
        end
    end
end
assign usb_frame = (fram_cmd> 32'd2000);
assign usb_rd_en = data_wr_valid;
```

再看其中的关键设计，fram_cmd 计数器用保存上位机发过来的帧信号，这里就是用计数器值去产生读通道的帧同步信号。这个设计非常巧妙，不耗费代码，但是却实现了功能。

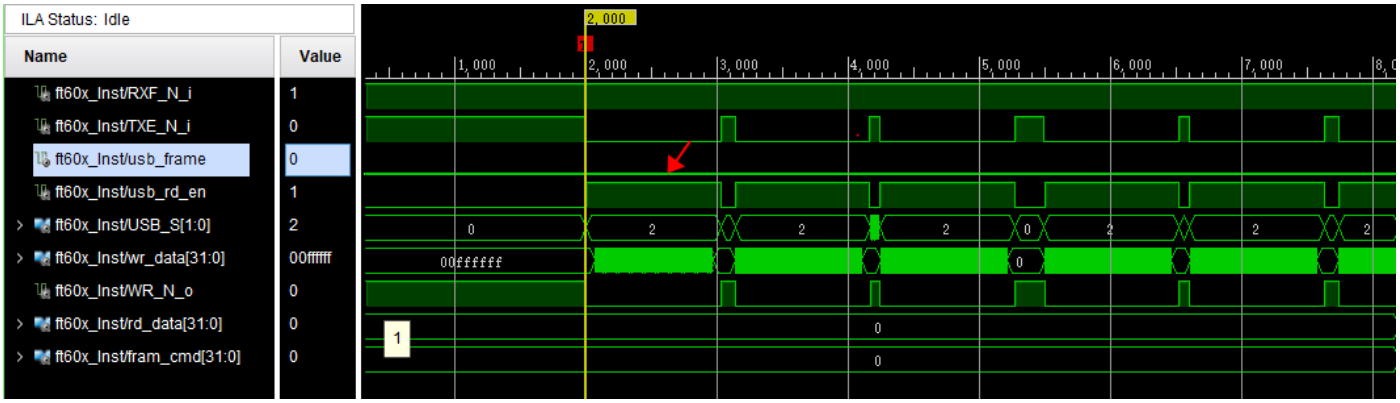
再看我们用 data_wr_valid 信号去实现了读使能，由于 USB 的数据对于 FT601Q 来说最大 4KB，每次 4KB 一包完成后，必然有个停顿，进行下次一传输，正好我们就可以用这个信号作为每次的同步读取信号，类似行同步信号。

4.5 在线逻辑分析仪观察信号

比如下图是我们抓紧到的在线逻辑分析仪观察关键信号的截图
下图是帧同步



下图可以理解未场同步，每次最大 4KB 数据

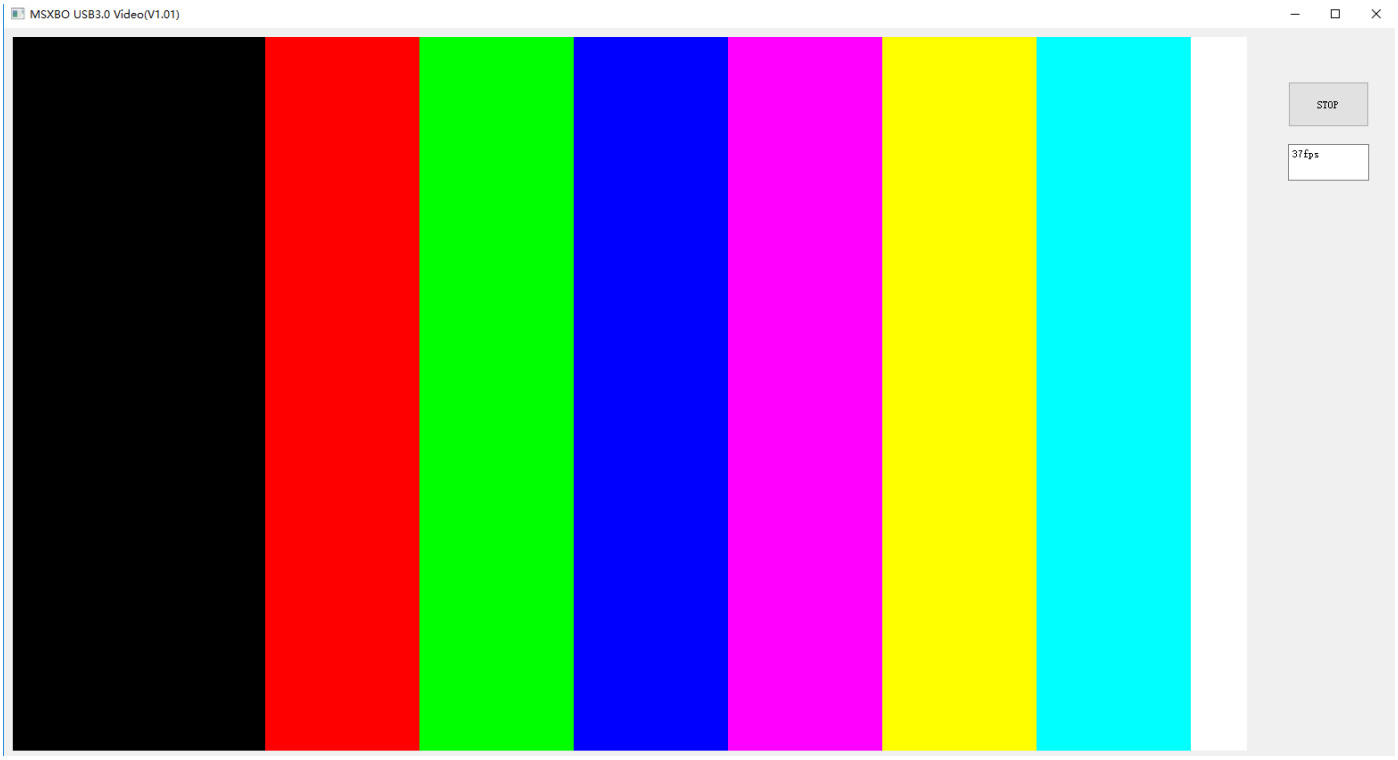


4.6、测试结果

A703_35T 资源和硬件限制，不能做摄像头采集。 本例子中使用的是彩条方格测试。

4.6.1 彩条方格测试

通过 FPGA 内部产生测试数据观察图像采集是否真确，这种方法成本最低，最有效，最有利于分析问题，如果测试图像输出没问题，基本上后面加入摄像头后问题就不大了。对于初学者，切勿急功近利，我们开发这么久了，还要注意循序渐进呢。我们这个 USB3.0 方案，从硬件筹备到目前开发的软件比较齐全都快 1 年多了。当然有了我们的方案你也许只要 1 天就搞定了。所以买一套板子买的是一套具备非常大价值的方案，以及学习资料。一个字”值”！测试图像是 1280X720P 的，可以看到最大帧率可以到 37FPS(上位机定时器是 30ms 采集一次)


MSXBO USB3.0 Video(V1.01)
STOP
37fps

CH05 USB3.0 UVC 相机 FT602Q 芯片方案

软件版本：VIVADO2019.2
操作系统：WIN10 64bit
硬件平台：适用XILINX 7系列FPGA(没有USB3.0接口的开发板需要购买FEP USB3.0子卡)
米联客(MSXBO)技术社区： www.uisrc.com 欢迎大家给我提问！！
米联客技术社区新域名www.uisrc.com正在备案中，备案期间请继续使用老域名www.osrc.cn

5.1 概述

本例子中使用的是 FEP 接口外接 USB602Q 子卡。

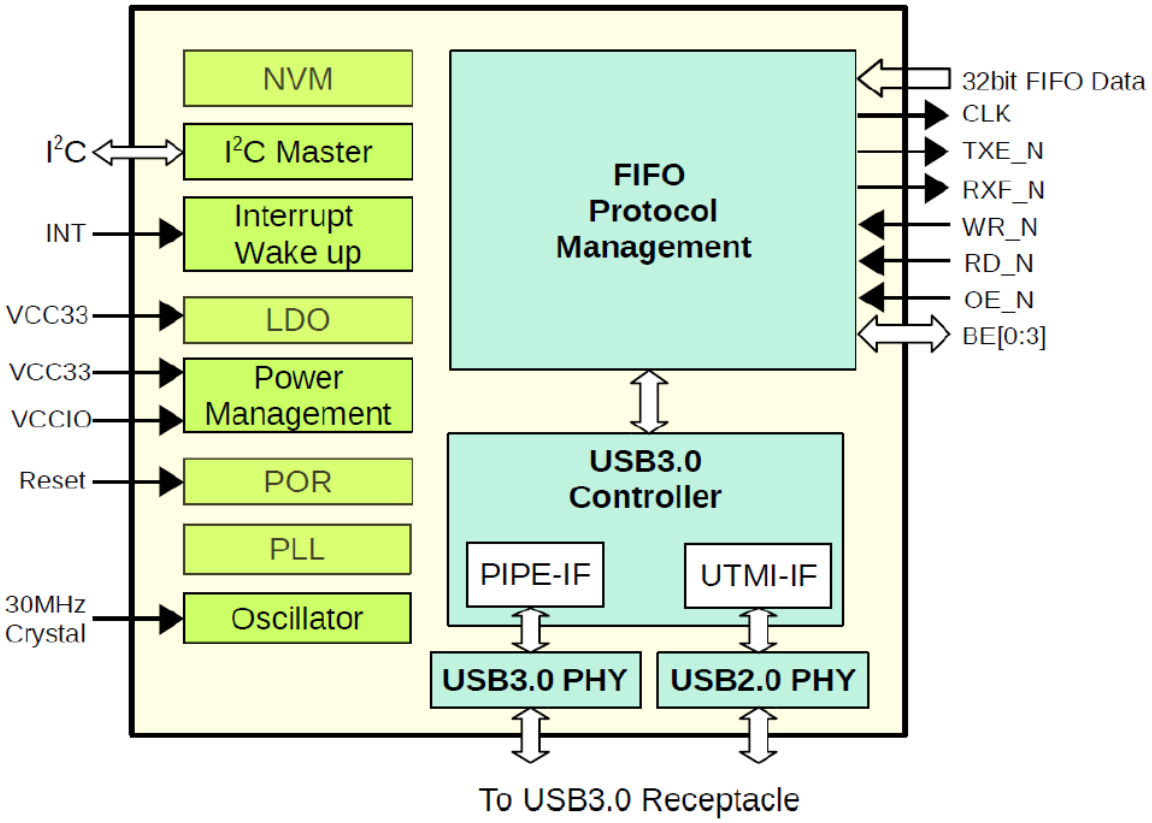
UVC 全称为 USB Video Class，即：USB 视频类，是一种为 USB 视频捕获设备定义的协议标准。是 Microsoft 与另外几家设备厂商联合推出的为 USB 视频捕获设备定义的协议标准，目前已成为 USB org 标准之一。支持 USB Video Class (UVC) standard 1.1 可以让相机在所有的作业系统以及平台中使用(Windows, Linux, Mac etc.)。用户只需连接相机便可进行图像传输，而无需安装任何驱动程序。UVC 相机最适合作为工业网络相机在视频会议、站亭系统、小型设备生产、物流业等应用中使用。

FT602 是 USB-to-FIFO 接口 SuperSpeed USB（USB 3.1 Gen 1）USB 视频类（UVC）桥接芯片具有以下特点

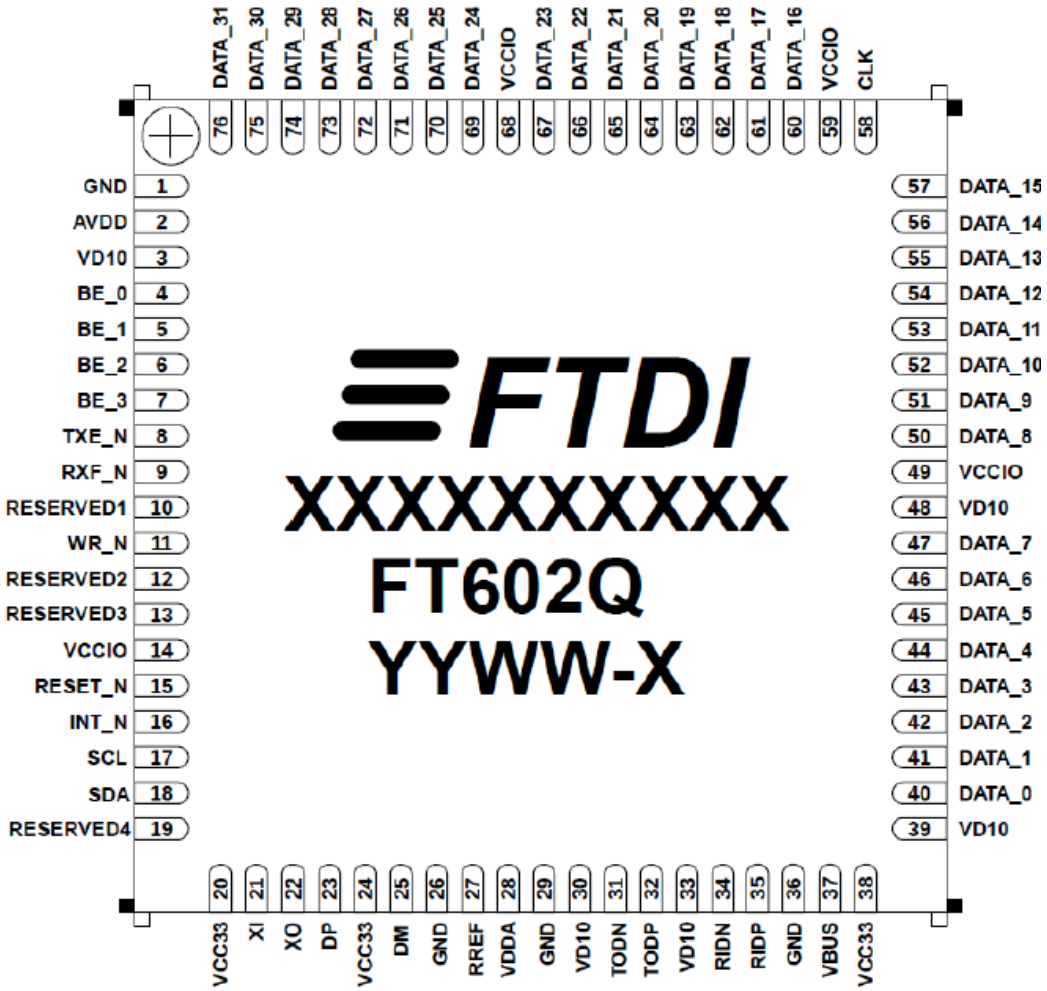
1、支持 USB 3.1 GEN 1 超高速：
（5Gbps）/ USB 2.0 高速（480Mbps）
2、支持 USB 传输类型：
控制/散装/中断
3、支持 UVC 1.1 版：
支持最多 4 个视频输入通道
3、FIFO 总线：
支持 2 个并行从 FIFO 总线协议，245 FIFO 和多通道 FIFO 模式，数据突发速率高达 400MB / s，32 位并行接口
4、内置 16kB FIFO 数据缓冲 RAM。
5、用于视频设备的内置 I2C 主接口
6、组态
支持多电压 I / O：1.8V，2.5V 和 3.3V。
内部 LDO 1.0V 稳压器。
集成的上电复位电路。
用户可编程 USB 和 UVC 描述符。
工业工作温度范围：-40 至 85°C。
符合 RoHS 标准的紧凑型无铅 QFN-76 封装

5.2 硬件方案

5.2.1 芯片框图

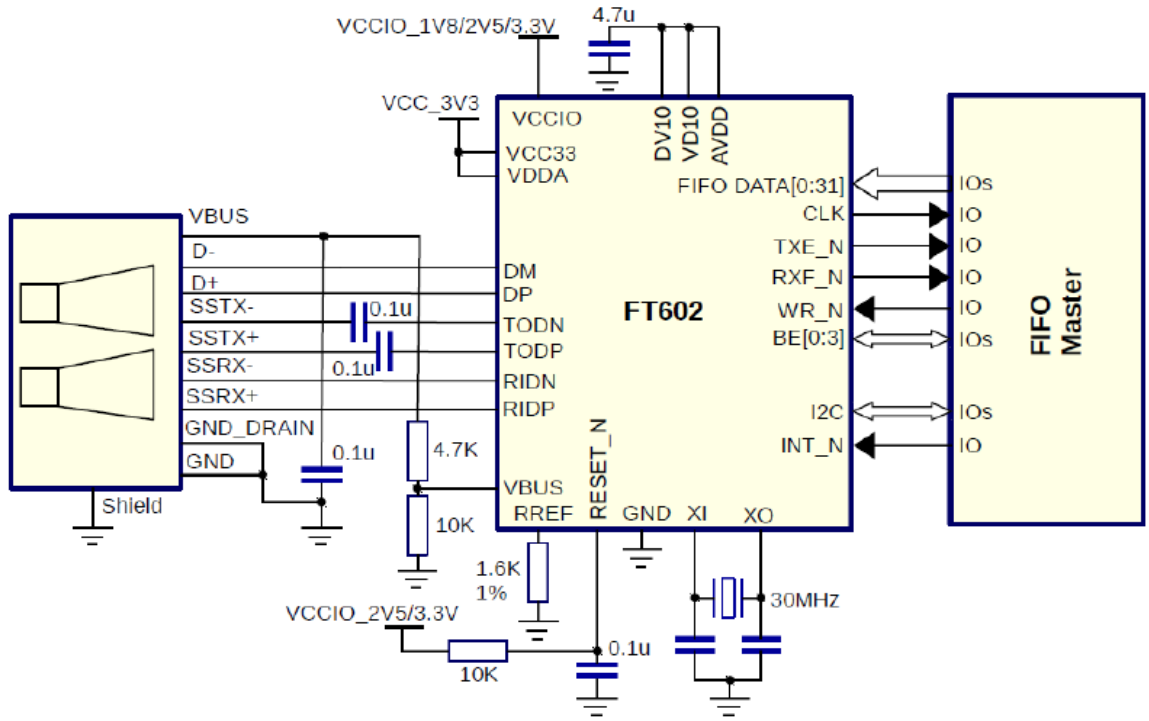


5.2.2 芯片封装



FT602Q 的封装和 FT601Q 的封装是完全兼容的，而且是 QFN 封装，焊接方便

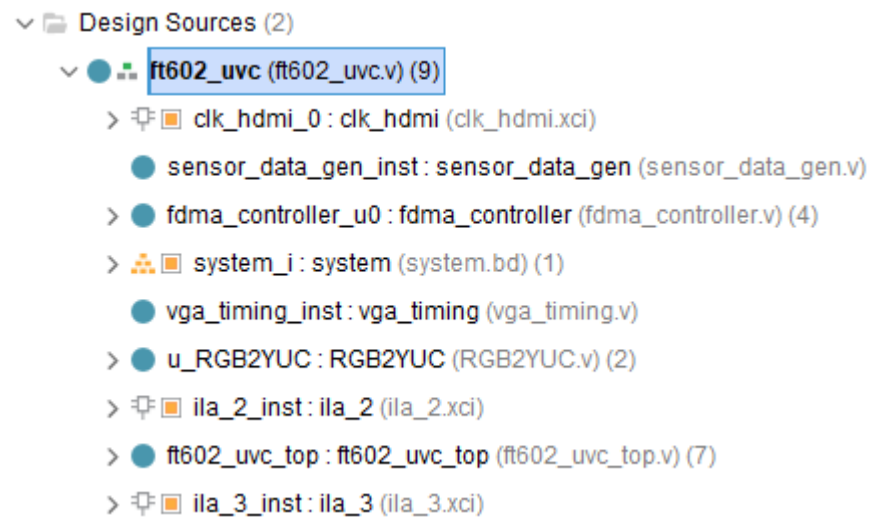
5.2.3 参考电路



可以看到参考电路非常简单，只需要几个电容和几个电阻就可以搞定这方案。

5.3 代码结构

代码结构介绍以 UVC 相机，OV5640 方案介绍：



其中关于 FDMA 的构架部分我们不再介绍，FMDA 的 IP 在我们米联客视频缓存的方案中大量应用，具体的内容可以参考专门讲解 FDMA 的章节，对于 FDMA 使用不熟悉的，可以论坛给我们发帖。

这里重点讲 2 个模块：

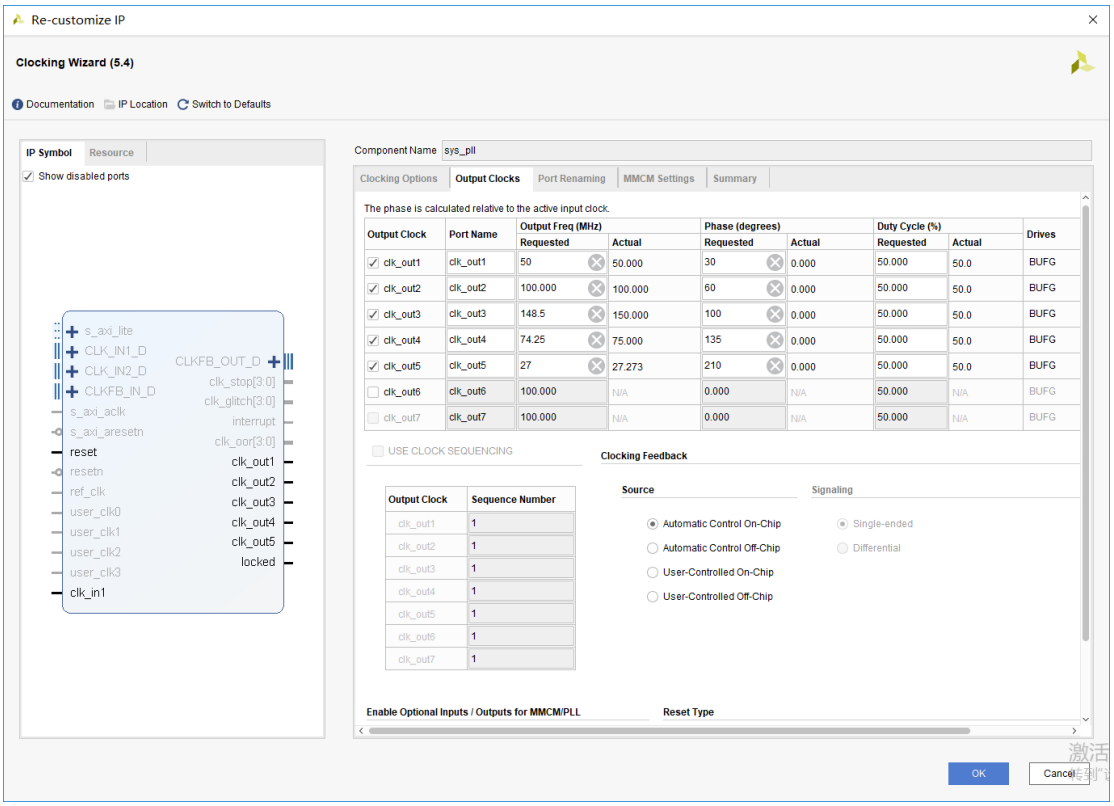
1、RGB2YUV-将 RGB24BITS 数据流转换为 YUV422 的数据流，

2、ft602_uvc_top 配置 ft602 接口芯片，将 YUV422 的视频数据流发送给接口芯片

通过 USB3.0 连接线连接到 PC 机，形成一个照相机设备。

视频制式转换部分不再分析，我们看一下 FT602 的逻辑部分，结构如下：

可以看到顶层模块 ft602_uvc_top.v 包含了 7 个子模块。其中，sys_pll.v 用于生成 ft602 模块所需的时钟，主要是生产视频时许所需的时钟 27MHz (VGA), 74.25MHz (HD) and 148.5MHz (Full HD) 。



模块 ft602_i2c_slv.v 模拟一个 IIC 从机，FT602Q 芯片通过该接口读取 FPGA 中视频信息，同时将上位机的配置信息通过该接口配置到 FPGA。

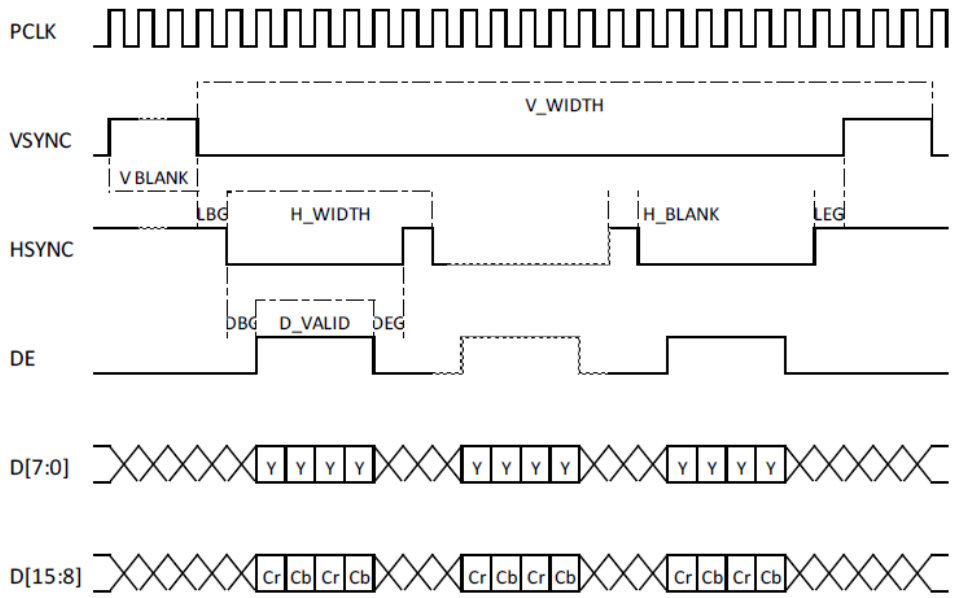
模块 ft602_uvc_reg.v 包含了 UVC 协议对应的一些寄存器，关于视频制式的信息也从这里修改。

模块 ft602_uvc_fsm.v 是 uvc 在总线读写 FIFO 控制的状态机，四个通道，实际工程中，我们只使用一个通道。

模块 Module ft602_pre_fet.v 数据预读取，用于优化时序。类似于 cache 功能。

模块 ft602_pch_inp.v 为视频流处理模块，将 IMG 数据流转换为 FT602Q 对应的接口数据格式，通过 FIFO 缓存。

其中，ft602_patt_gen 模块生成一个测试用的视频源。



模块 ft602_img_rx 处理 IMG 输入视频流数据或者前面生成的测试数据流，将视频格式的数据流转换为 FT602 总线对应的数据格式。

模块 ft602_dclk_fifo 控制双口 ram，接收 ft602_img_rx 模块的视频流数据，同时对外提供 FIFO 读接口。

模块结构如下图：

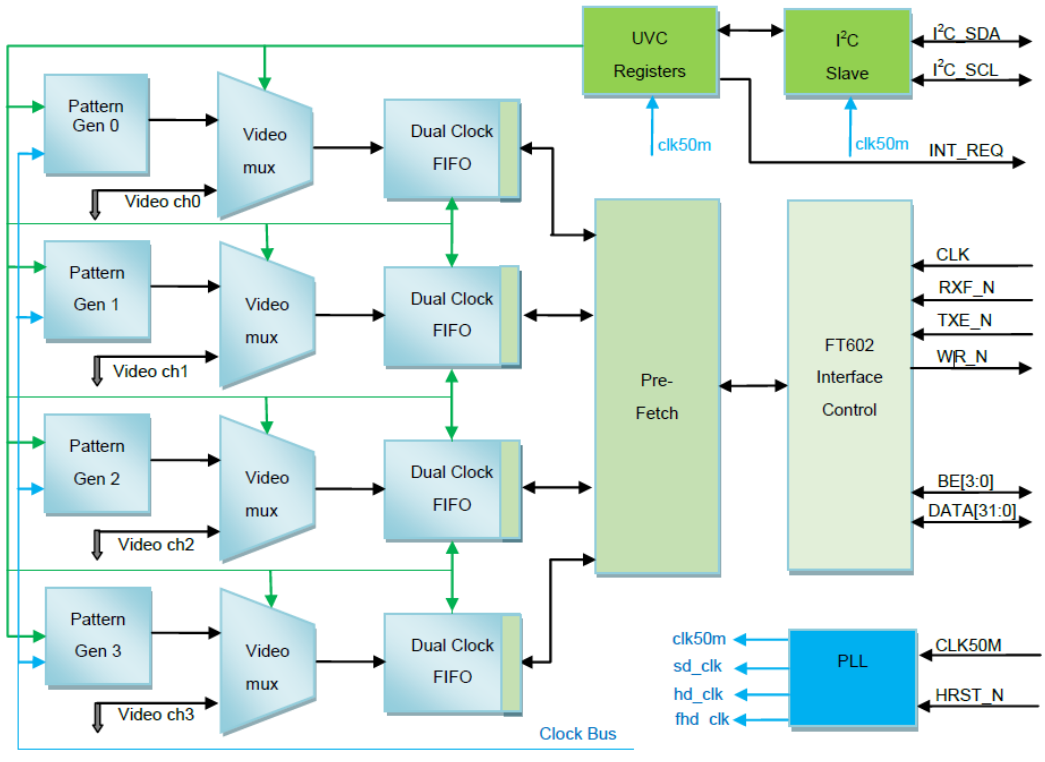


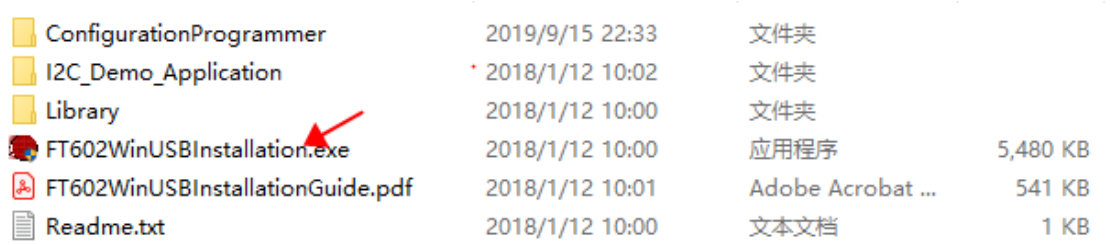
Figure 1 – Architecture

注意，在我们的工程中，只是用了一个通道，也就是 Video ch0 。

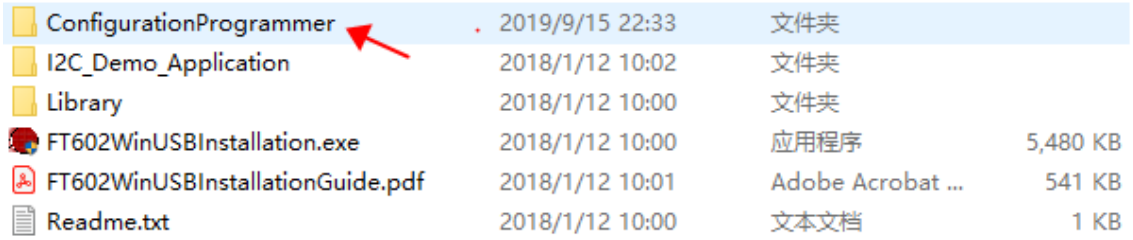
5.4 下载测试

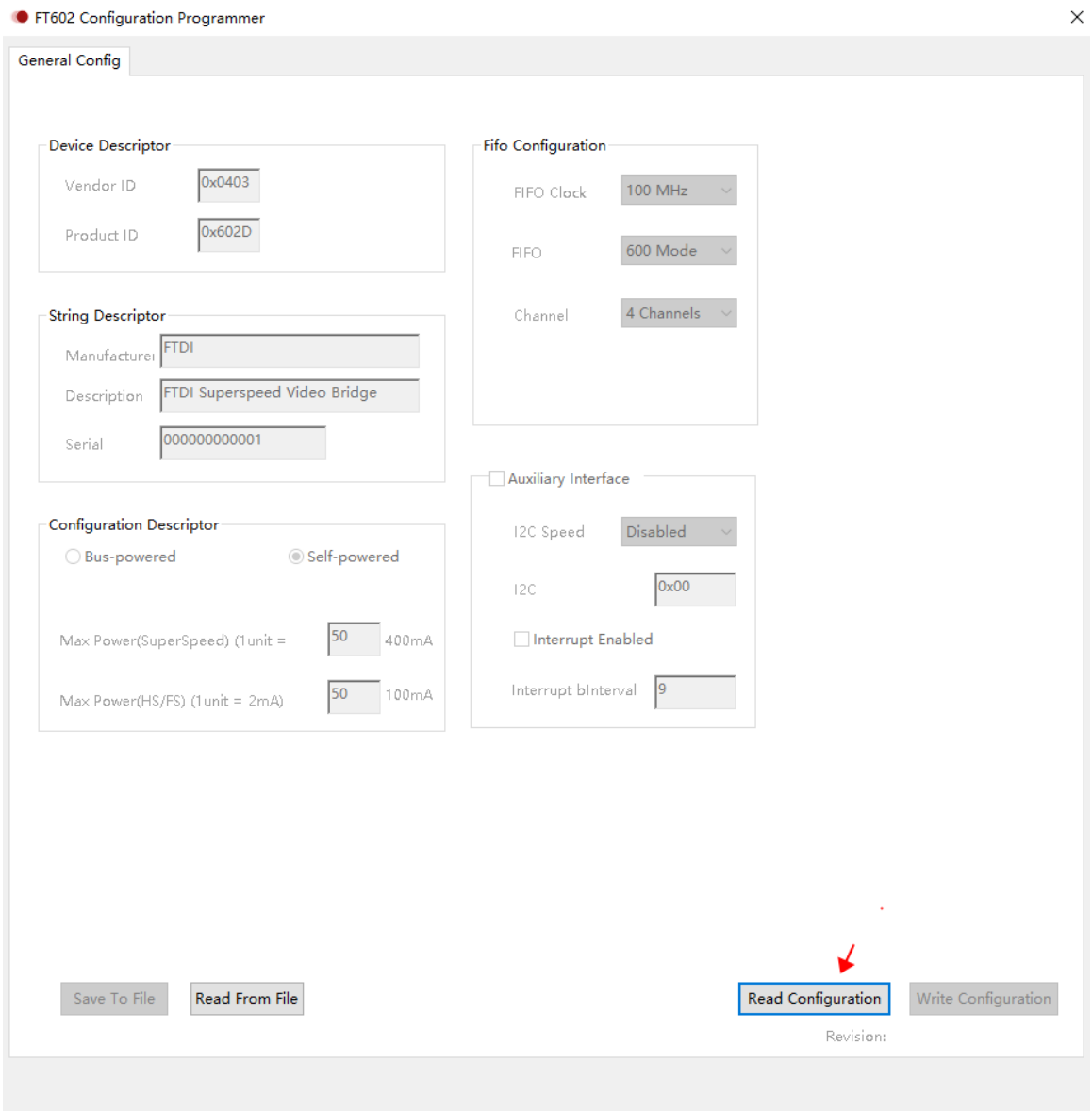
好的，工程结构了解后，编译工程，下载 bit 文件到开发板，连接板上 usb3.0 接口到 PC 机的 USB3.0 接口，连接 IMG 输入视频源。这时，就能在照相机设备中看到 FT Superspeed Video Channel1 的设备。

这证明我们的 usb 设备已经正确识别到了。
这时，打开 win10 系统自带的 camera 软件，就能看到从 IMG 接口输入的视频了。
但是，有时有些设备还是不能成功打开视频。这时，需要我们使用工具设置 FT602Q 的一些参数了。如下图：

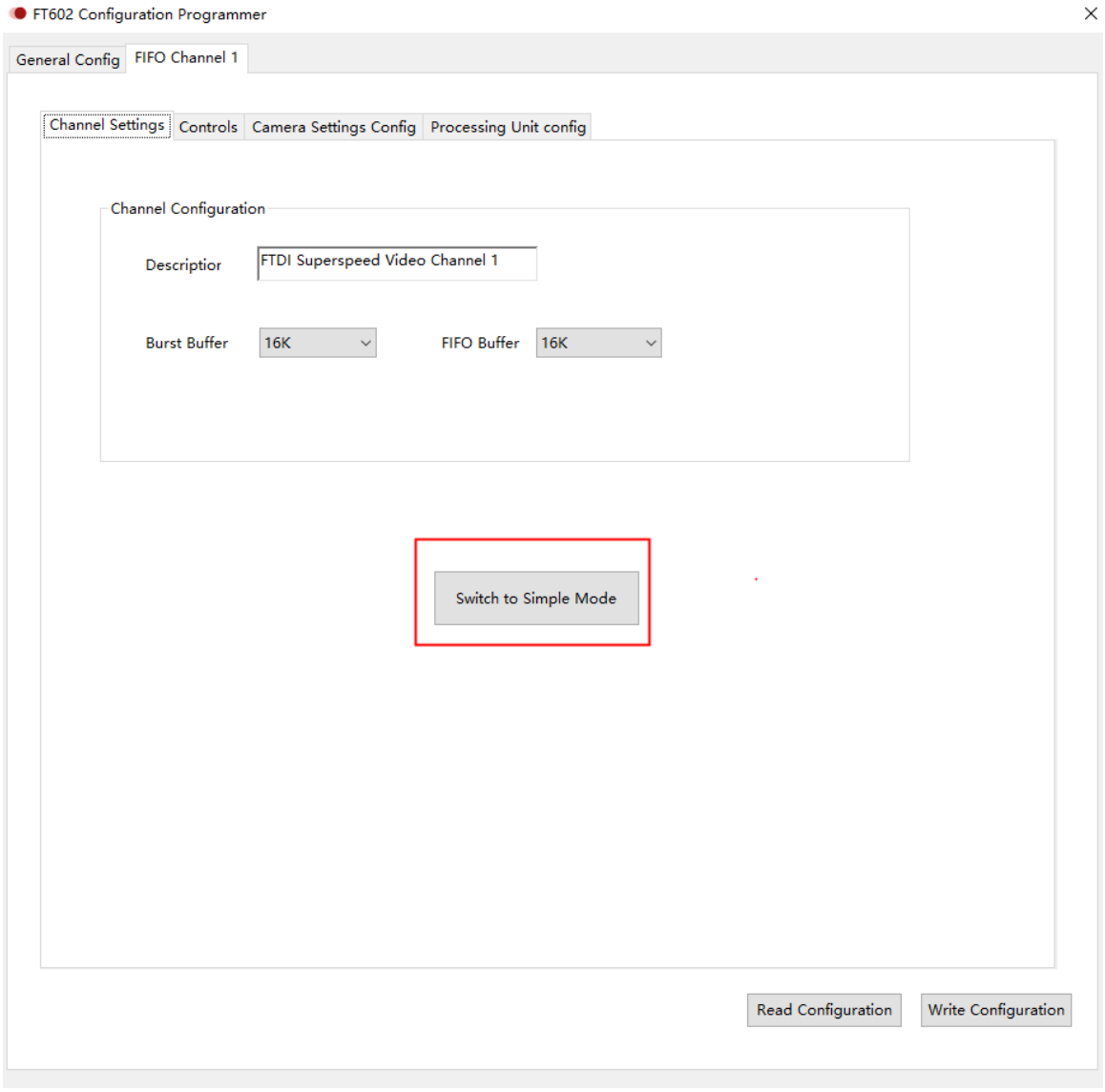


首先，双击 FT602WinUSBInstallation.exe 安装驱动,安装上驱动之后就可以通过软件修改 FT602 的配置了。
在文件夹 ConfigurationProgrammer 中，打开软件 FT602ChipConfigurationProg.exe

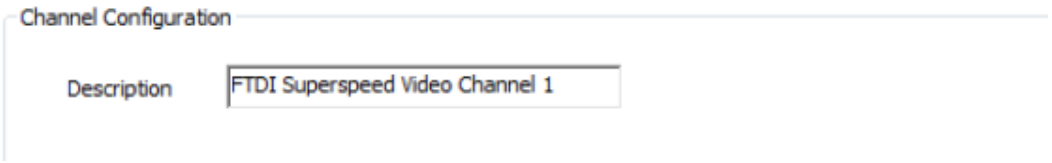




读取 FT602Q 芯片的配置信息，这里可以看到设备的基本信息，同时还包含 FIFO 的配置和 IIC 的配置，这部分基本采用默认方式就可以。点击 FIFO Channel1 选项卡，转到高级设置中：



注意，这里的 BurstBuffer 和 FIFO Buffer 都设置为最大的 16K。同时我们也可以修改设备描述 FT Superspeed Video Channel1，换成一个自定义的设备名。

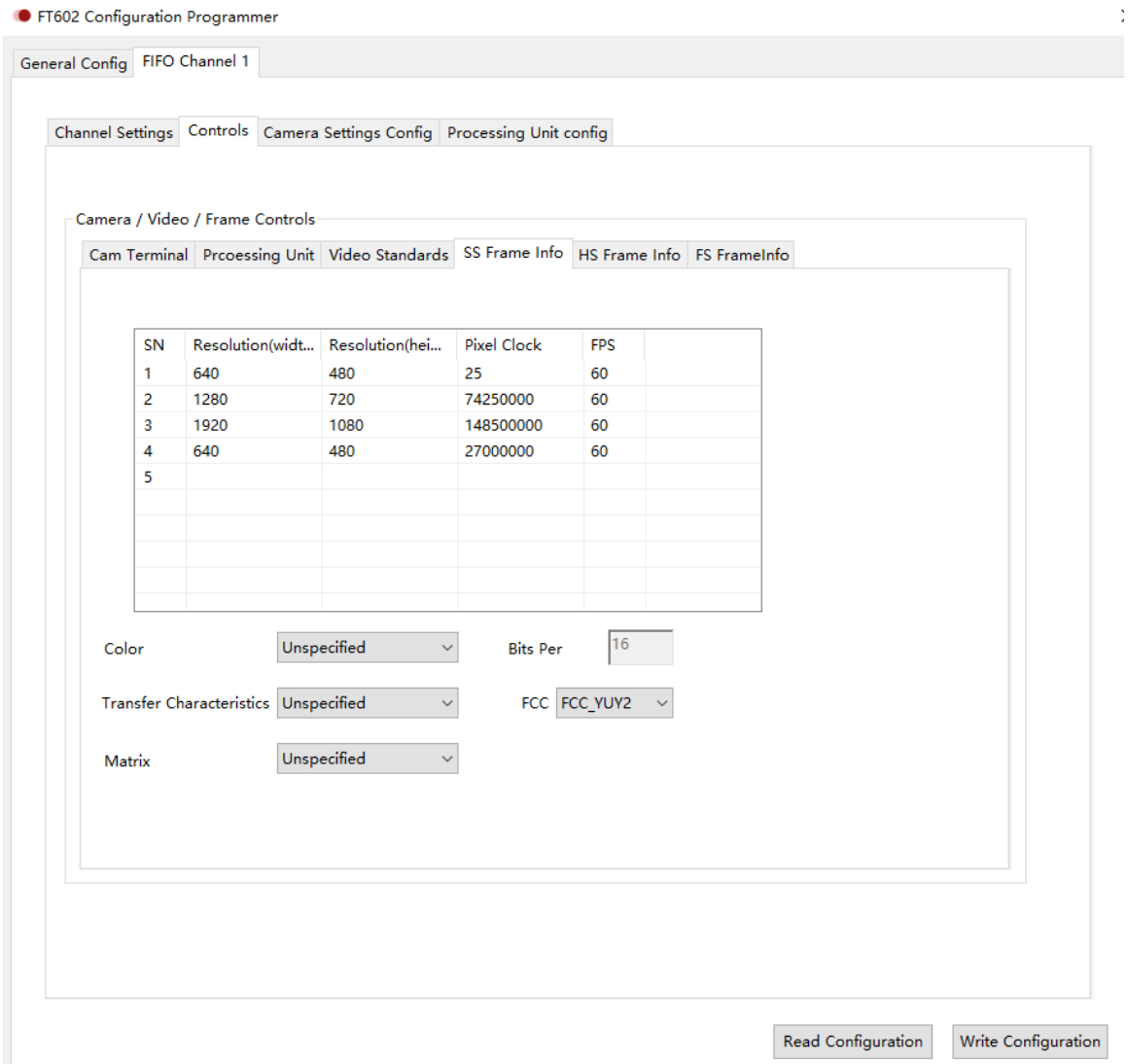


高级选项中主要涉及到视频制式和处理的信息，这里我们只调整 Controls 选项卡下的 SS Frame Info，这是因为我们做 IMG 传输需要使用的 USB 的超速传输模式。

Index	USB MODE	Resolution	Pixel Clock
1	High Speed / Full Speed	QVGA	27 MHz
2	Super Speed	VGA	27 MHz
3	Super Speed	HD	74.25 MHz
4	Super Speed	Full HD	148.5 MHz

Table 3 – Pixel Clock

来看一下 SS Frame Info：



我们看到在视频制式信息里，已经包含了 1080P 720P 640P 的信息，但是，有的设备并不能直接识别哪一种分辨率，比如我们能这里使用：

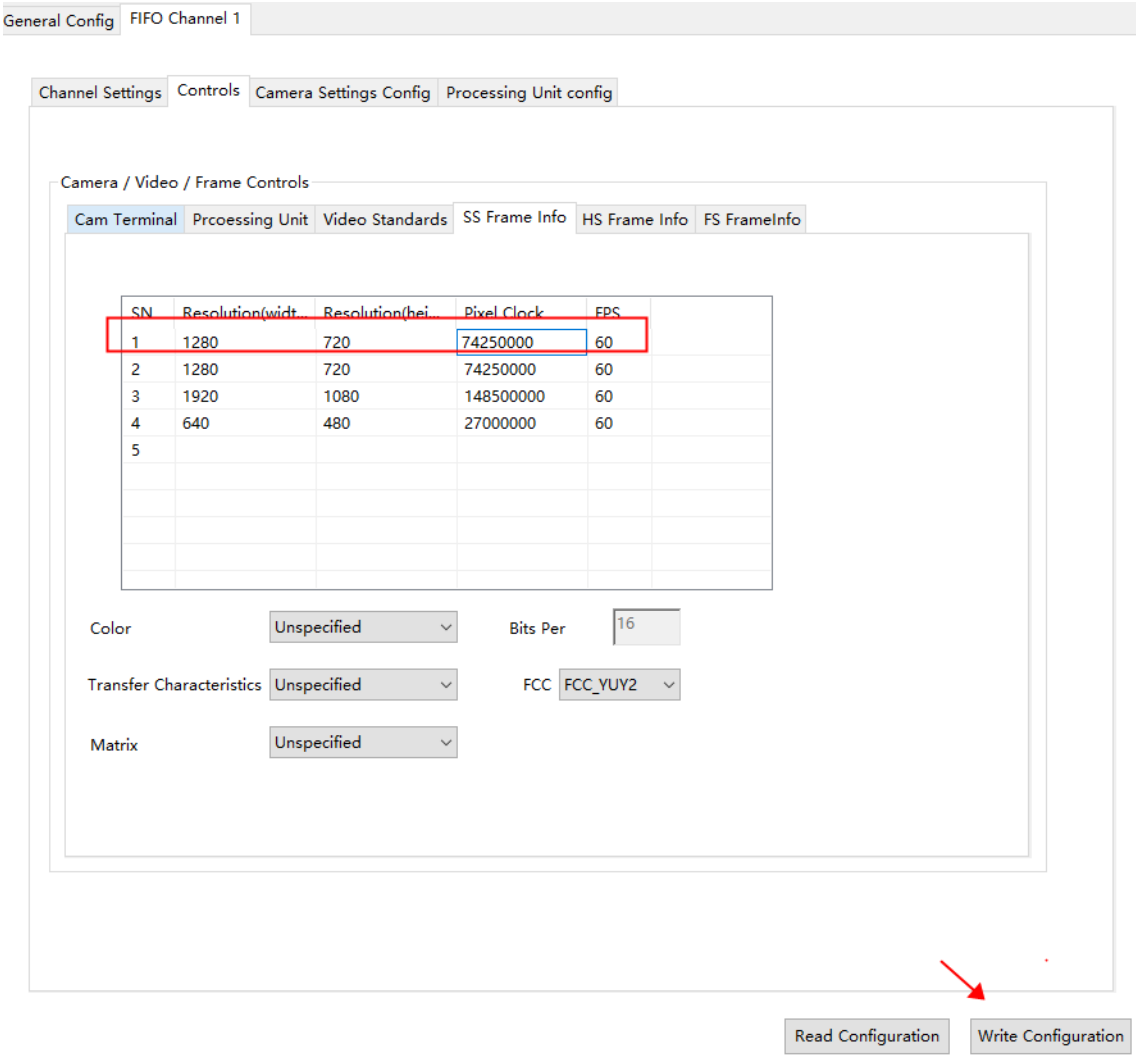
OV5640 设置 1280X720 分辨率 60 帧 时钟 74.25M

9V034 设置 640X480 分辨率 60 帧 时钟 25M

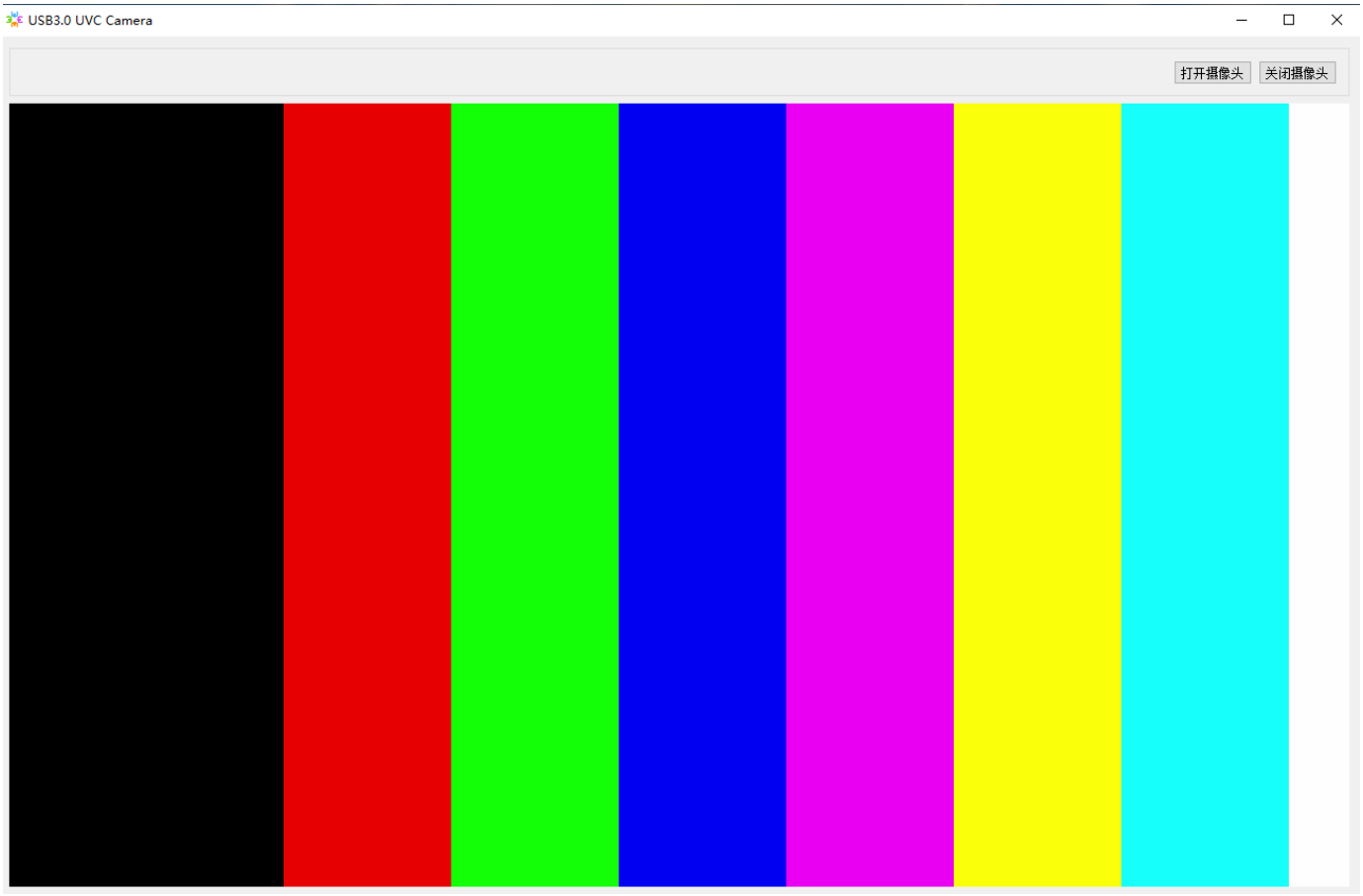
HDMI 设置 1920X1080 分辨率 30 帧 时钟 74.25M

5.5 UVC 彩条方格测试

A703_35T 资源和硬件限制，不能做摄像头采集。本例子中使用的是彩条方格测试设置如下，然后将配置信息写入器件：



这样，重启此设备(重新插拔 USB)，就可以在视频软件中打开设备了。


USB3.0 UVC Camera
打开摄像头
关闭摄像头