视频图像光纤传输例程

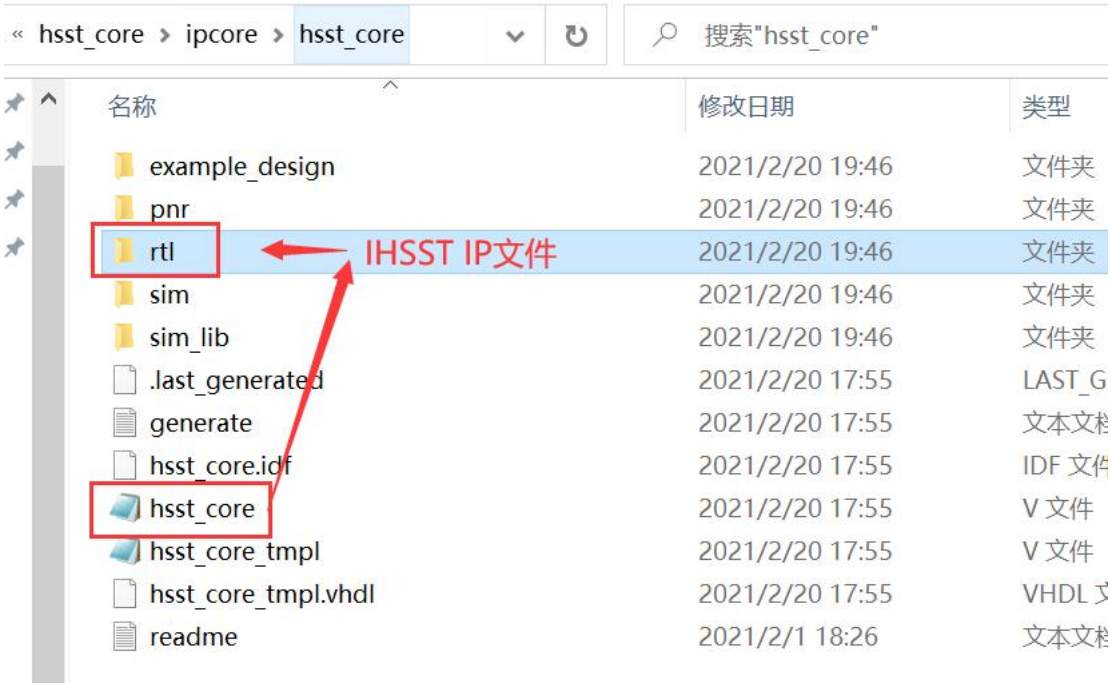
1 实验简介

本实验将介绍通过光纤实现视频图像的传输，视频图像由黑金双目摄像头模块 AN5642 采集，再通过开发板上的其中两路光模块进行视频信号的光纤发送和接收，然后在 HDMI 显示器上显示出来。

2 实验原理

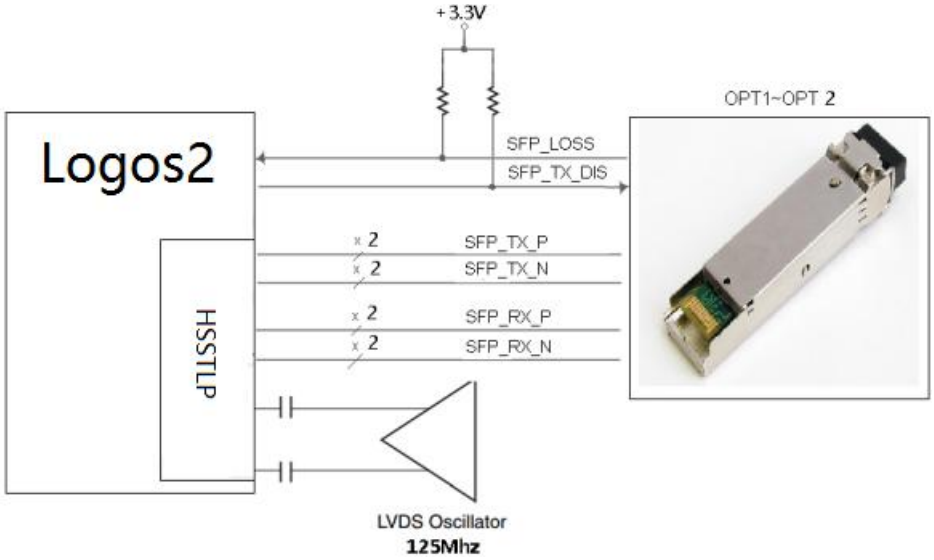
2.1 HSSTLP IP 设计

HSSTLP IP 产生在《光纤通信测试例程》教程中已讲述，这里不再做介绍。由于我们通信数据是以 5G 速率进行收发，在这里我们可以直接在《光纤通信测试例程》例程中进行修改。注意当 IP 设置不变时如下与 IP 相关的文件我们可不用修改，可以直接引用。



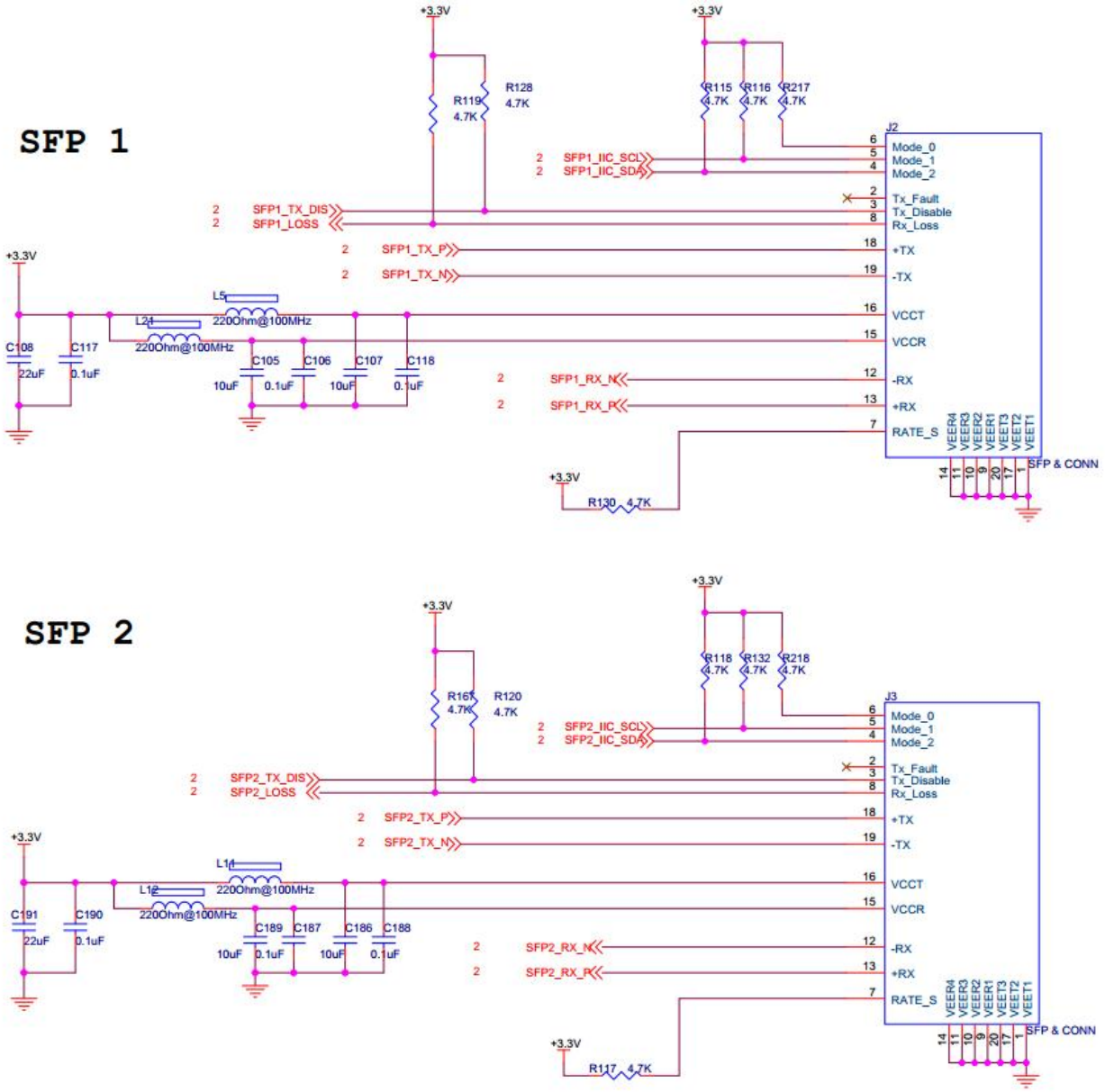
2.2 硬件介绍

在开发板上，有 2 路光纤接口 SFP1~SFP2, 分别连接到 FPGA 芯片的 HSSTLP 的通道上。FPGA 和光纤连接的设计示意图如下图所示:



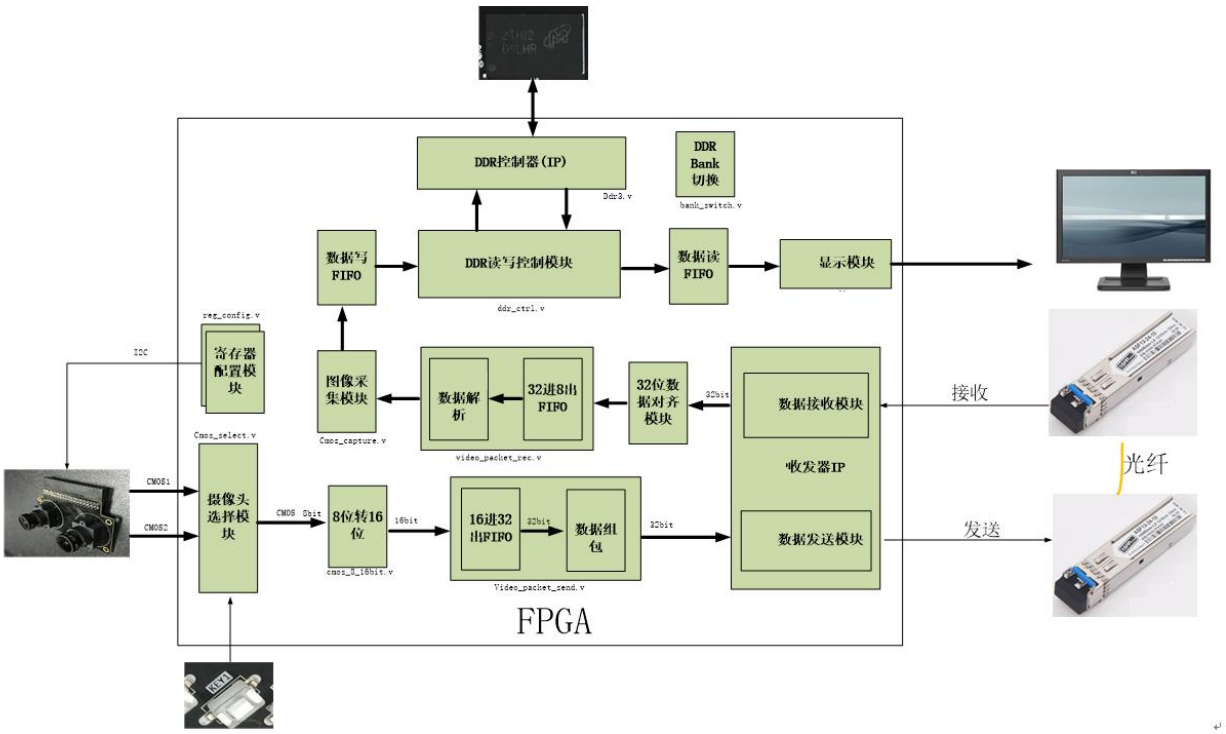
其中 SFP1 光模块接口连接到 Q3 HSSTLP 的 Channel3 上，SFP2 跟 Q3 HSSTLP 的的 Channel2 相连。光模块和 FPGA 之间用 0.1uf 的电容隔开，使用 AC Couple 的模式。

光模块的 LOSS 信号和 TX_Disable 信号连接到 FPGA 的普通 IO 上。LOSS 信号用来检测光模块的光接收是否丢失，如果没有插入光纤或者 Link 上，LOSS 信号为高，否则为低。TX_Disable 信号用来使能或者不使能光模块的光发射，如果 TX_Disable 信号为高，光发射关闭，否则光发送使能，正常使用的时候需要拉低此信号。硬件原理图如下：



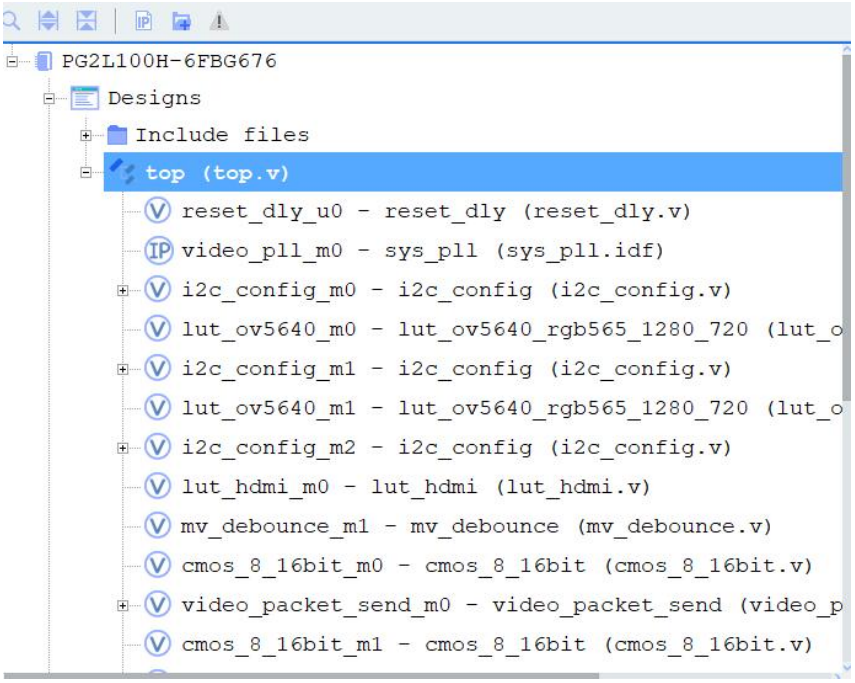
3 程序设计

视频图像光纤传输的 FPGA 程序设计的逻辑框图如下图所示：



视频图像从双目摄像头 AN5642 传入 FPGA 后，先通过摄像头切换模块来选择其中的一路，切换使用开发板上的按键来实现，视频图像先转换成 16 位的数据宽度，再存入到一个 16 位进，32 位出的 FIFO 中。当 FIFO 中的数据达到一定量的时候（1 行视频数据），从 FIFO 中取出数据使用 HSSTLP IP 发送给外部的光模块 1，光模块 1 把电信号转换成光信号通过光纤传输到光模块 2，光模块 2 又把光信号转换成电信号输入到 FPGA 的 HSSTLP 接收，HSSTLP 接收到的数据需要做一个 32 位数据对齐之后，并解析出视频图像部分的数据，再把数据存入到 FIFO 中。后面就是把图像传到 DDR 的缓存中，在 HDMI 显示器上显示。

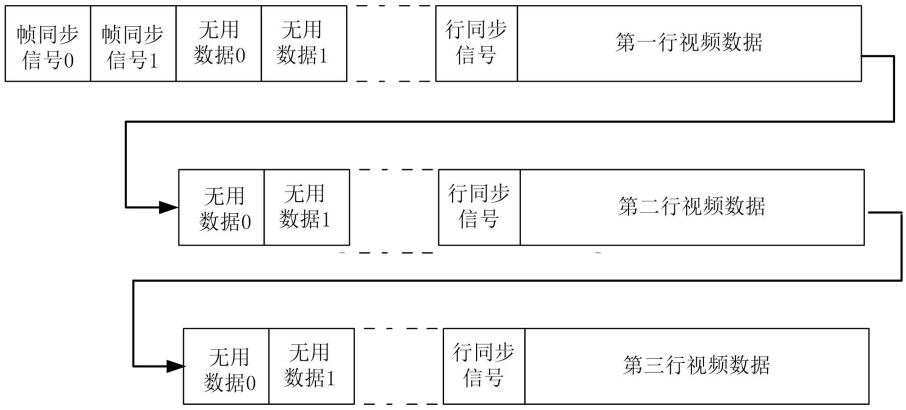
视频图像光纤传输设计好的工程如下图所示：



因为大部分程序跟前面的双目 HDMI 切换显示实验一样，我们这里只对增加的部分模块做一下功能说明：

1） 视频数据包准备模块 video_packet_send.v

接收到的视频图像数据存放在 16 位进，32 位出的 FIFO 中，在 video_packet_send.v 中会由一个状态机来发送视频图像的数据，首先一帧图像开始传输前，HSSTLP 会发送帧同步信号。再判断这个 FIFO 中的数据量，如果 FIFO 内的数据还没有一行的视频数据，HSSTLP 则发送无用的数据，当 FIFO 内已经有一行视频的数据时，HSSTLP 会先发送行同步信号，然后再把这一行视频的数据通过 HSSTLP 发送出去。一行数据发送完成，再重新判断 FIFO 内的数据量，FIFO 数据量达到一行视频数据时，接着发送第二行视频图像。HSSTLP 数据发送一帧图像的流程如下图：



所有的 HSSTLP 发送的数据位数为 32 位，帧同步信号、行同步信号、无用数据定义如下：

图像帧同步信号 0：定义为 32 位的“ff_00_00_bc”

图像帧同步信号 1：定义为 32 位的“ff_00_01_bc”

插入的无用数据 0：定义为 32 位的“ff_55_55_bc”

插入的无用数据 1：定义为 32 位的“ff_aa_aa_bc”

图像行同步信号：定义为 32 位的“ff_00_02_bc”

这些同步信号和无用数据的高 24 位数据是用户自己定义的，低 8 位"bc"是 K28.5 码控制字符。

向 HSSTLP 发送 K28.5 码控制字符时，需要拉高 gt_tx_ctrl 信号的对应位，标示发送数据里的某个字节位为 K 码控制字。所以这里在向 HSSTLP 发送同步信号和无用数据的时候，gt_tx_ctrl 信号设置为 0001，发送视频数据的时候则置为 0000。

```
105         case(state)
106             SEND_FRAME_SYNC0:                   //发送视频的帧同步信号0
107             begin
108                 state <= SEND_FRAME_SYNC1;
109                 gt_tx_data <= 32'hff_00_00_bc;
110                 gt_tx_ctrl <= 4'b0001;
111             end
112             SEND_FRAME_SYNC1:                   //发送视频的帧同步信号1
113             begin
114                 state <= SEND_OTHER0;
115                 gt_tx_data <= 32'hff_00_01_bc;
116                 gt_tx_ctrl <= 4'b0001;
117             end
118             SEND_OTHER0:                        //发送视频的无用的信号ff_55_55_bc
119             begin
120                 state <= SEND_OTHER1;
121                 gt_tx_data <= 32'hff_55_55_bc;
122                 gt_tx_ctrl <= 4'b0001;
123             end
124             SEND_OTHER1:                        //发送视频的无用的信号ff_aa_aa_bc
125             begin
```

2）位数据对齐模块 word_align.v

收发器外部用户数据接口的宽度为 32 位，内部数据宽度为 20 位(8b/10b 转换)。在实际测试过程中发现，发送的 32 位数据会有可能出现 16 位的数据的移位，就是说发送的数据和接收到的数据会有 16 位的错位，下表演示发送数据和接收数据移位的情况：

HSSTLP 发送的数据		HSSTLP 接收的数据	
数据 1	11111111	数据 1	11112222
数据 2	22222222	数据 2	22223333
数据 3	33333333	数据 3	33334444
数据 4	44444444	数据 4	44445555
数据 5	55555555	数据 5	5555.....
.....	.....	.....	.....

因为我们在发送同步信号和无用数据的时候加入了 K 码控制字，并且设置 gt_tx_ctrl 信号为 0001, 如果出现 16 位数据移位的情况，接收到的同步信号和无用数据时，K 码控制字也会跟着移位，gt_tx_ctrl 的信号就会变成 0100。所以我们在程序可以通过判断 gt_tx_ctrl 信号的值来判断接收到的数据是否移位，如果接收到的 gt_tx_ctrl 为 0001，跟我们发送的时候一样，说明数据没有移位；如果接收到的 gt_tx_ctrl 为 0100，接收到的数据移位，需要重新组合，在 word_align.v 模块里完成。

3）视频数据解析模块 video_packet_rec.v

因为接收到的 32 位数据中只有一部分是视频图像的数据，其它的是帧同步，行同步和无用的数据，在 video_packet_rec.v 模块里需要把视频图像的数据解析出来存入到一个 32 位进，8 位出的 FIFO 中。

程序的一个功能是检测 HSSTLP 数据中的行同步信号（数据为 ff_00_02_bc），如果接收到行同步信号，就把后面接收的一行视频数据存放到 FIFO 中。

```
//解析行同步信号
always@(posedge rx_clk or posedge rst)
begin
    if(rst)
        wr_en <= 1'b0;
    else if(gt_rx_ctrl == 4'b0001 && gt_rx_data == 32'hff_00_02_bc)
        wr_en <= 1'b1;
    else if(wr_cnt == ({1'b0,vout_width[15:1]} - 16'd1))
        wr_en <= 1'b0;

end
```

程序的另一个功能是恢复视频图像的帧同步信号（数据为 ff_00_00_bc），如果接收到帧同步信号，则置位视频图像的帧信号 vs。

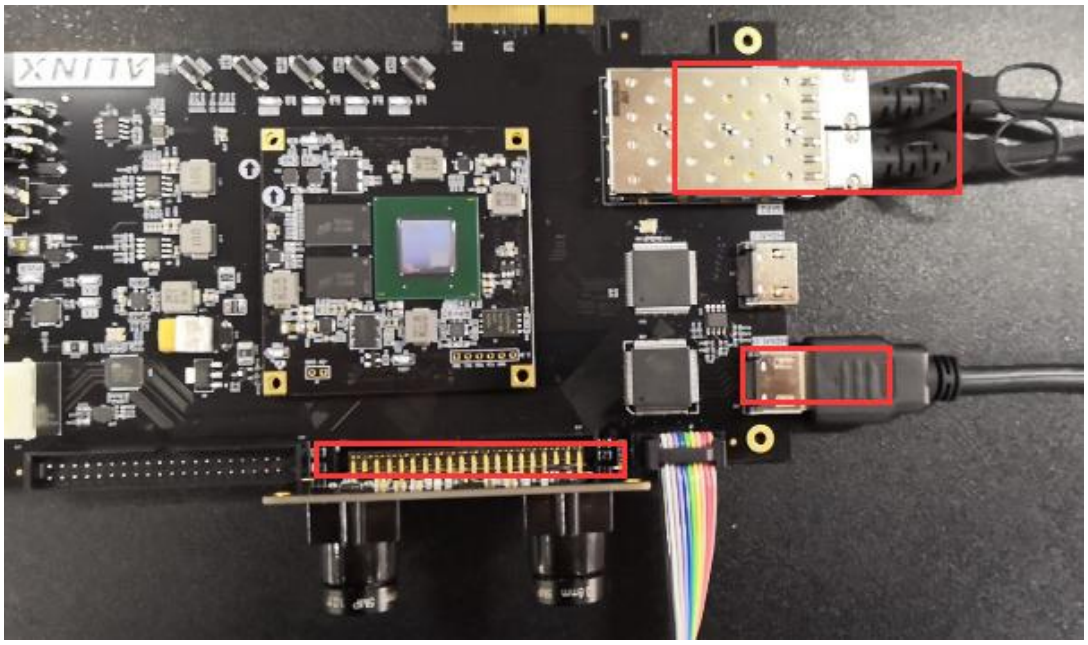
```
24  //恢复帧同步信号
25  always@(posedge rx_clk or posedge rst)
26  begin
27      if(rst)
28          vs_r <= 1'b0;
29      else if(gt_rx_ctrl == 4'b0001 && gt_rx_data == 32'hff_00_00_bc)
30          vs_r <= 1'b1;
31      else if(vs_cnt > 16'd100)
32          vs_r <= 1'b0;
33  end
```

另外模块中会判断 FIFO 中存入的视频数据，如果 FIFO 内的数据量大于一行视频的数据，则产生 FIFO 的读使能信号，把 FIFO 的一行视频数据输出给外部接口模块。

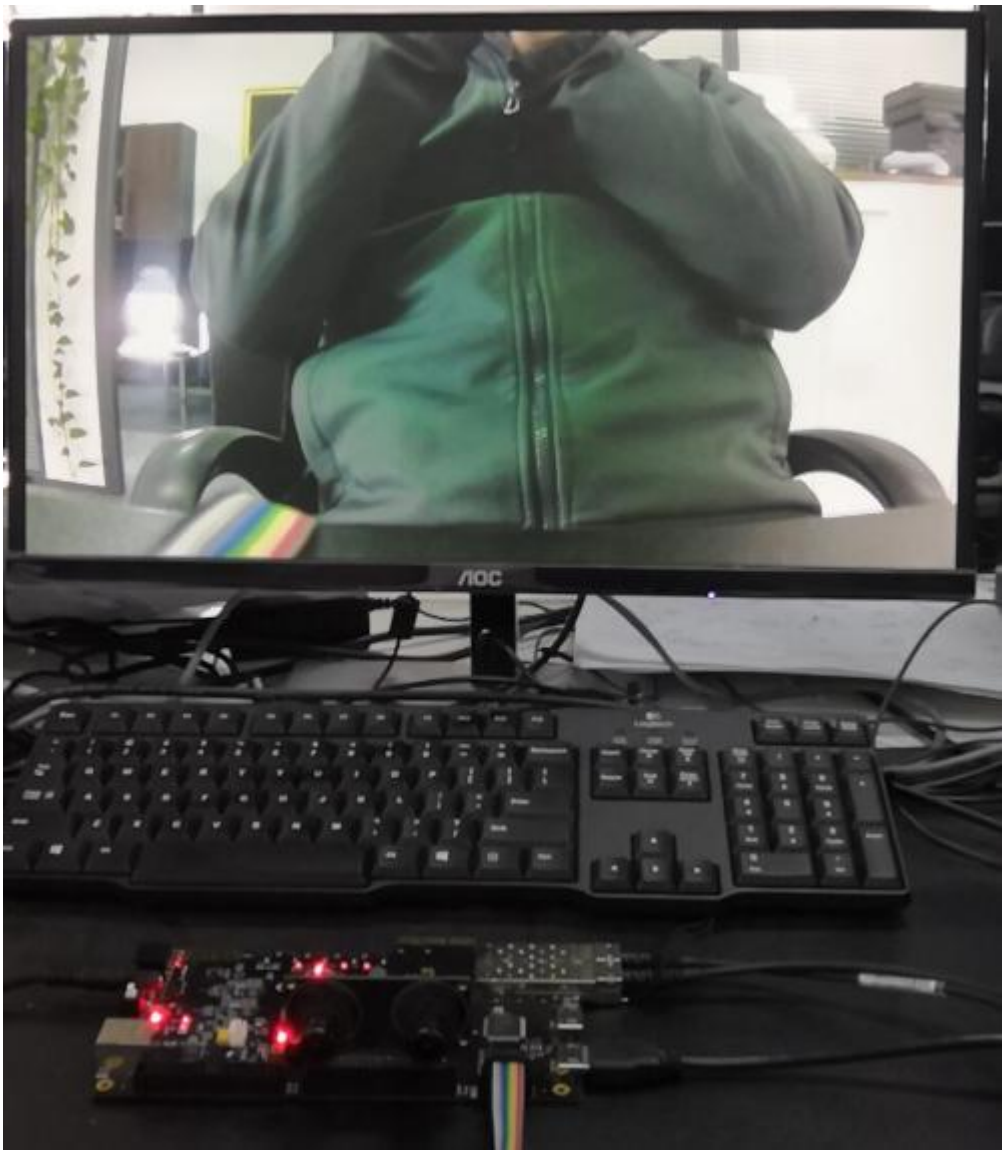
其它模块在这里我们不做介绍了。

4 光纤视频传输现象

编译项目通过后我们就可以开始 AN5642 双目光纤视频传输的实验了。开发板的扩展口(J13)上插上摄像头 AN5642 模块，开发板的 HDMI 口连接 HDMI 显示器，光模块插入到 SFP1~SFP2 的接口上，再用光纤互连。操作如下图所示，由于 AXP100 与图处中开发板的区别只是 FPGA 芯片，其它没有变化，在这里不再贴图。



再下载位流文件到 FPGA, 我们就可以在 HDMI 显示器上看到视频图像从 AN5642 采集后 FPGA 先通过光纤传输，再环路回 FPGA，然后在 HDMI 显示器上显示的视频图像了。



按一下开发板上的 KEY2 按钮，视频图像会切换到另外一路的摄像头。