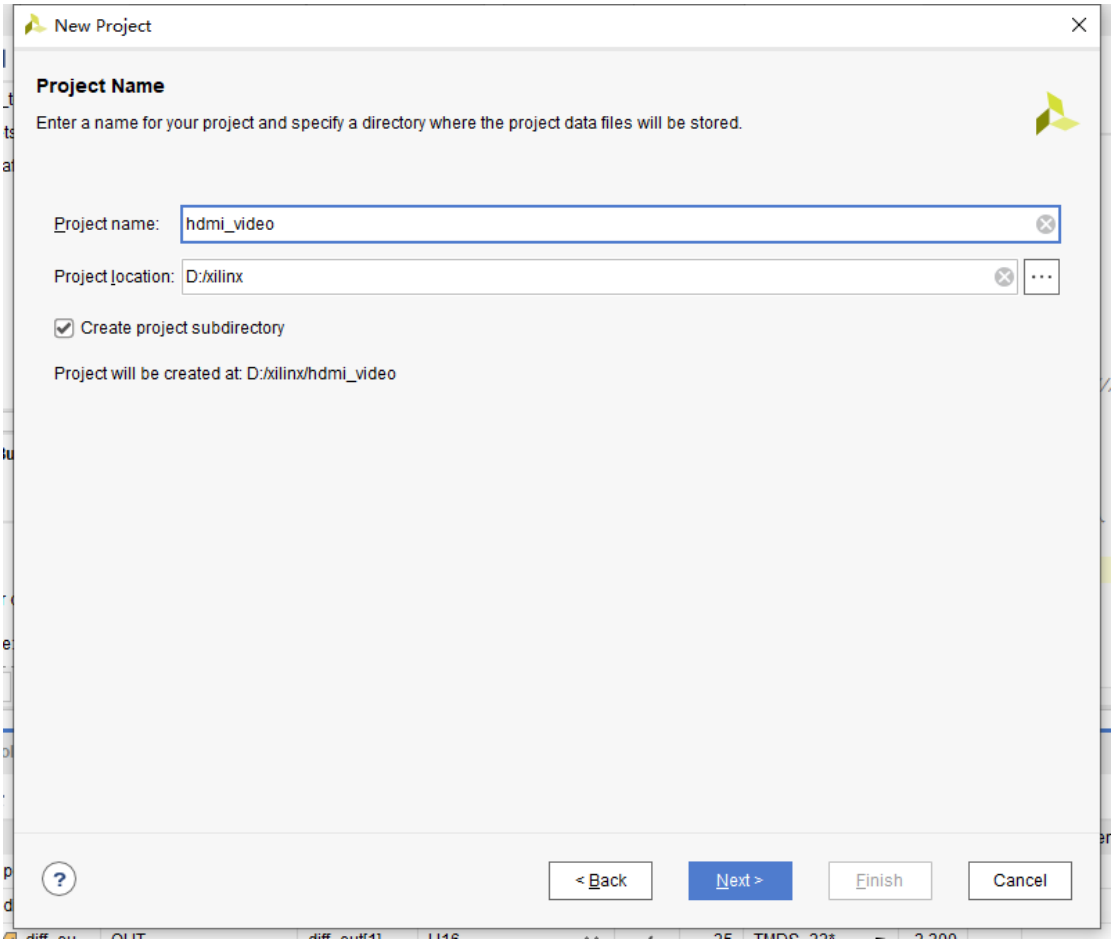
[L06]HDMI显示 - PL

FPGA的特点大概都知道,这里不啰嗦,除了赛灵思自己做的IP,还有很多优秀的IP可以用,比如Digilent做的RGB2DVI模块,就可以刷一个RGB信号然后显示HDMI,当然模块源码未必会提供.

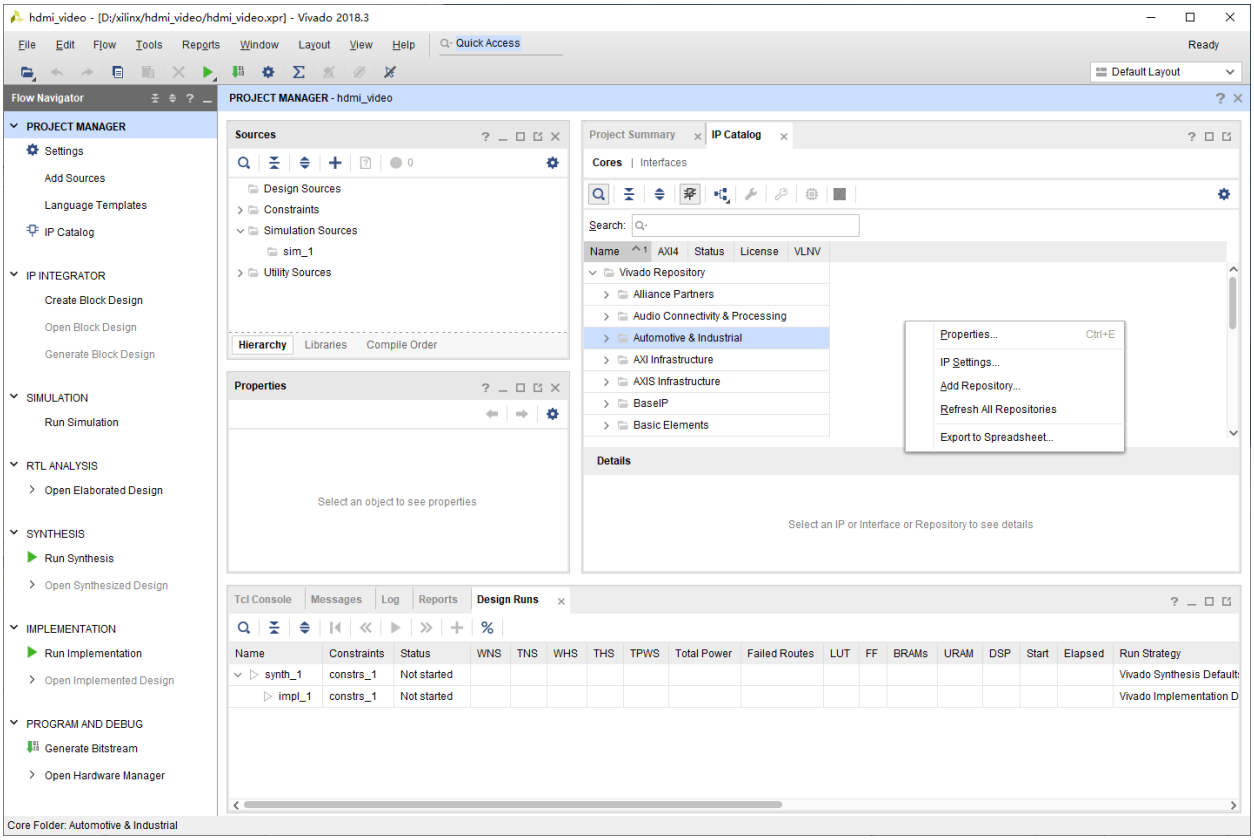
链接:https://digilent.com/reference/vivado/library

下载地址:https://github.com/Digilent/vivado-library

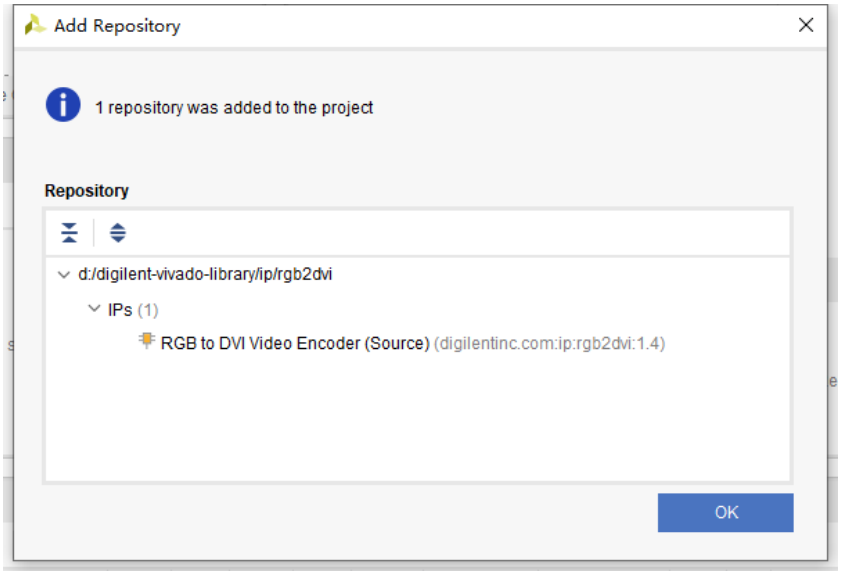
新建一个工程,名为hdmi_video,具体操作就不截图了.



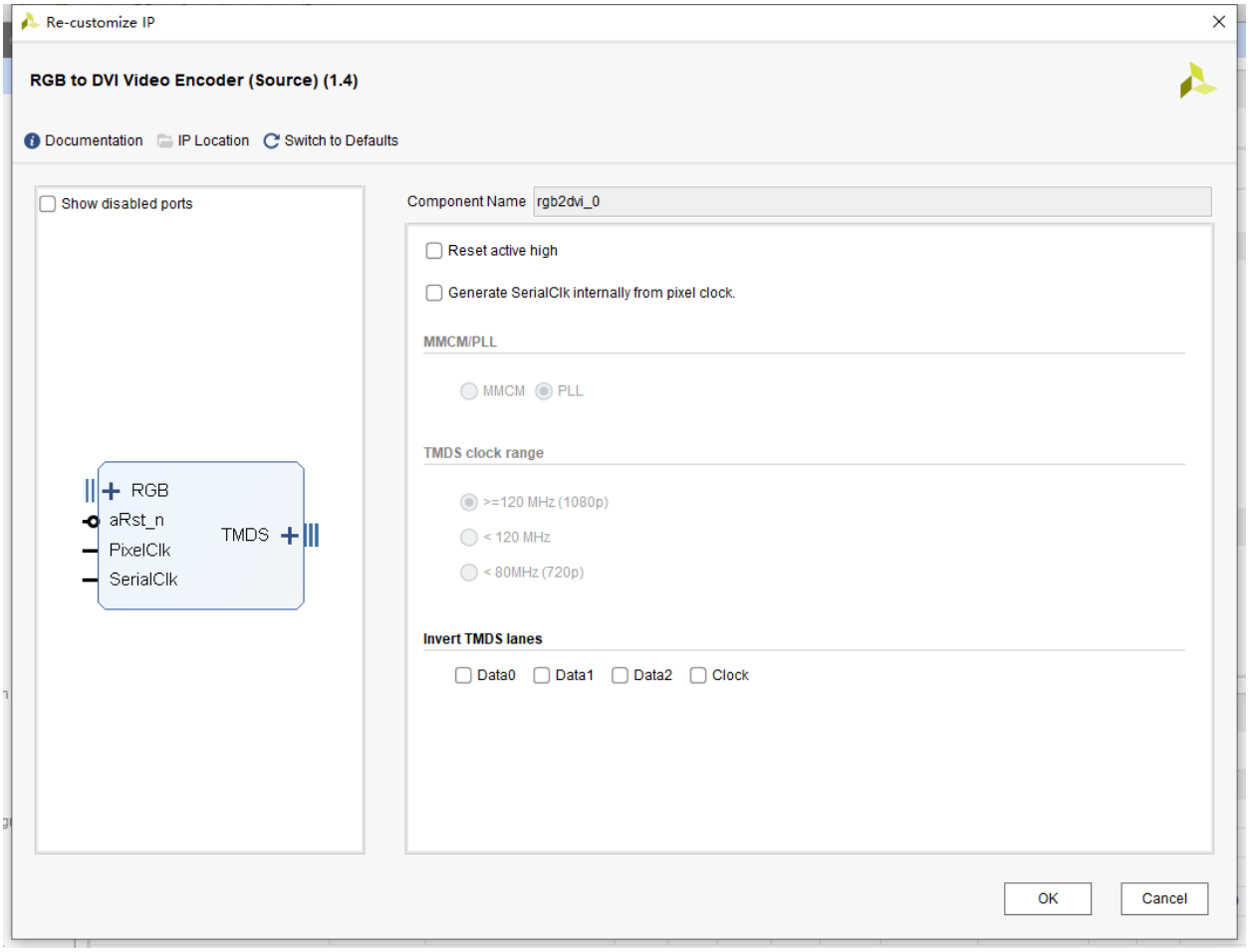
在IP Catalog右键Add Repository...



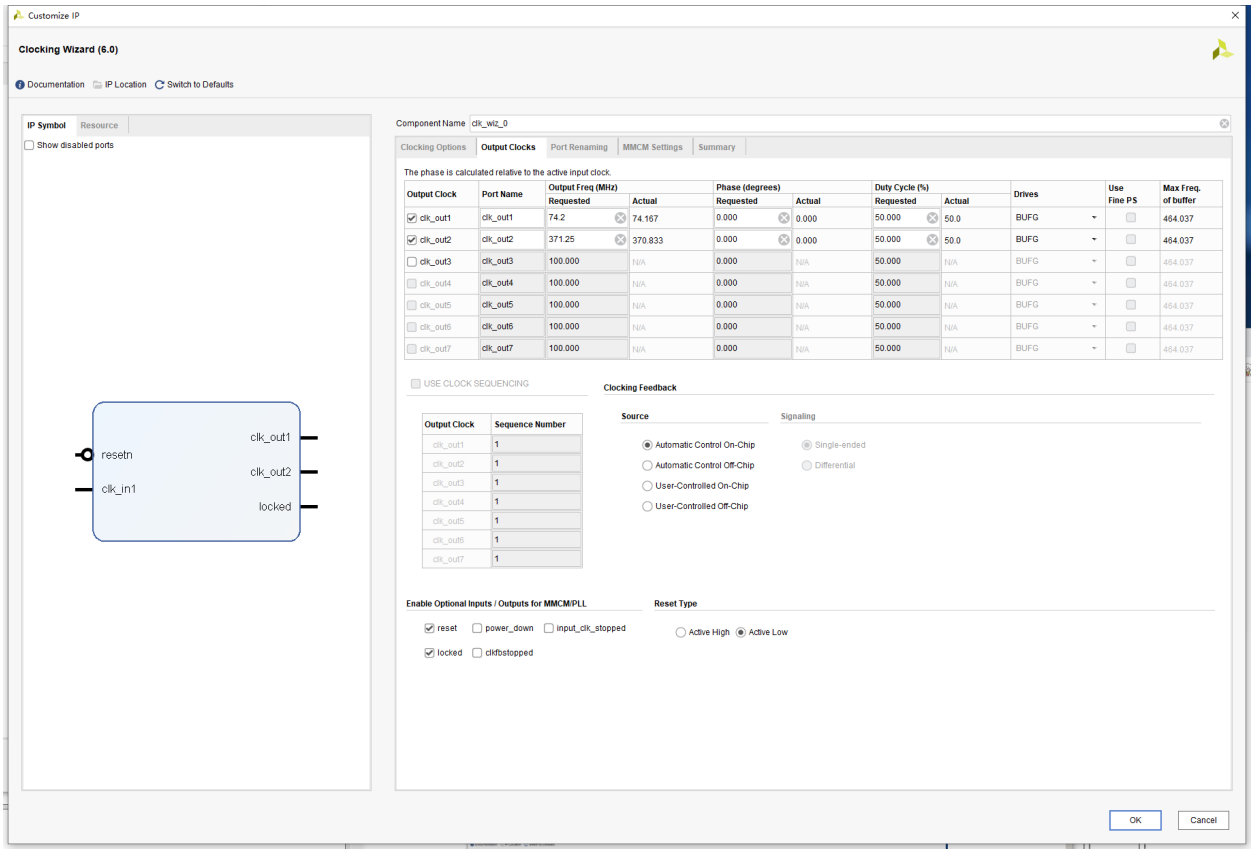
添加RGB2DVI IP.



配置了一下,并且时钟由外部其他做,就暂时不由他产生了.



然后再新建一个PLL IP,即Clocking Wizard,生成74.25的像素时钟和371.25的TMDS编码时钟.



但是现在还要准备数据,即RGB数据,以RGB时序送入模块里,比如VSYNC,HSYNC生成模块.

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date: 2022/02/22 20:23:46
// Design Name:
// Module Name: vga_sync
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////

module vga_sync
    (
        output        hsync,
        output        vsync,
        output reg    ready,
        output [10:0] x_addr,
        output [10:0] y_addr,
        input         clk,
        input         rst
    );

parameter  H_FRONT_PROCH  =88;
parameter  H_SYNC_TIME    =44;
```

```
parameter  H_BACK_PROCH  =148;
parameter  H_ADDR_TIME    =1920;
parameter  H_TIME_TOTAL=H_FRONT_PROCH+H_SYNC_TIME+H_BACK_PROCH+H_ADDR_TIME;
parameter  H_ADDR_START_PIX  =H_FRONT_PROCH+H_SYNC_TIME;
parameter  H_ADDR_END_PIX    =H_FRONT_PROCH+H_SYNC_TIME+H_ADDR_TIME;

parameter  V_FRONT_PROCH  =4;
parameter  V_SYNC_TIME   =5;
parameter  V_BACK_PROCH  =36;
parameter  V_ADDR_TIME    =1080;
parameter  V_TIME_TOTAL=V_FRONT_PROCH+V_SYNC_TIME+V_BACK_PROCH+V_ADDR_TIME;
parameter  V_ADDR_START_PIX  =V_FRONT_PROCH+V_SYNC_TIME;
parameter  V_ADDR_END_PIX    =V_FRONT_PROCH+V_SYNC_TIME+V_ADDR_TIME;

reg [12:0] cnt_hreg; // HSYNC 计数器
reg [12:0] cnt_vreg; // VSYNC 计数器

// 产生HSYNC
always @(posedge clk, negedge rst)
begin
    if(!rst)
        cnt_hreg <= 'd0;
    else if(cnt_hreg == H_TIME_TOTAL-1)
        cnt_hreg <= 'd0;
    else
        cnt_hreg <= cnt_hreg + 'b1;
end

// 产生VSYNC
always @(posedge clk, negedge rst)
begin
    if(!rst)
        cnt_vreg <= 'd0;
    else if(cnt_vreg == V_TIME_TOTAL-1)
        cnt_vreg <= 'd0;
    else if(cnt_hreg == H_TIME_TOTAL-1)
        cnt_vreg <= cnt_vreg + 'b1;
end

// 数据有效区域
always @(posedge clk, negedge rst) begin
    if(!rst)
        ready <= 'b0;
    else if((cnt_hreg >= H_ADDR_START_PIX && cnt_hreg < H_ADDR_END_PIX) && (cnt_vreg >= V_ADDR_START_PIX && cnt_vreg <V_ADDR_END_PIX))
        ready <= 'b1;
    else
        ready <= 'b0;
end

assign hsync = (cnt_hreg < H_FRONT_PROCH) ? 'b0 : 'b1;
assign vsync = (cnt_vreg < V_FRONT_PROCH) ? 'b0 : 'b1;
assign x_addr = ready ? cnt_hreg - H_ADDR_START_PIX : 'd0;
assign y_addr = ready ? cnt_vreg - V_ADDR_START_PIX : 'd0;

endmodule
```

再做一个渐变色生成模块.

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date: 2022/02/22 21:00:29
// Design Name:
// Module Name: color_generate
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
```

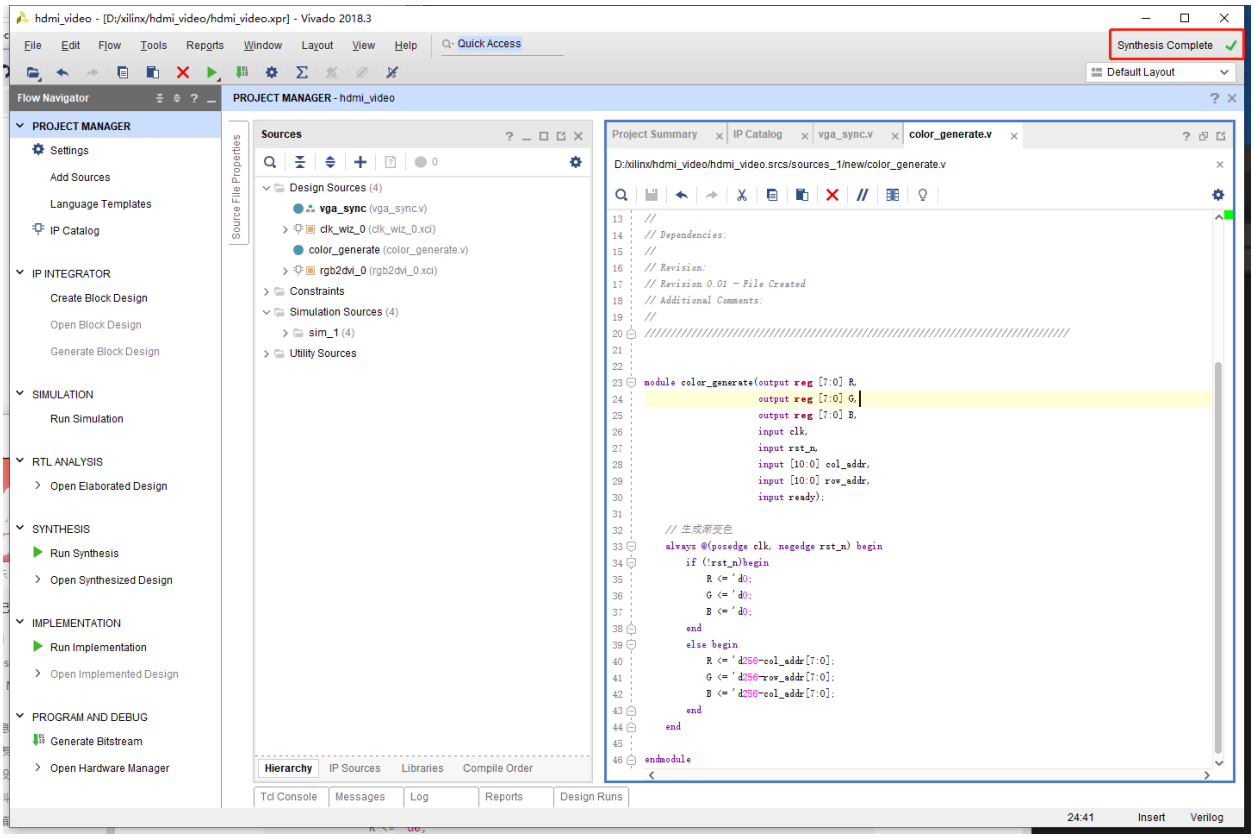
```
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////


module color_generate(output reg [7:0] R,
                      output reg [7:0] G,
                      output reg [7:0] B,
                      input clk,
                      input rst,
                      input [10:0] x_addr,
                      input [10:0] y_addr,
                      input ready);

    // 生成渐变色
    always @(posedge clk, negedge rst) begin
        if (!rst)
        begin
            R <= 'd0;
            G <= 'd0;
            B <= 'd0;
        end
        else
        begin
            R <= 'd256-x_addr[7:0];
            G <= 'd256-y_addr[7:0];
            B <= 'd256-x_addr[7:0];
        end
    end

endmodule
```

现在综合一下,确定没什么毛病.



然后做一个顶层的调用.

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date: 2022/02/22 21:11:17
// Design Name:
// Module Name: hdmi_video
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////

module hdmi_video (input clk, // 输入时钟
                   input rst, // 复位信号
                   output wire[2:0] TMDS_DATA_P1, // TMDS 数据
                   output wire[2:0] TMDS_DATA_N1, // TMDS 数据
                   output wire TMDS_CLK_P1, // TMDS 时钟
                   output wire TMDS_CLK_N1, // TMDS 时钟
                   output reg HDMI_OUT_EN // TMDS 使能
                   );

wire clk_74p25MHz; // 像素时钟
wire clk_371p25MHz; // TMDS时钟

// 同步信号
wire Hsync;
wire Vsync;

// 显示颜色
wire [7:0] Red;
wire [7:0] Green;
wire [7:0] Blue;

wire ready; // 数据有效

wire [10:0] x_addr; // X坐标
wire [10:0] y_addr; // Y坐标

always@(posedge clk or negedge rst)
begin
    if(!rst)
    begin
        HDMI_OUT_EN <= 'b0;
    end
    else
    begin
        HDMI_OUT_EN <= 'b1;
    end
end

// 时钟生成
clk_wiz_0 clk_wiz_0_inst(
    .clk_out1(clk_74p25MHz),
    .clk_out2(clk_371p25MHz),
    .clk_in1(clk),
    .resetn(rst)
);

rgb2dvi_0 rgb2dvi_0_inst(
    .TMDS_Clk_p(TMDS_CLK_P1),
    .TMDS_Clk_n(TMDS_CLK_N1),
    .TMDS_Data_p(TMDS_DATA_P1),
    .TMDS_Data_n(TMDS_DATA_N1),
    .aRst_n(1'b1), // 这里奇怪的是不能输入一个复位信号
    .vid_pData({Red, Blue, Green}),
    .vid_pVDE(ready),
    .vid_pHSync(Hsync),
```

```
    .vid_pVSync(Vsync),
    .PixelClk(clk_74p25MHz),
    .SerialClk(clk_371p25MHz)
);

vga_sync vga_sync_inst(
    .hsync(Hsync),
    .vsync(Vsync),
    .ready(ready),
    .x_addr(x_addr),
    .y_addr(y_addr),
    .clk(clk_74p25MHz),
    .rst(rst)
);

color_generate color_generate_inst(
    .R(Red),
    .G(Green),
    .B(Blue),
    .clk(clk_74p25MHz),
    .rst(rst),
    .x_addr(x_addr),
    .y_addr(y_addr),
    .ready(ready)
);

endmodule
```

然后设置管脚约束,除了图形化配置,其实也可以写xdc文件,比如如下格式.

```
set_property IOSTANDARD LVCMOS33 [get_ports clk]
set_property IOSTANDARD LVCMOS33 [get_ports HDMI_OUT_EN]
set_property IOSTANDARD LVCMOS33 [get_ports rst]
set_property IOSTANDARD TMDS_33 [get_ports {TMDS_DATA_P1[2]}]
set_property IOSTANDARD TMDS_33 [get_ports {TMDS_DATA_P1[1]}]
set_property IOSTANDARD TMDS_33 [get_ports {TMDS_DATA_P1[0]}]
set_property IOSTANDARD TMDS_33 [get_ports TMDS_CLK_P1]
set_property PACKAGE_PIN K17 [get_ports clk]
set_property PACKAGE_PIN F17 [get_ports HDMI_OUT_EN]
set_property PACKAGE_PIN M19 [get_ports rst]
set_property PACKAGE_PIN B19 [get_ports {TMDS_DATA_P1[2]}]
set_property PACKAGE_PIN C20 [get_ports {TMDS_DATA_P1[1]}]
set_property PACKAGE_PIN D19 [get_ports {TMDS_DATA_P1[0]}]
set_property PACKAGE_PIN H16 [get_ports TMDS_CLK_P1]
```

最后当然是写入到板子上了.