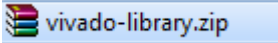
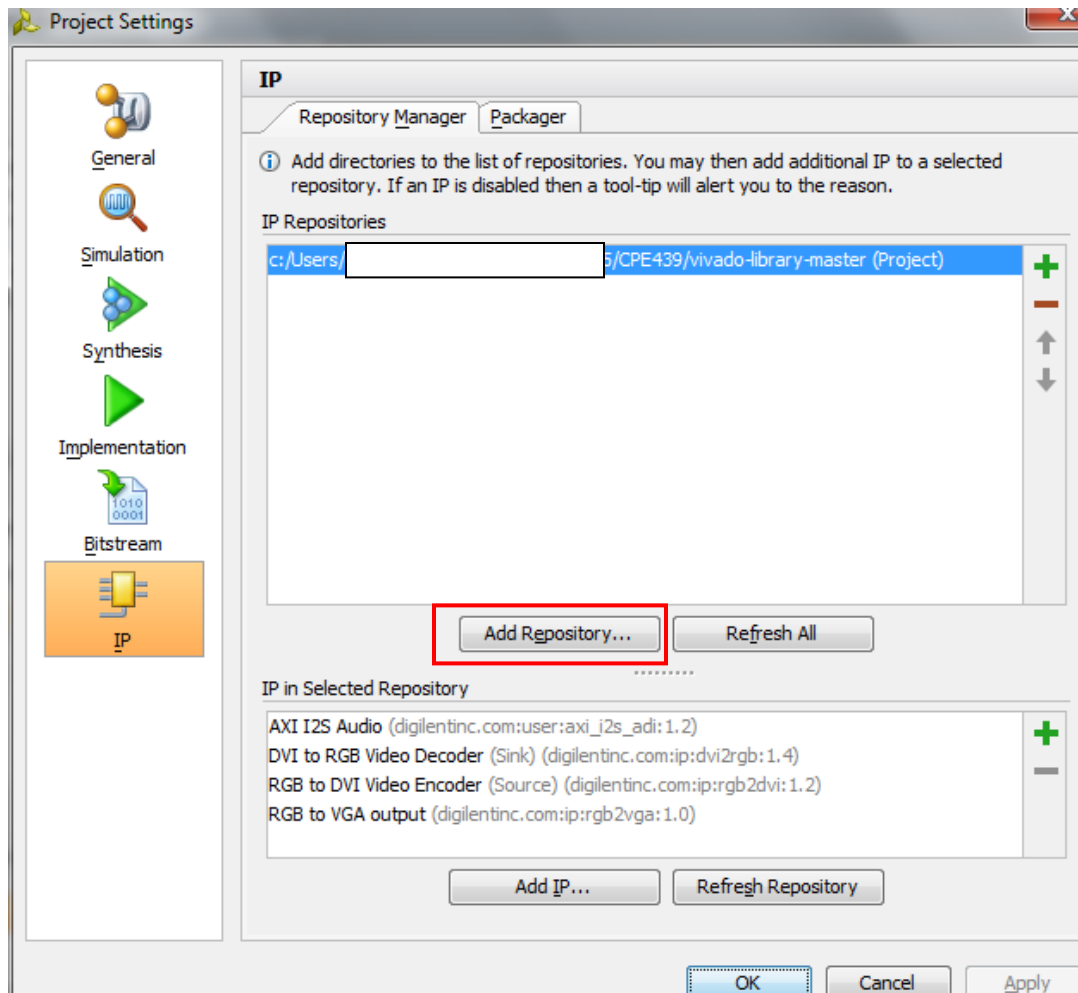


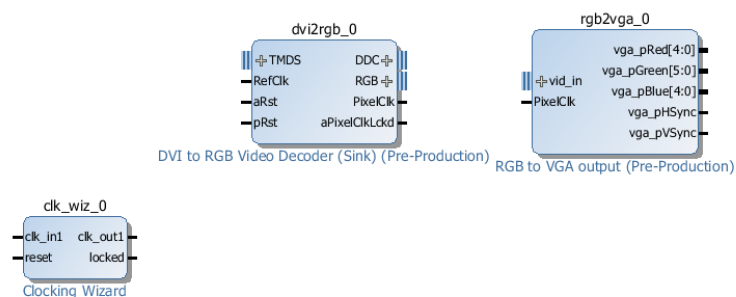
Run Bin Yu
CPE 439 Spring 2015

IP folder zip file:  Be sure to unzip all files in the same folder, and add the unzip folder to the project.

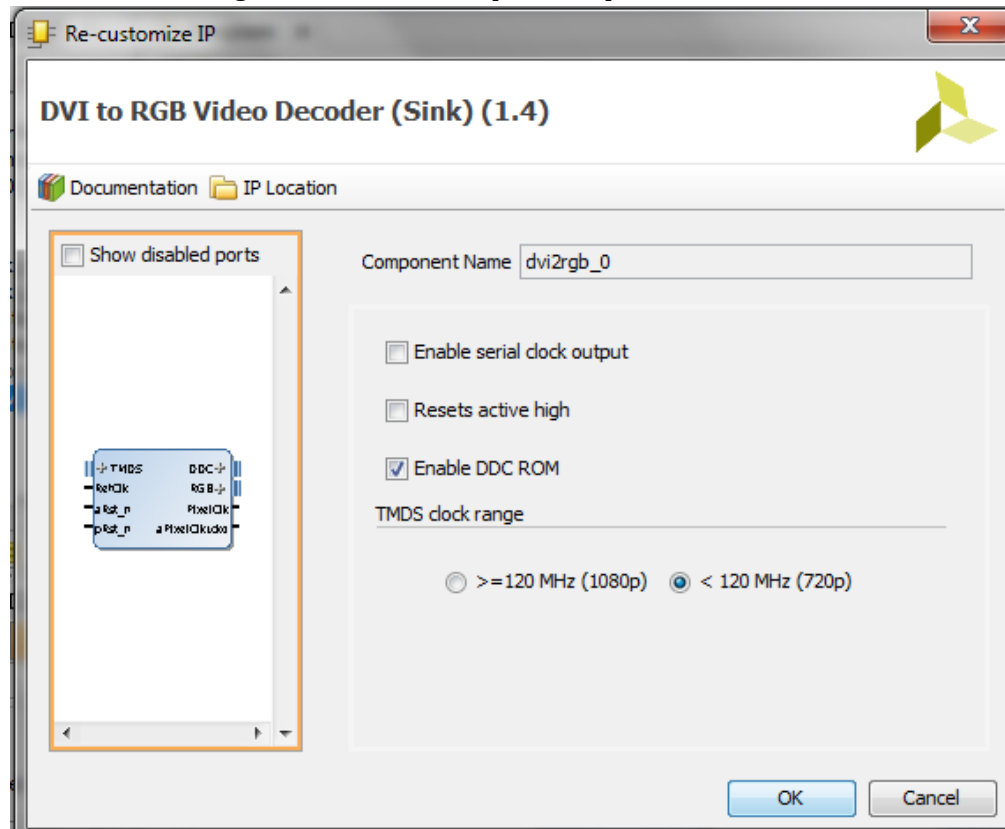


HDMI Input setup:

1. Add “dvi2rgb”, “clock wizard” and “rgb2vga” to the block diagram.

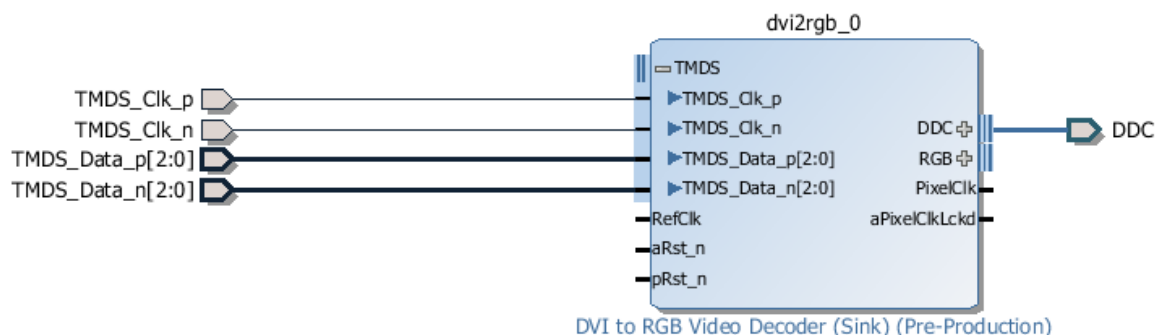


2. Double click dvi2rgb to customize the ip, set it up as shown below.



It is recommended to choose <120MHz(720p) since Zybo has $F_{BUF\ MAX} = 600\text{MHz}$ so the maximum serial clock (which is 5x Pixel clock) is 600MHz and as a result the Pixel clock could only goes up to 120MHz. Note that 1080p HDMI resolution requires 148.5MHz pixel clock which is out of this range. Select $\geq 120\text{MHz}$ might cause errors.

3. Click on the "+" sign of "TMDS" interface on the dvi2rgb ip, right click to make an external for each port of this interface.
Right click "DDC" interface and make an external for this interface.



4. Customize “Clock wizard” as shown below.

Component Name: clk_wiz_0

Clocking Options | Output Clocks | PLL2 Settings | Port Renaming | Summary

Primitive: ☐ MMCM ☒ PLL

Clocking Features | Jitter Optimization

☒ Frequency Synthesis ☐ Minimize Power ☒ Balanced

☒ Phase Alignment ☐ Minimize Output Jitter

☐ Dynamic Reconfig ☐ Maximize Input Jitter filtering

☐ Safe Clock Startup

Dynamic Reconfig Interface Options

☒ AXI4Lite ☐ DRP ☐ Phase Duty Cycle Config

Input Clock Information

	Input Clock	Input Frequency(MHz)		Jitter Options	Input Jitter	Source
<input checked="" type="checkbox"/>	Primary	<input type="text" value="Auto"/> 100.000	19.000 - 800.000	UI	0.010	Single e
<input type="checkbox"/>	Secondary	<input type="text" value="Auto"/> 100.000	100.000 - 200.000		0.010	Single e

Make sure to choose “PLL” primitive since dvi2rgb uses “MMCM” primitive to decode TMDS signals and Zybo has only one “MMCM” primitive.

“Input Clock” is set to “Auto”.

Component Name: clk_wiz_0

Clocking Options | **Output Clocks** | PLL2 Settings | Port Renaming | Summary

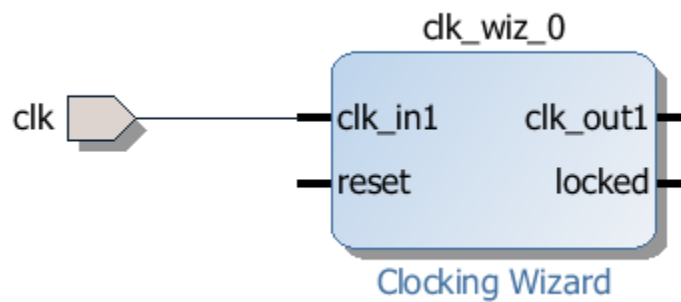
The phase is calculated relative to the active input clock.

Output Clock	Output Freq (MHz)		Phase (degrees)		Duty Cycle (%)
	Requested	Actual	Requested	Actual	
<input checked="" type="checkbox"/> clk_out1	200.000	200.000	0.000	0.000	50.000
<input type="checkbox"/> clk_out2	100.000	N/A	0.000	N/A	50.000
<input type="checkbox"/> clk_out3	100.000	N/A	0.000	N/A	50.000

“Output Clocks” set to 200MHz, which provides dvi2rgb with “RefClk”.

- For “clk_in1” of the clock wizard, you could either connect it to ps7’s FCLK0 or Zybo’s built-in 125MHz clock.

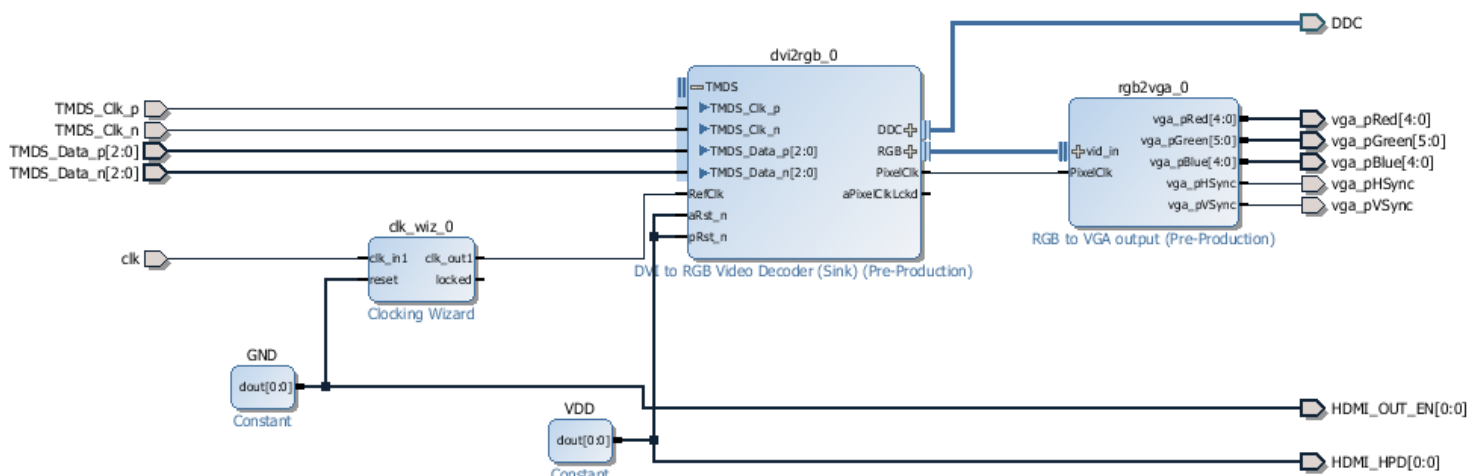
To use the built-in 125MHz clock, create an external for “clk_in1”. Then we could map this external port “clk” to the one on “ZYBO_Master.xdc”.



```
##Clock signal
##IO_L11P_T1_SRCC_35
set_property PACKAGE_PIN L16 [get_ports clk]
set_property IOSTANDARD LVCMOS33 [get_ports clk]
create_clock -add -name sys_clk_pin -period 8.00 -waveform {0 4} [get_ports clk]
```

Make sure clock wizard still has “PLL” as primitive.

- Finish all connections shown below, make constants of GND and VDD (value of 0 and 1) and ports called “HDMI_OUT_EN” and “HDMI_HPD” (hot-plug detect).



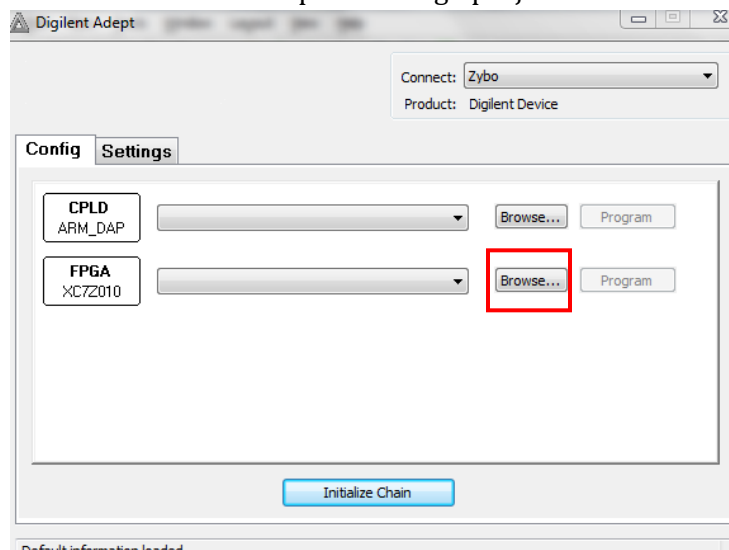
7. Create wrapper from the block diagram. Note that DDC interface now generates two “inout” in the wrapper.

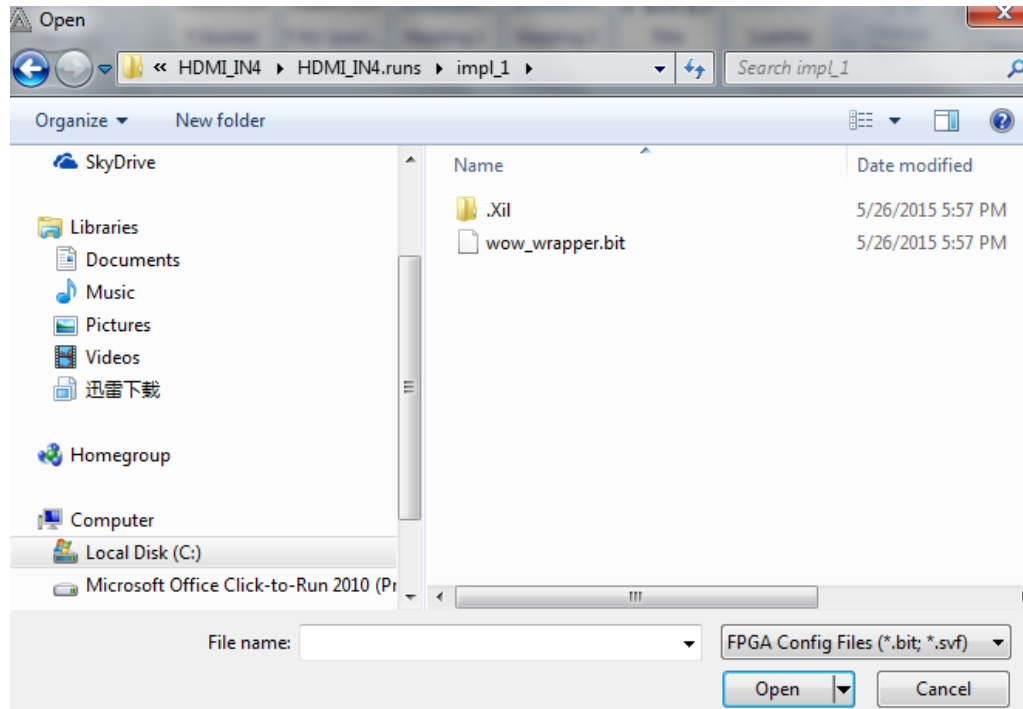
```
inout ddc_scl_io;  
inout ddc_sda_io;
```

Be sure to match HDMI’s SCL and SDA ports on “ZYBO_Master.xdc” with “ddc_scl_io” and “ddc_sda_io”.

Also, make sure to match all other port names shown on the block diagram.

8. Generate bitstream. After finish, turns on Zybo and open Adept 2 (no need to use SDK since this is a data pass-through project which does not require processing).



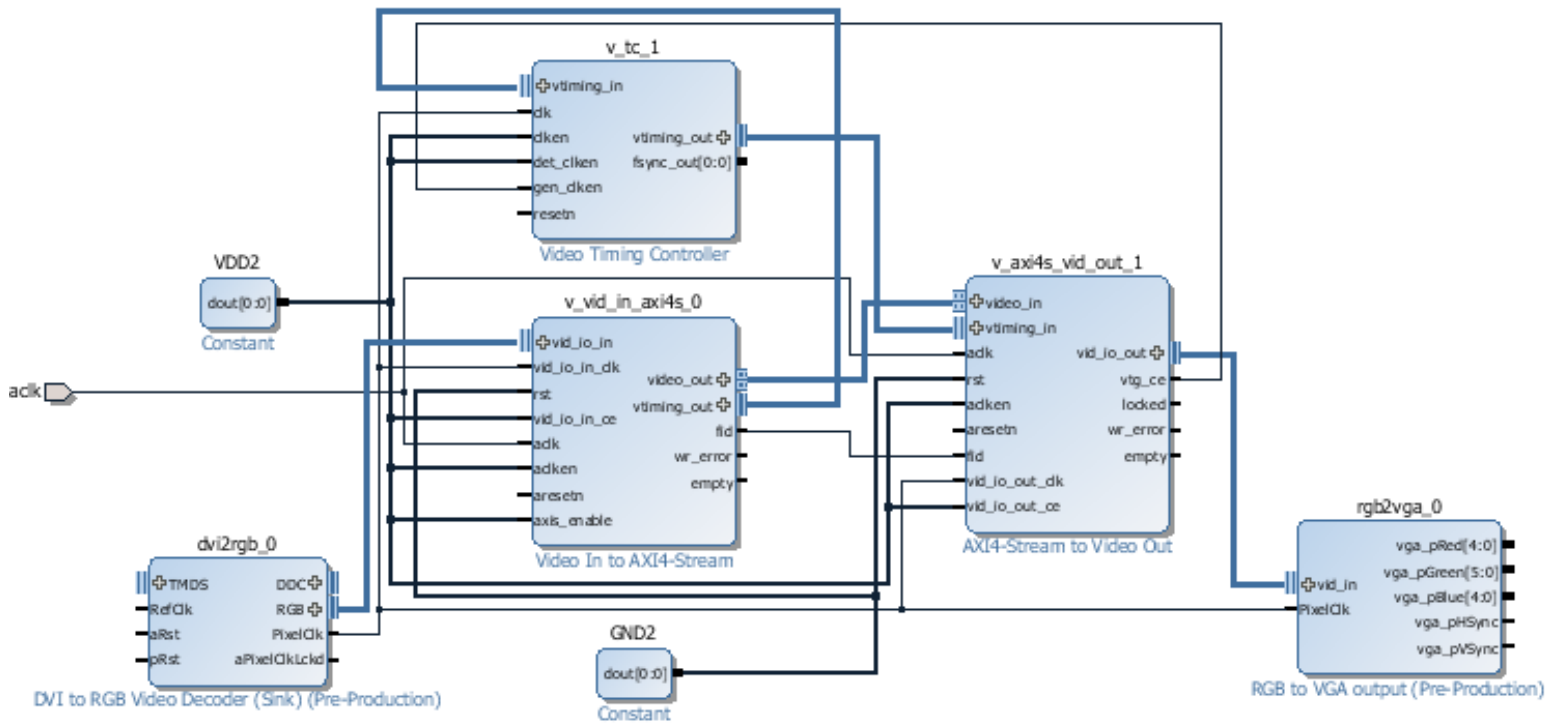


Select the bit file under the file path shown above, and program the FPGA.

Actual resolution might depend on the HDMI source. Zybo has its “default resolution” of 1280x1024@60Hz which generates 108MHz pixel clock. You could verify this by probing the clock signal (the 4th pair differential signal from left to right) on the HDMI connector, which should be 108MHz sin-wave. Or alternatively map the output pixel clock signal to one of the LED to gain an easier access.

You might need to unplug and plug the HDMI cable to enable the input.

In projects with AXI bus stream, connect dvi2rgb’s video signals and pixel clock signal to a vid_in_axi4s block which converts the video and timing signals into axi bus stream format. Extra Video Timing Controller (configured to detect and support 1280x1024p resolution) might be needed to ensure all blocks who receive the video stream to be synchronized with the input signals.



Note that `ack` is the axi stream clock, it could be the axi bus clock or other. But it is recommended to be higher or at least equal to pixel clock to reduce buffer size. Also note that all reset/`resetn`, `areset`/`aresetn` and `en/ce` signals are application-dependent (except `vtg_ce` → `gen_clken` in many cases, is required).

HDMI Output (with Test Pattern Generator)

1. Since Zybo only has one HDMI connector, to test the HDMI output, Test Pattern Generator (tpg) is used to generate the video. Add tpg block, set its parameters as shown below.

The screenshot shows the configuration window for the Test Pattern Generator (TPG) component. The component name is set to `v_tpg_0`. Under the "Optional Features" section, the "AXI4-Lite Register Interface" checkbox is highlighted with a red box and is unchecked. Other optional features like "Enable AXI4-Stream Slave Interface", "Enable Video Timing Interface", and "Enable INTC Ports" are also unchecked. The "Test Pattern Generator Options" section includes "Input Video Format" set to "YCbCr422", "Output Video Format" set to "RGB", "Enable Motion" unchecked, "Motion Speed" set to 0, "Test Pattern" set to "Color Bars", and "Bayer Phase" set to "4 None". The "Input Frame Dimensions" section shows "Number of Pixels per Scanline (Default)" set to 1280 and "Number of Scanlines per Frame (Default)" set to 720, both highlighted with a red box. The range for both is [32 - 7680].

Note that 1280x720 (720p) resolutions is chosen in this experiment. Test pattern could be any option except passthrough (since we don't have video input to pass through). We are not going to do any processing on the video and run the hardware without SDK, so make sure AXI4 Interface option is unchecked.

2. Set rgb2dvi block to this configuration:

The screenshot shows the configuration window for the RGB to DVI Video Encoder (Source) (1.2) component. The component name is set to `rgb2dvi_0`. The "Reset active high" and "Generate SerialClk internally from pixel clock." checkboxes are checked. Under the "MMCM/PLL" section, the "MMCM" radio button is selected. The "TMDs clock range" section shows the "< 120 MHz (720p)" radio button selected. On the left, a port diagram shows the component with inputs for RGB, Rst, and Kxclk, and an output for TMDs. The "Show disabled ports" checkbox is checked. The window has "OK" and "Cancel" buttons at the bottom right.

3. Add Video Timing Controller (vtc) block and set it up as shown below.
vtc is set generate timing signals to support 720p and runs forever, so we just need clock generation without detection. Also, AXI4 Interface is not needed in this case.

Component Name: v_tc_0

Detection/Generation | Default/Constant | Frame Sync Position

Optional Features

- ☐ Include AXI4-Lite Interface
- ☐ Include INTC Interface
- ☐ Interlaced Video Support
- ☐ Synchronize Generator to Detector or to fsync_in

Max Clocks Per Line: 4096 | Max Lines Per Frame: 4096

Frame Syncs: 1

☒ Enable Generation | ☐ Enable Detection

Generation Options	Detection Options
<input type="checkbox"/> Field ID Generation	<input type="checkbox"/> Field ID Detection
<input checked="" type="checkbox"/> Vertical Blank Generation	<input checked="" type="checkbox"/> Vertical Blank Detection
<input checked="" type="checkbox"/> Horizontal Blank Generation	<input checked="" type="checkbox"/> Horizontal Blank Detection
<input checked="" type="checkbox"/> Vertical Sync Generation	<input checked="" type="checkbox"/> Vertical Sync Detection
<input checked="" type="checkbox"/> Horizontal Sync Generation	<input checked="" type="checkbox"/> Horizontal Sync Detection
<input checked="" type="checkbox"/> Active Video Generation	<input checked="" type="checkbox"/> Active Video Detection

Component Name: v_tc_0

Detection/Generation | **Default/Constant** | Frame Sync Position

Video Format

Video Mode: 720p

Horizontal Settings

4. Add clock wizard to generate a 75MHz pixel clock (720p), be sure to select PLL as primitive.

5. Add AXI4 Stream to Video block (v_axi4s_vid_out), set up is shown below.

The screenshot shows the configuration window for the component 'v_axi4s_vid_out_0'. The 'Timing Mode' section has 'Slave' selected. The 'Pixels Per Clock' is set to 1. The 'Video Component Width' is set to 8. The 'Video Format' is set to RGB. The 'FIFO Depth' has radio buttons for 32, 1024 (selected), 2048, 4096, and 8192. The 'Hysteresis Level' is set to 12, with a range of [8 - 1008] indicated.

Component Name	v_axi4s_vid_out_0	
Timing Mode		
<input checked="" type="radio"/> Slave <input type="radio"/> Master		
Pixels Per Clock	1	
<input type="radio"/> Auto	Video Component Width	8
<input type="radio"/> Auto	Video Format	RGB
FIFO Depth		
<input type="radio"/> 32 <input checked="" type="radio"/> 1024 <input type="radio"/> 2048 <input type="radio"/> 4096 <input type="radio"/> 8192		
Hysteresis Level	12	[8 - 1008]

6. Connect all blocks as shown below. Note that individual connections between blocks without using interface (bold blue connection) are used for debugging purposes. LEDs are added to gain accesses to the timing signals in cases that require debugging checks and you are free to connect them to other interested points (except the TMDSSs).
7. After creating wrapper file and matching all ports on ZYBO_Master.xdc, generate bitstream and use the same method of uploading the bit file to zybo using Adept 2. Connect HDMI output to the display and see if pattern is shown on the screen.

