

Project - Bee Invaders

Tutorial 1: Fill A VGA Screen With One Colour

This Tutorial Is Specifically For The Digilent Basys 3 Board



Proposed Game

Score 00000

Lives 3

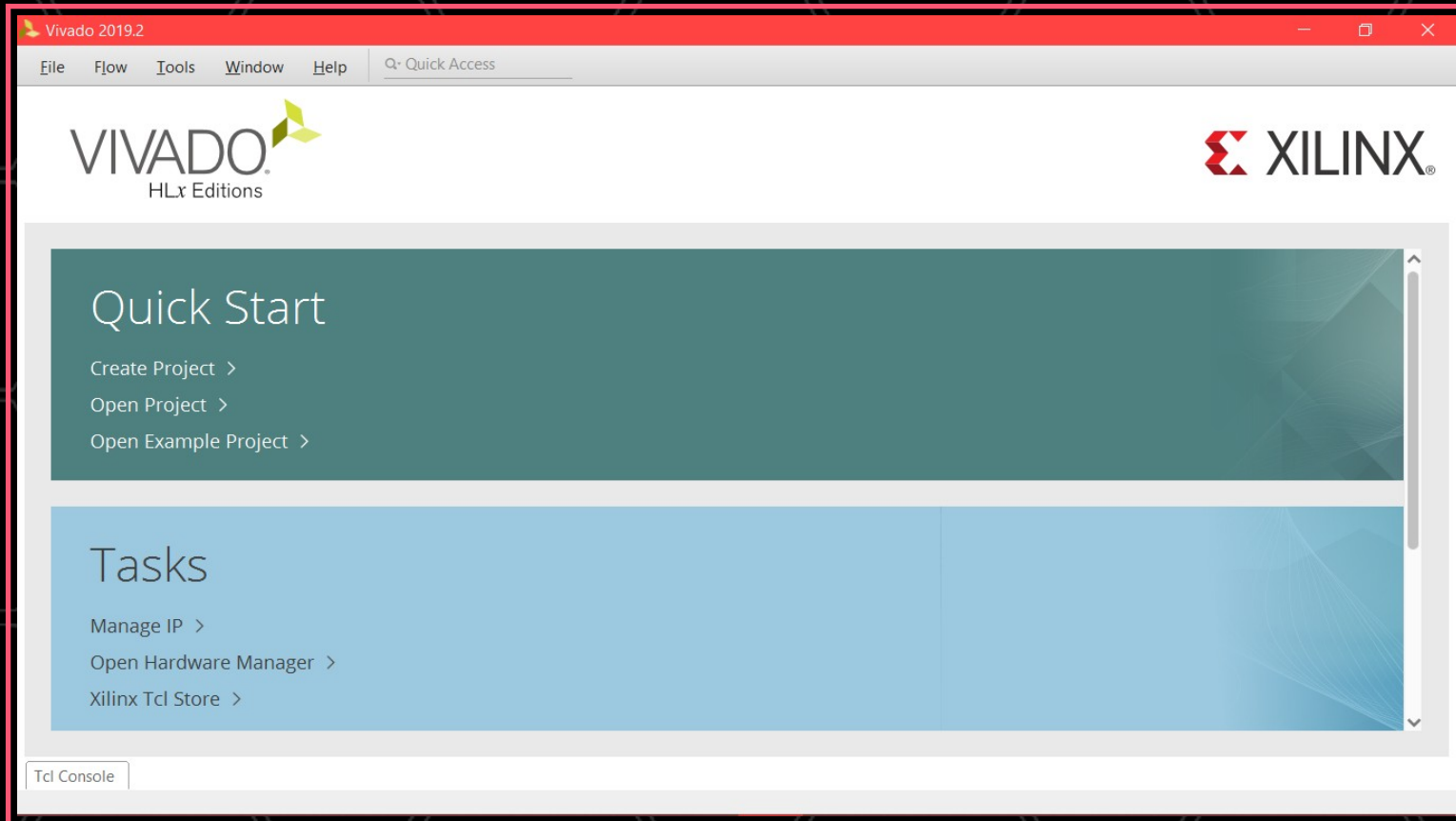


Instructions

01

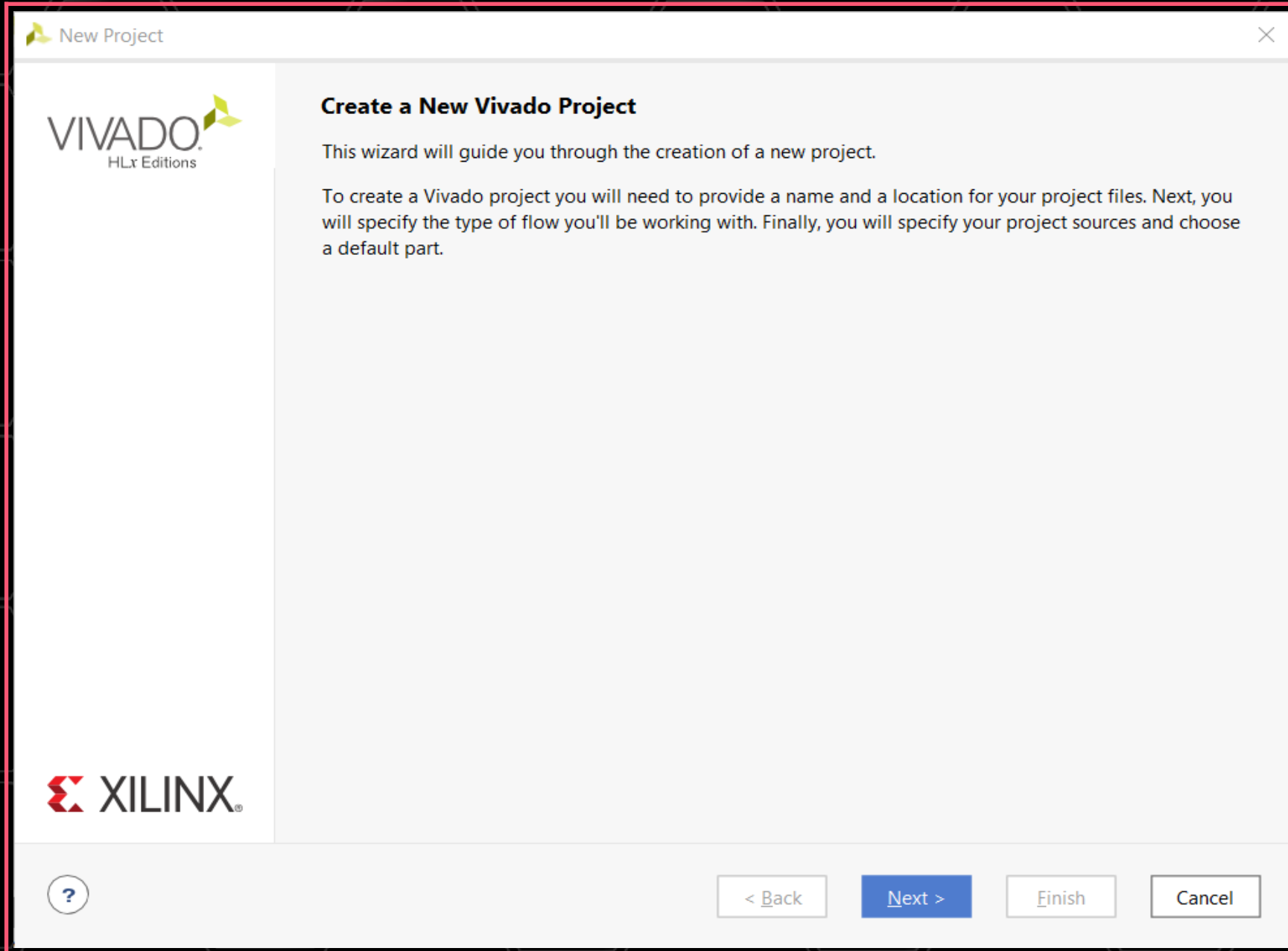
Create a folder on your Windows Desktop called "BeeInvaders" and inside the folder create a folder called "Tutorials Basys 3"

Download from Xilinx website the free software "Vivado Design Suite: Self Extracting Web Installer / WebPack Edition" (version 2019.2 in this case). Disable any Antivirus software during installation. Run Vivado 2019.2 and you will see a screen similar to this;



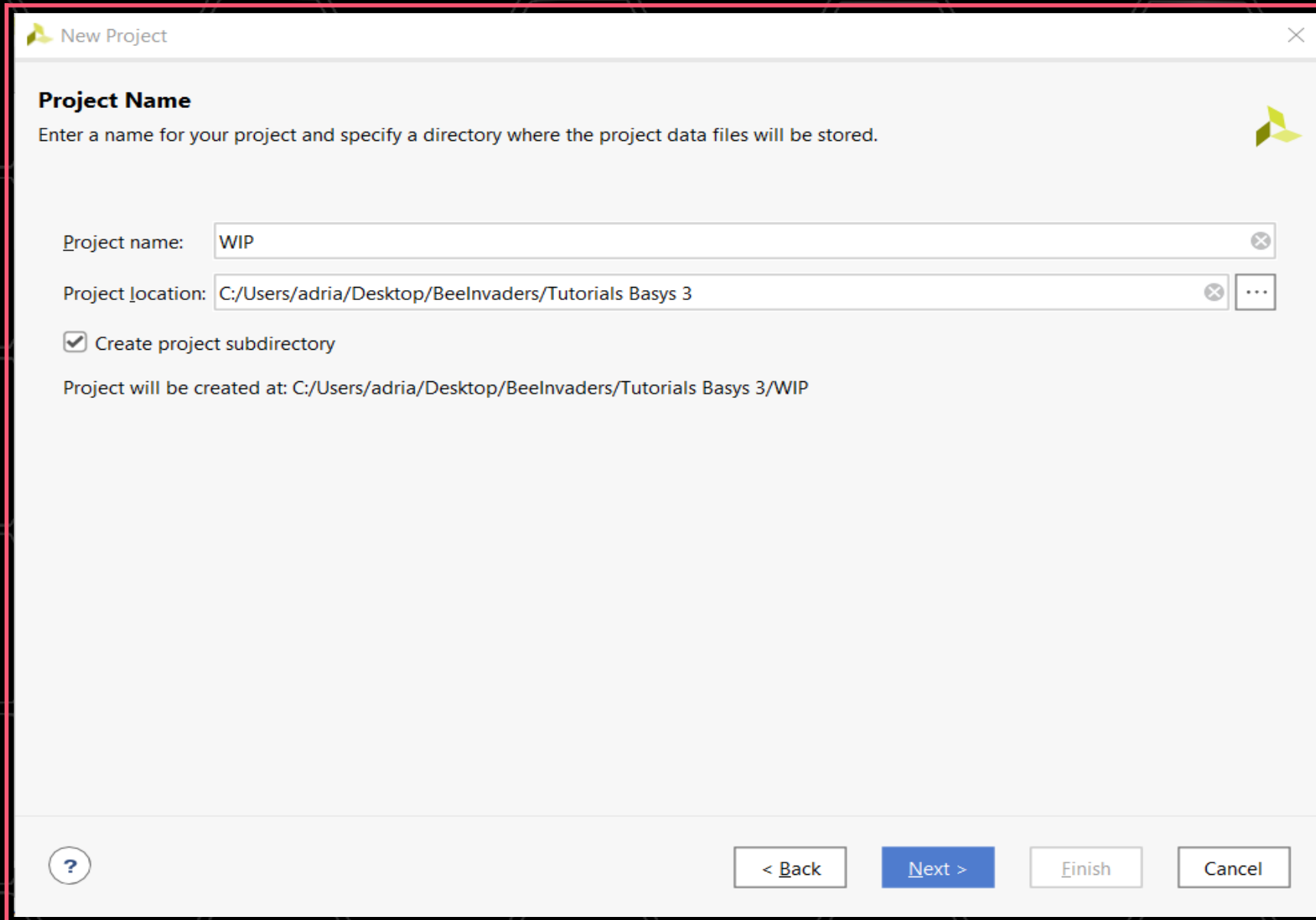
02

Click "Create Project" and when the below screen appears click "Next"



03

Enter "WIP" in the "Project name:" box and change the path for the "Project location:" box to the "BeeInvaders/Tutorials Basys 3" folder you created in step 1. Tick the box entitled "Create project subdirectory" and click "Next"



New Project

Project Name

Enter a name for your project and specify a directory where the project data files will be stored.

Project name: WIP

Project location: C:/Users/adria/Desktop/BeelInvaders/Tutorials Basys 3

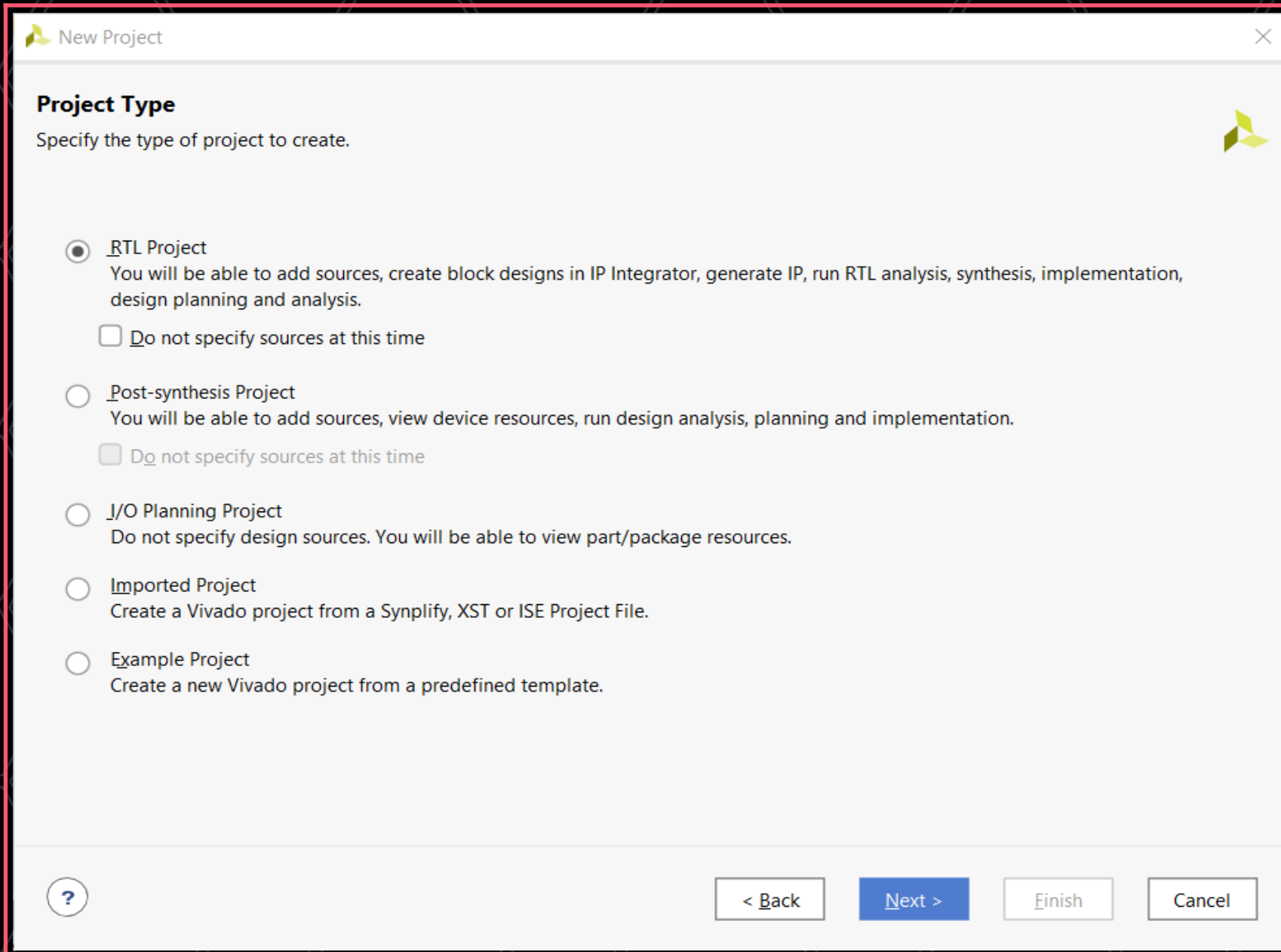
☒ Create project subdirectory

Project will be created at: C:/Users/adria/Desktop/BeelInvaders/Tutorials Basys 3/WIP

? < Back Next > Finish Cancel

04

Select "RTL Project" and click "Next"



New Project

Project Type


Specify the type of project to create.


- ☒ **RTL Project**
You will be able to add sources, create block designs in IP Integrator, generate IP, run RTL analysis, synthesis, implementation, design planning and analysis.
☐ Do not specify sources at this time
- ☐ **Post-synthesis Project**
You will be able to add sources, view device resources, run design analysis, planning and implementation.
☐ Do not specify sources at this time
- ☐ **I/O Planning Project**
Do not specify design sources. You will be able to view part/package resources.
- ☐ **Imported Project**
Create a Vivado project from a Synplify, XST or ISE Project File.
- ☐ **Example Project**
Create a new Vivado project from a predefined template.

? < Back Next > Finish Cancel

05

Click "Next" at the screen below

 New Project ×

Add Sources 

Specify HDL, netlist, Block Design, and IP files, or directories containing those files, to add to your project. Create a new source file on disk and add it to your project. You can also add and create sources later.

+

-

↑

↓

Use Add Files, Add Directories or Create File buttons below

Add Files

Add Directories

Create File

☐ Scan and add RTL include files into project

☐ Copy sources into project

☒ Add sources from subdirectories

Target language: Verilog

Simulator language: Mixed

?

< Back


Next >

Finish


Cancel





06

Click "Next" at the screen below

 New Project ×

Add Constraints (optional)
Specify or create constraint files for physical and timing constraints.




Use Add Files or Create File buttons below

Add FilesCreate File

☐ Copy constraints files into project



< Back


Next >


Finish

Cancel

07

When the below screen appears click the "Boards" tab

 New Project ✕

Default Part 

Choose a default Xilinx part or board for your project.

Parts | Boards

[Reset All Filters](#)

Category:

Package:

Temperature:

Family:

Speed:

Static power:

Search:

Part	I/O Pin Count	Available IOBs	LUT Elements	FlipFlops	Block RAMs	Ultra RAMs	DSPs	Gb Tra
xc7k70tfbv484-2L	484	285	41000	82000	135	0	240	4
xc7k70tfbv484-1	484	285	41000	82000	135	0	240	4
xc7k70tfbv676-3	676	300	41000	82000	135	0	240	8
xc7k70tfbv676-2	676	300	41000	82000	135	0	240	8
xc7k70tfbv676-2L	676	300	41000	82000	135	0	240	8
xc7k70tfbv676-1	676	300	41000	82000	135	0	240	8
xc7k70tlfbg484-2L	484	285	41000	82000	135	0	240	4

?

< Back


Next >

Finish

Cancel

08

Select the "Basys3" board and click "Next" at the screen below

 New Project ×

Default Part
Choose a default Xilinx part or board for your project.

Parts | **Boards**

[Reset All Filters](#) Update Board Repositories

Vendor: All Name: All Board Rev: Latest

Search: Q

Display Name	Preview	Vendor	File Version	Part	I/O
Arty		digilentinc.com	1.1	xc7a35ticsg324-1L	32
Basys3		digilentinc.com	1.1	xc7a35tcp236-1	23
Cmod S7-25		digilentinc.com	1.0	xc7s25csga225-1	22
Cmod A7-15t					

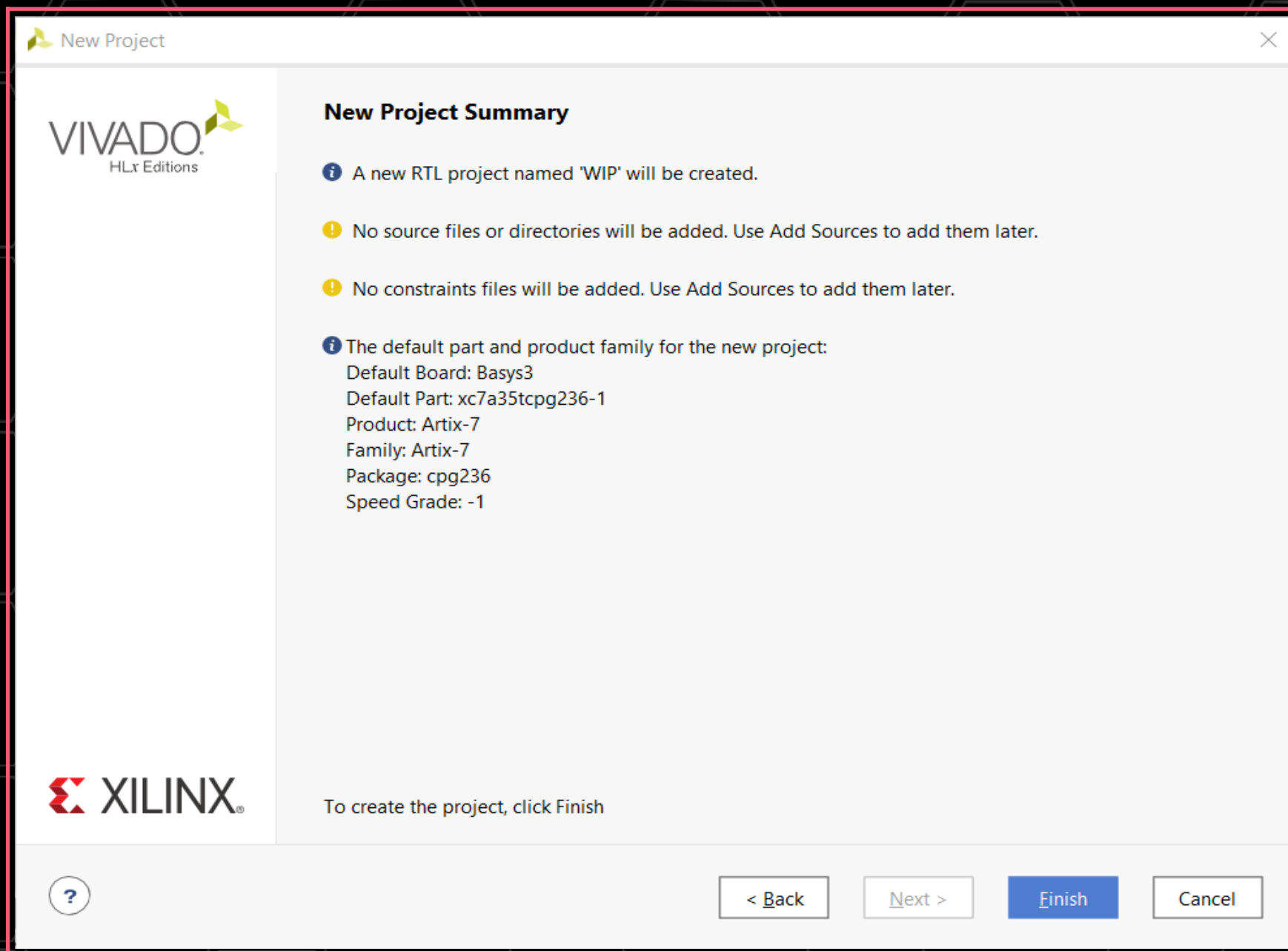
< >

?

< Back Next > Finish Cancel

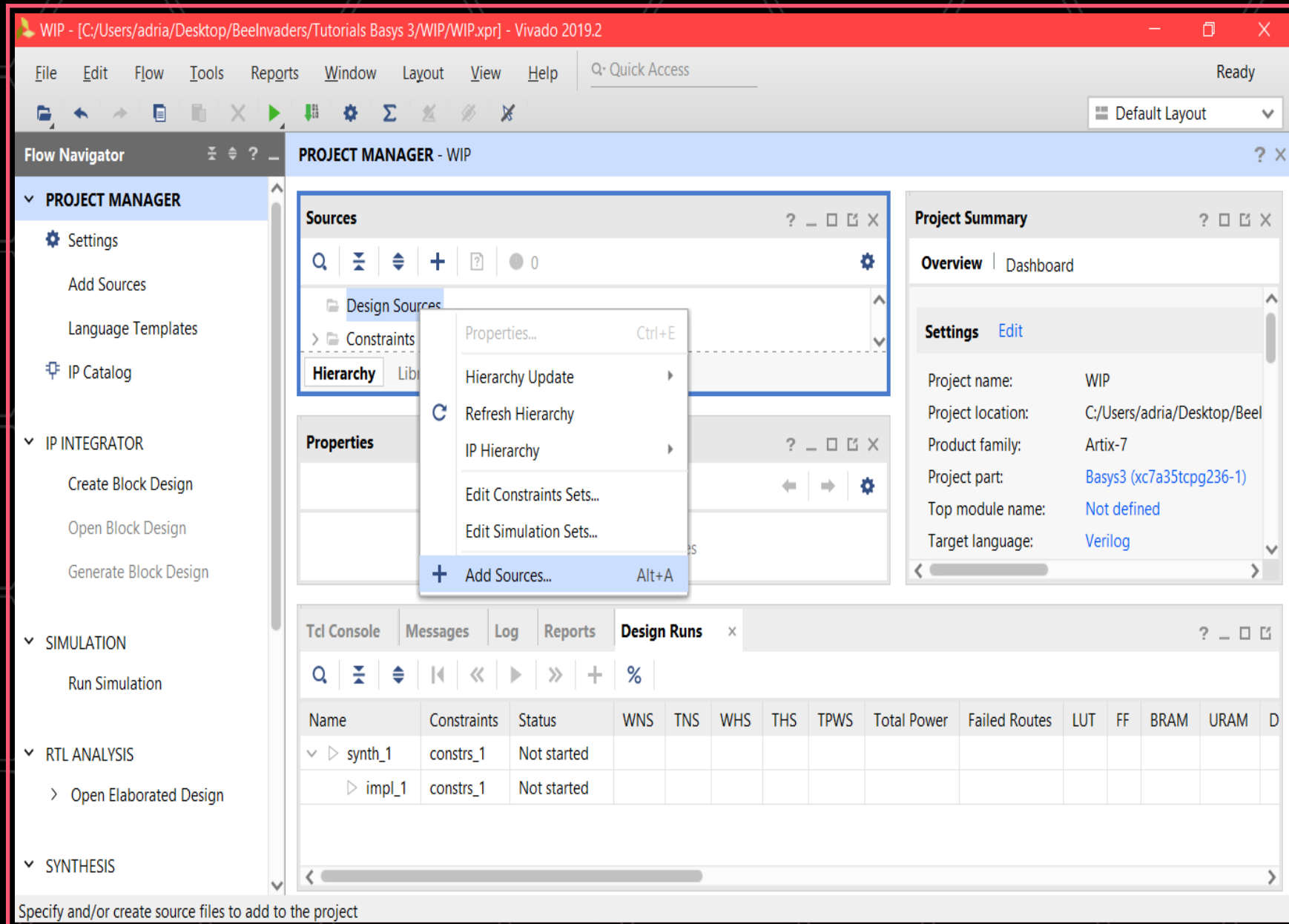
09

Click "Finish" at the screen below



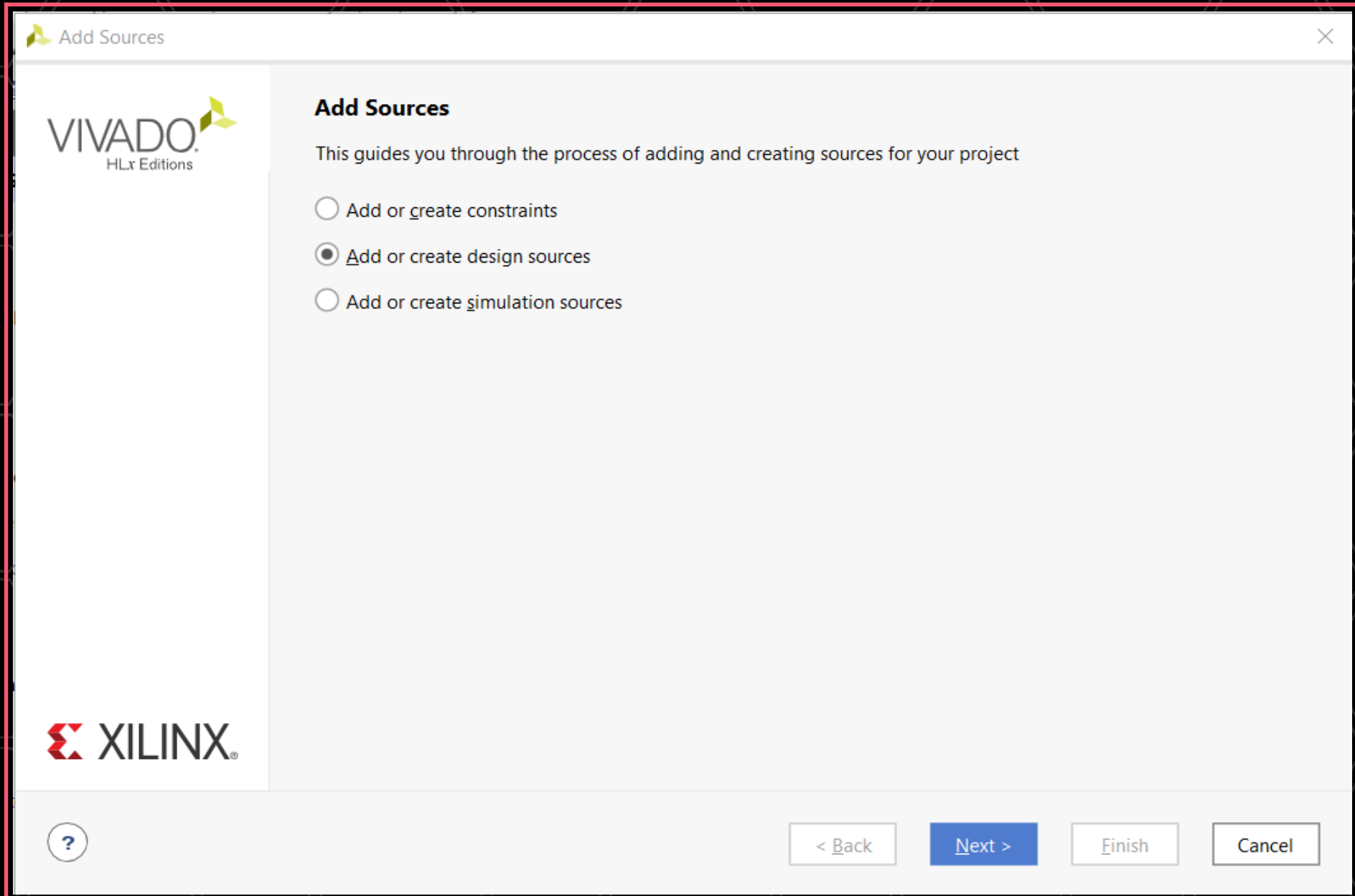
11

Right click on "Design Sources" and left click on "Add Sources" as shown below



12

Select "Add or create design sources" and click "Next"



The image shows a screenshot of the 'Add Sources' dialog box in the Vivado IDE. The dialog has a title bar with the text 'Add Sources' and a close button. On the left side, there is a sidebar with the Vivado logo and the text 'HLx Editions' at the top, and the Xilinx logo at the bottom. The main area of the dialog is titled 'Add Sources' and contains a descriptive text: 'This guides you through the process of adding and creating sources for your project'. Below this text, there are three radio button options: 'Add or create constraints', 'Add or create design sources' (which is selected), and 'Add or create simulation sources'. At the bottom of the dialog, there is a navigation bar with a help icon (a question mark in a circle) on the left, and four buttons on the right: '< Back', 'Next >' (highlighted in blue), 'Finish', and 'Cancel'.

Add Sources

This guides you through the process of adding and creating sources for your project

- ☐ Add or create constraints
- ☒ Add or create design sources
- ☐ Add or create simulation sources

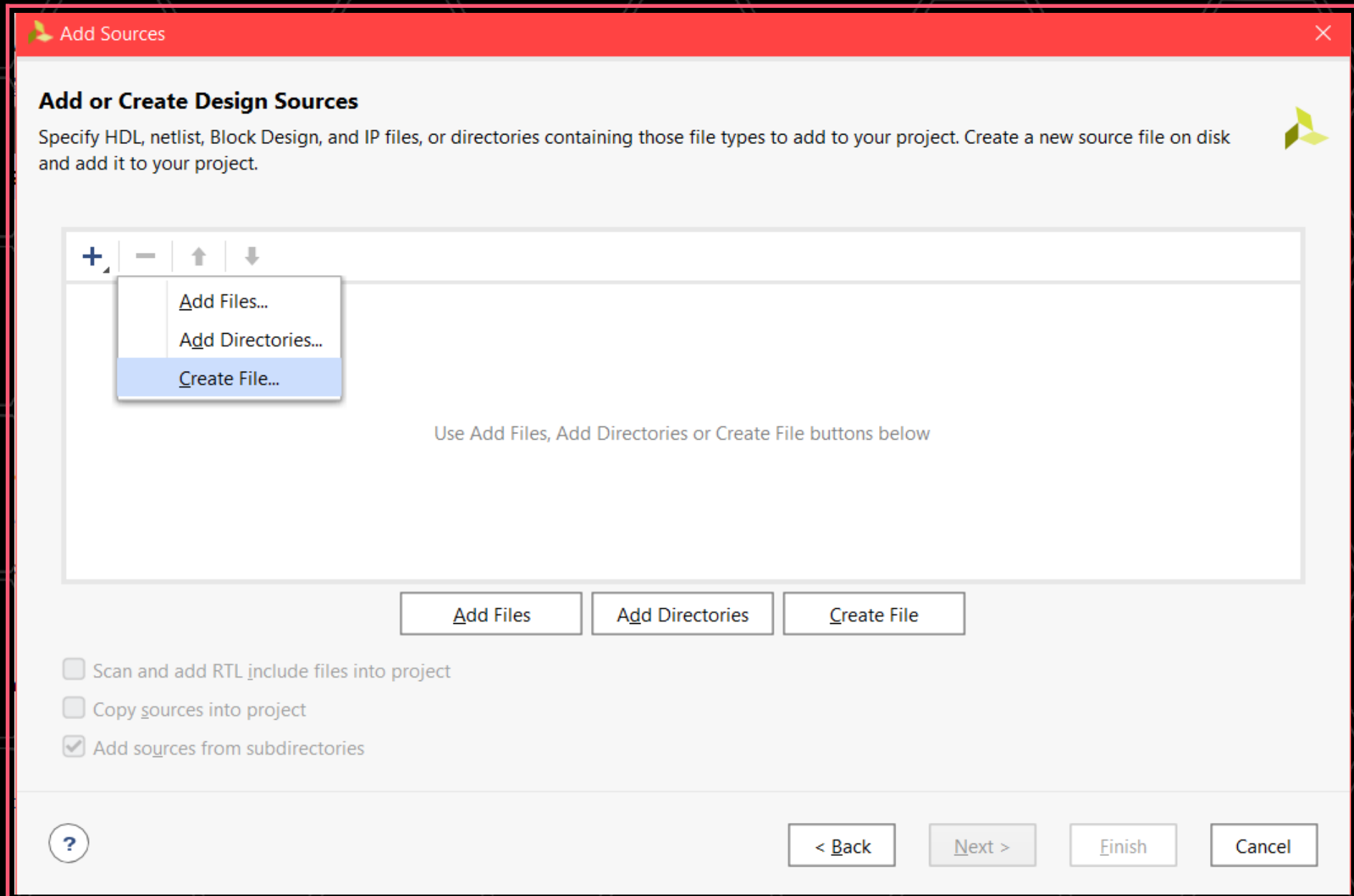
VIVADO
HLx Editions

XILINX

? < Back Next > Finish Cancel

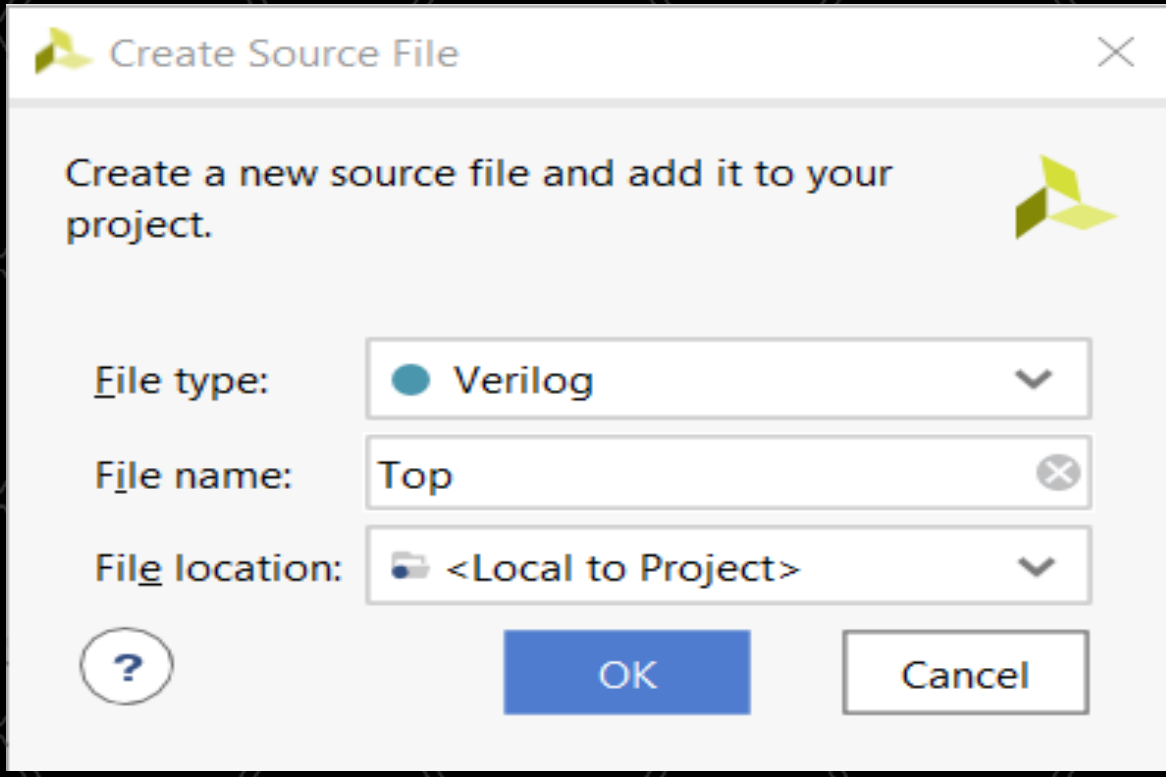
13

Select "+" and click on "Create File" or click on the "Create File" button



14

Make sure "Verilog" is the "File Type:", enter "Top" in the box entitled "File name:", ensure "Local to Project" is the "File location:" and click "OK"



The image shows a "Create Source File" dialog box with a white background and a thin red border. At the top left is a yellow logo, and at the top right is a close button (X). Below the title bar, the text "Create a new source file and add it to your project." is displayed next to a yellow logo. The dialog contains three input fields: "File type:" with a dropdown menu showing "Verilog" and a blue circle icon; "File name:" with a text box containing "Top" and a clear button (X); and "File location:" with a dropdown menu showing "<Local to Project>" and a folder icon. At the bottom left is a help button (question mark in a circle). At the bottom right are two buttons: "OK" (blue) and "Cancel" (white with a grey border).

Create Source File

Create a new source file and add it to your project.

File type: Verilog

File name: Top


File location: <Local to Project>


?

OK Cancel





15


Select "Finish" at the next screen

 Add Sources ×

Add or Create Design Sources 

Specify HDL, netlist, Block Design, and IP files, or directories containing those file types to add to your project. Create a new source file on disk and add it to your project.

	Index	Name	Library	Location
	1	Top.v	xil_defaultlib	<Local to Project>

Add Files


Add Directories

Create File

☐ Scan and add RTL include files into project

☐ Copy sources into project

☒ Add sources from subdirectories



< Back

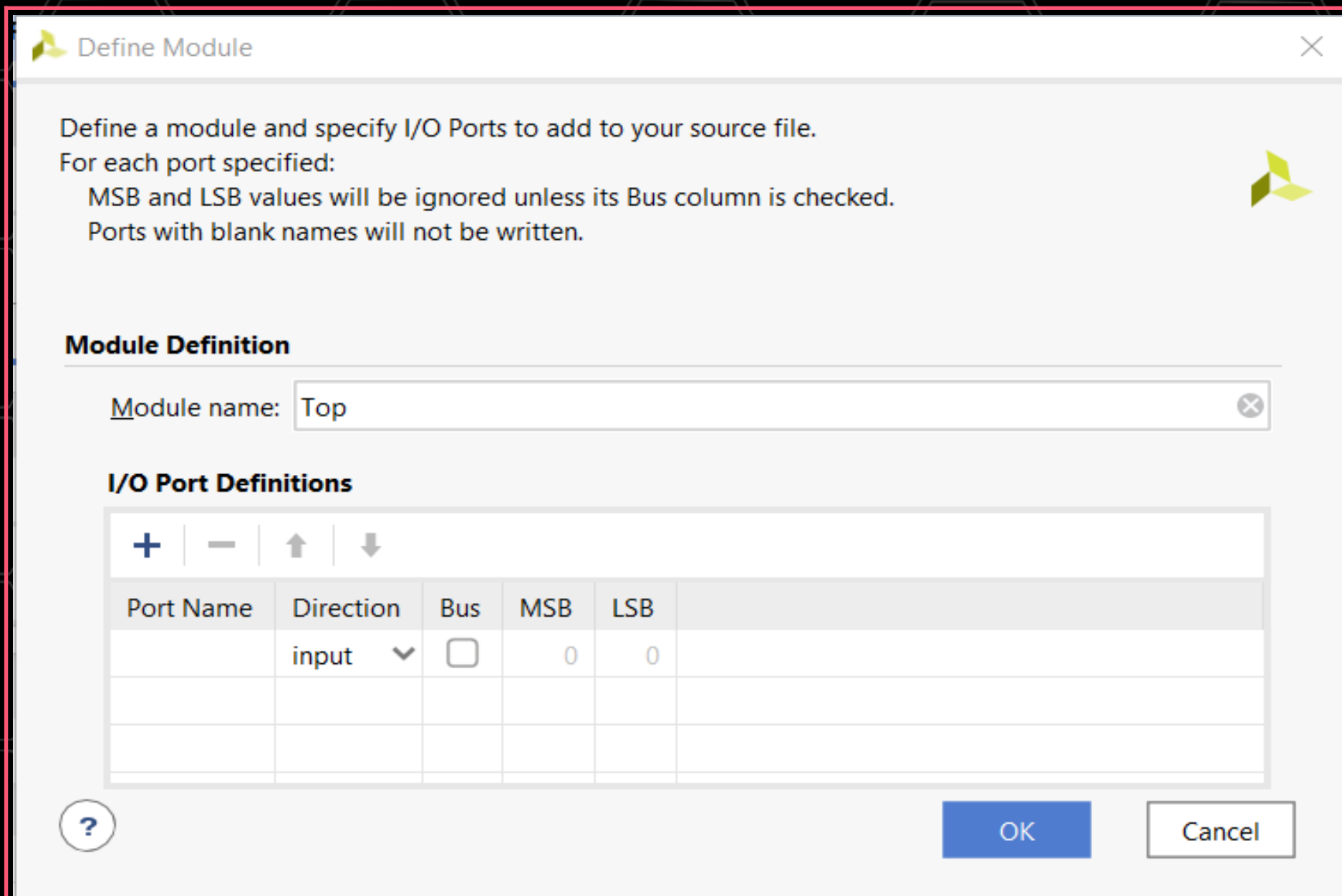
Next >

Finish

Cancel

16

Select "OK" at the next screen



Define a module and specify I/O Ports to add to your source file.
For each port specified:
MSB and LSB values will be ignored unless its Bus column is checked.
Ports with blank names will not be written.

Module Definition

Module name:

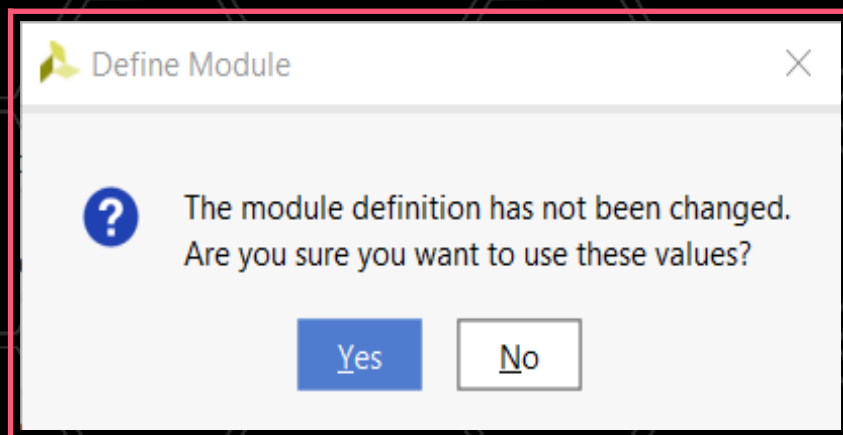
I/O Port Definitions

Port Name	Direction	Bus	MSB	LSB
	input	<input type="checkbox"/>	0	0

Buttons: ? OK Cancel

17

Select "Yes" at the next screen



18

Double click on "Top (Top.v)" in the Sources (design) panel to open the "Top.v" module. You can click on the square button which will maximize the Top.v module

The screenshot displays the Vivado 2019.2 IDE interface. The top menu bar includes File, Edit, Flow, Tools, Reports, Window, Layout, View, and Help. The toolbar contains various icons for file operations and design actions. The left sidebar shows the Flow Navigator with sections for PROJECT MANAGER, IP INTEGRATOR, SIMULATION, RTL ANALYSIS, and SYNTHESIS. The PROJECT MANAGER section is expanded, showing the Sources panel with a list of Design Sources (1) including Top (Top.v). The Source File Properties panel for Top.v is also visible, showing it is Enabled. The main editor area displays the Project Summary for Top.v, showing the file path C:/Users/adria/Desktop/BeelInvaders/Tutorials Basys 3/WIP/WIP.srcs/sources_1/new/Top.v. The editor shows the module definition for Top.v, with the line `module Top(` highlighted. The bottom status bar indicates the current design run is synth_1, with constraints_1, and the status is Not started.

WIP - [C:/Users/adria/Desktop/BeelInvaders/Tutorials Basys 3/WIP/WIP.xpr] - Vivado 2019.2

File Edit Flow Tools Reports Window Layout View Help Quick Access Ready

Flow Navigator

PROJECT MANAGER - WIP

PROJECT MANAGER

- Settings
- Add Sources
- Language Templates
- IP Catalog

IP INTEGRATOR

- Create Block Design
- Open Block Design
- Generate Block Design

SIMULATION

- Run Simulation

RTL ANALYSIS

- Open Elaborated Design

SYNTHESIS

Sources

- Design Sources (1)
 - Top (Top.v)

Hierarchy Libraries Compile Order

Source File Properties

- Top.v
- Enabled
- General Properties

Project Summary x Top.v x

C:/Users/adria/Desktop/BeelInvaders/Tutorials Basys 3/WIP/WIP.srcs/sources_1/new/Top.v

```
13 //
14 // Dependencies:
15 //
16 // Revision:
17 // Revision 0.01 - File Created
18 // Additional Comments:
19 //
20 //////////////////////////////////////////////////
21
22
23 module Top(
```

Tcl Console Messages Log Reports Design Runs x

Name	Constraints	Status	WNS	TNS	WHS	THS	TPWS	Total Power	Failed Routes	LUT	FF	BRAM	URAM	D
synth_1	constrs_1	Not started												
impl_1	constrs_1	Not started												

23:1 Insert Verilog

19

Remove all the code in the "Project Summary: Top.v" box and copy & paste the code from either the "Top.v" file you downloaded or from below, into the "Project Summary: Top.v" box

```
//-----  
// Top Module  
// Digilent Basys 3  
// BeeInvaders Tutorial 1 : Onboard clock 100MHz  
// VGA Resolution: 640x480 @ 60Hz  
// Pixel Clock 25MHz  
//-----  
`timescale 1ns / 1ps  
  
// Setup Top Module  
module Top(  
    input wire CLK, // Onboard clock 100MHz : INPUT Pin W5  
    input wire RESET, // Reset button : INPUT Pin U18  
    output wire HSYNC, // VGA horizontal sync : OUTPUT Pin P19  
    output wire VSYNC, // VGA vertical sync : OUTPUT Pin R19  
    output reg [3:0] RED, // 4-bit VGA Red : OUTPUT Pin G19, Pin H19, Pin J19, Pin N19  
    output reg [3:0] GREEN, // 4-bit VGA Green : OUTPUT Pin J17, Pin H17, Pin G17, Pin D17  
    output reg [3:0] BLUE, // 4-bit VGA Blue : OUTPUT Pin N18, Pin L18, Pin K18, Pin J18  
);  
  
    // Setup Reset button  
    wire rst = RESET; // reset : active high (BTNC)  
  
    // generate 25MHz pixel clock using a "Fractional Clock Divider"  
    reg [15:0] counter1;  
    reg pix_clk;  
    always @(posedge CLK)  
        // divide 100MHz by 4 = 25MHz : (2^16)/4 = 16384 decimal or 4000 hex  
        {pix_clk, counter1} <= counter1 + 16'h4000;  
  
    // instantiate vga640x480 code  
    wire [9:0] x; // pixel x position: 10-bit value: 0-1023 : only need 800  
    wire [9:0] y; // pixel y position: 10-bit value: 0-1023 : only need 525  
    wire active; // high during active pixel drawing  
    vga640x480 display (  
        .i_clk(CLK),  
        .i_pix_clk(pix_clk),  
        .i_rst(rst),  
        .o_hsync(HSYNC),  
        .o_vsync(VSYNC),  
        .o_x(x),  
        .o_y(y),  
        .o_active(active)  
    );  
  
    // setup palette and RGB registers  
    reg [7:0] palette [0:192]; // 8 bit values from the 192 hex entries in the colour palette  
    reg [7:0] COL; // holds hex colour palette value to display on the screen  
    reg [7:0] colourR; // 8 bit hex value for RED  
    reg [7:0] colourG; // 8 bit hex value for GREEN  
    reg [7:0] colourB; // 8 bit hex value for BLUE
```

```

// load colour palette
initial begin
    $readmemh("pal24bit.mem", palette); // load 192 hex values into "palette"
end

// fill the active area of the screen
always @ (posedge CLK)
begin
    COL <= 8'h19; // set colour to pink (decimal 25)
    if (active)
        begin
            colourR <= palette[(COL*3)]; // retrieve RED palette hex value
            colourG <= palette[(COL*3)+1]; // retrieve GREEN palette hex value
            colourB <= palette[(COL*3)+2]; // retrieve BLUE palette hex value
            RED <= colourR[7:4]; // output 4 left hand bits of the 8 bit RED value retrieved
            GREEN <= colourG[7:4]; // output 4 left hand bits of the 8 bit GREEN value retrieved
            BLUE <= colourB[7:4]; // output 4 left hand bits of the 8 bit BLUE value retrieved
        end
    else
        begin
            RED <= 0; // set RED, GREEN & BLUE
            GREEN <= 0; // to "0" when x,y outside of
            BLUE <= 0; // the active display area
        end
    end
end
endmodule

```

If you maximized the "Top.v" window you will now need to restore it by clicking on the "Restore" button

20

Repeat steps 11 to 18 as follows:

Step 11: Right click on "Design Sources" and left click on "Add Sources"

Step 12: Select "Add or create design sources" and click "Next"

Step 13: Select "+" and click on "Create File" or click on the "Create File" button

Step 14: Enter "vga640x480" in the box entitled "File name" and click "OK"

Step 15: Select "Finish"

Step 16: Select "OK"

Step 17: Select "Yes"

Step 18: Double click "vga640x480 (vga640x480.v)" in the Sources (design) panel to open the module (maximize the window if it helps to see it)

Remove all the code in "Project Summary: vga640x480.v" box (maximize it if it helps) and copy & paste the code from either the "vga640x480.v" file you downloaded or from below, into the "Project Summary: vga640x480.v" box

```
//-----  
// vga640x480 Module  
// Digilent Basys 3  
// BeeInvaders : Onboard clock 100MHz  
// VGA Resolution: 640x480 @ 60Hz  
// Pixel Clock 25MHz  
//-----  
`timescale 1ns / 1ps  
  
// Setup vga640x480 Module  
module vga640x480(  
    input wire i_clk, // 100MHz onboard clock  
    input wire i_pix_clk, // 25MHz pixel clock  
    input wire i_rst, // reset  
    output wire o_hsync, // horizontal sync  
    output wire o_vsync, // vertical sync  
    output wire o_active, // high during active pixel drawing  
    output wire [9:0] o_x, // current pixel x position  
    output wire [9:0] o_y // current pixel y position
```

```

);

// setup VGA timings
//-----
// VGA 640x480 Horizontal Timing (line)
localparam HSYNCSTART = 16; // horizontal sync start
localparam HSYNCEND = 16 + 96; // horizontal sync end
localparam HACTIVESTART = 16 + 96 + 48; // horizontal active start
localparam HACTIVEEND = 16 + 96 + 48 + 640; // total line length in pixels
reg [9:0] H_SCAN; // line position

// VGA 640x480 Vertical timing (frame)
localparam VSYNCSTART = 10; // vertical sync start
localparam VSYNCEND = 10 + 2; // vertical sync end
localparam VACTIVESTART = 10 + 2 + 33; // vertical active start
localparam VACTIVEEND = 10 + 2 + 33 + 480; // vertical active end
reg [9:0] V_SCAN; // screen position

// set sync signals to low (active) or high (inactive)
assign o_hsync = ~((H_SCAN >= HSYNCSTART) & (H_SCAN < HSYNCEND));
assign o_vsync = ~((V_SCAN >= VSYNCSTART) & (V_SCAN < VSYNCEND));

// set x and y values
assign o_x = (H_SCAN < HACTIVESTART) ? 0 : (H_SCAN - HACTIVESTART);
assign o_y = (V_SCAN < VACTIVESTART) ? 0 : (V_SCAN - VACTIVESTART);

// set active high during active area
assign o_active = ~((H_SCAN < HACTIVESTART) | (V_SCAN < VACTIVESTART));

// check for reset / create frame loop
always @ (posedge i_clk)
begin
    // check for reset button pressed
    if (i_rst) // jump to start of a frame and reset registers
    begin
        H_SCAN <= 0;
        V_SCAN <= 0;
    end

    // loop through a full screen
    if (i_pix_clk)
    begin
        if (H_SCAN == HACTIVEEND) // if at the end of a line update registers
        begin
            H_SCAN <= 0;
            V_SCAN <= V_SCAN + 1;
        end
        else
            H_SCAN <= H_SCAN + 1; // else increment horizontal counter

        if (V_SCAN == VACTIVEEND) // if at the end of a screen reset vertical counter
            V_SCAN <= 0;
    end
end
endmodule

```

If you maximized the "vga640x480.v" window you will now need to restore it by clicking on the "Restore" button

21

Repeat steps 11 to 15 as follows:

Step 11: Right click on "Design Sources" and left click on "Add Sources"

Step 12: Select "Add or create constraints" and click "Next"

Step 13: Select "+" and click on "Create File" or click on the "Create File" button

Step 14: Enter "basys3.xdc" in the box entitled "File name:" ("File type:" will say "XDC") and click "OK"

Step 15: Select "Finish"

Double click on "basys3.xdc" in the Sources (constraints) panel to open the module

Copy & paste the code from either the "basys3.xdc" file you downloaded or from below, into the "Project Summary: basys3.xdc" box

```
##-----  
## Constraints Module  
## Digilent Basys 3  
## BeeInvaders : Onboard clock 100MHz  
## VGA Resolution: 640x480 @ 60Hz  
## Pixel Clock 25MHz  
##-----  
  
## Clock  
set_property -dict {PACKAGE_PIN W5 IOSTANDARD LVCMOS33} [get_ports {CLK}];  
create_clock -add -name sys_clk_pin -period 10.00 \  
-waveform {0 5} [get_ports {CLK}];  
  
## Use BTNC as Reset Button (active high)  
set_property -dict {PACKAGE_PIN U18 IOSTANDARD LVCMOS33} [get_ports {RESET}];  
  
## VGA Connector  
set_property -dict {PACKAGE_PIN G19 IOSTANDARD LVCMOS33} [get_ports {RED[0]}};  
set_property -dict {PACKAGE_PIN H19 IOSTANDARD LVCMOS33} [get_ports {RED[1]}};  
set_property -dict {PACKAGE_PIN J19 IOSTANDARD LVCMOS33} [get_ports {RED[2]}};  
set_property -dict {PACKAGE_PIN N19 IOSTANDARD LVCMOS33} [get_ports {RED[3]}};  
set_property -dict {PACKAGE_PIN N18 IOSTANDARD LVCMOS33} [get_ports {BLUE[0]}};  
set_property -dict {PACKAGE_PIN L18 IOSTANDARD LVCMOS33} [get_ports {BLUE[1]}};  
set_property -dict {PACKAGE_PIN K18 IOSTANDARD LVCMOS33} [get_ports {BLUE[2]}};  
set_property -dict {PACKAGE_PIN J18 IOSTANDARD LVCMOS33} [get_ports {BLUE[3]}};  
set_property -dict {PACKAGE_PIN J17 IOSTANDARD LVCMOS33} [get_ports {GREEN[0]}};  
set_property -dict {PACKAGE_PIN H17 IOSTANDARD LVCMOS33} [get_ports {GREEN[1]}};
```

```
set_property -dict {PACKAGE_PIN G17 IOSTANDARD LVCMOS33} [get_ports {GREEN[2]}};  
set_property -dict {PACKAGE_PIN D17 IOSTANDARD LVCMOS33} [get_ports {GREEN[3]}};  
set_property -dict {PACKAGE_PIN P19 IOSTANDARD LVCMOS33} [get_ports {HSYNC}};  
set_property -dict {PACKAGE_PIN R19 IOSTANDARD LVCMOS33} [get_ports {VSYNC}};
```

```
## Configuration options, can be used for all designs  
set_property CONFIG_VOLTAGE 3.3 [current_design]  
set_property CFGBVS VCCO [current_design]
```

22

The code loads a file called "Pal24bit.mem". This is a colour palette which will include the background colour we are going to use. How this data was created will be explained in another tutorial



Either copy and paste the following data into a "Notepad" file.

```
00 01 00 21 0F 20 3C 0B 40 25 1C 0D 16 26 12 45 1D 0A 39 25 36 32 2C 01 CA 00 06 28 30 31 26 3D 01 43 4A 48 51 4F 00 8C 34 93
B1 26 B7 34 5F 36 7E 4B 2A 70 4A 71 31 69 04 8E 4B 19 6B 53 3D 52 5B 44 F1 21 F0 72 6B 07 62 69 67 F2 52 87 E1 5E 3F CD 58 D1
7D 89 04 51 8B 8E 3C 9B 29 85 8B 85 4E A9 01 CC 7D 3D B9 8D 07 9A 9C 00 F0 80 1E A2 96 8C 84 A7 67 BE 99 6A C1 9D 43 AD 9C AD
83 AD 8A 1A E2 2A A7 AD A9 B2 B9 00 E4 A9 0D 6C D1 00 FA B8 10 FD BE 32 AF DC 00 FD C3 4B C7 CC C9 F9 CF 00 D3 CA CE 7B EF 8D
DF DA 2B 7F EA F4 E0 E4 00 E6 E6 85 E1 E7 E5 FC FC 00 F4 EE F1 F6 F9 F6
```

Save this in the "WIP\WIP.srcs\sources_1\new" folder and call it "pal24bit.mem"

Or copy and paste the downloaded file "pal24bit.mem" into the "WIP\WIP.srcs\sources_1\new" folder

23

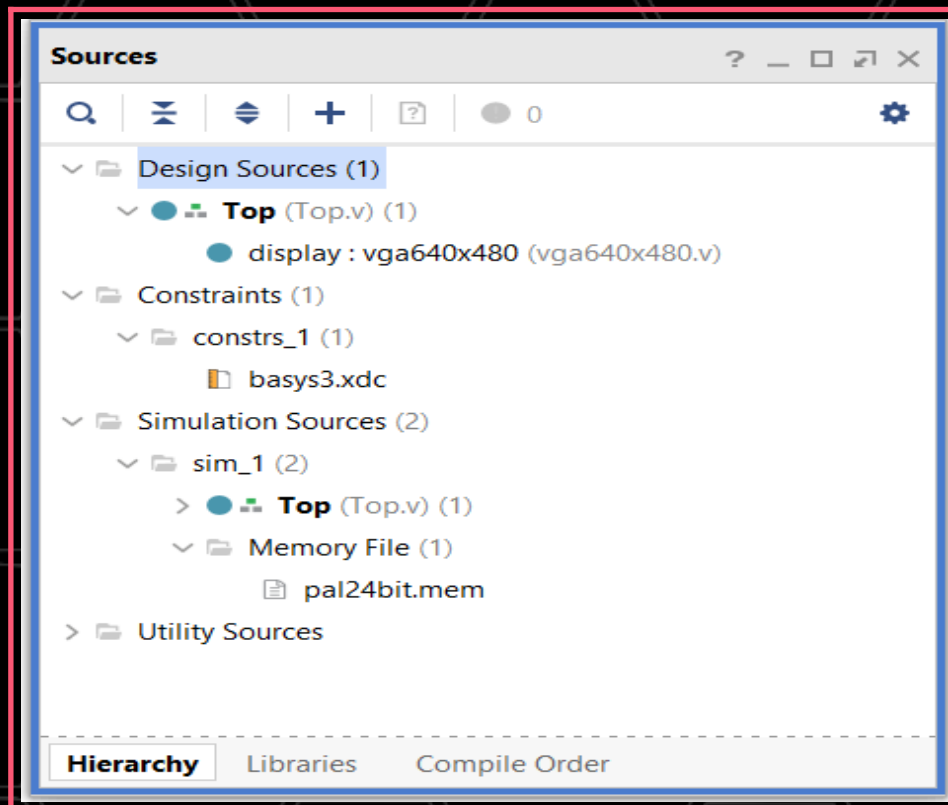
Right click on "Design Sources" and left click to select "Add Sources" (step 11)

Select "Add or create design sources" and click "Next" (step 12)

Select "Add File".

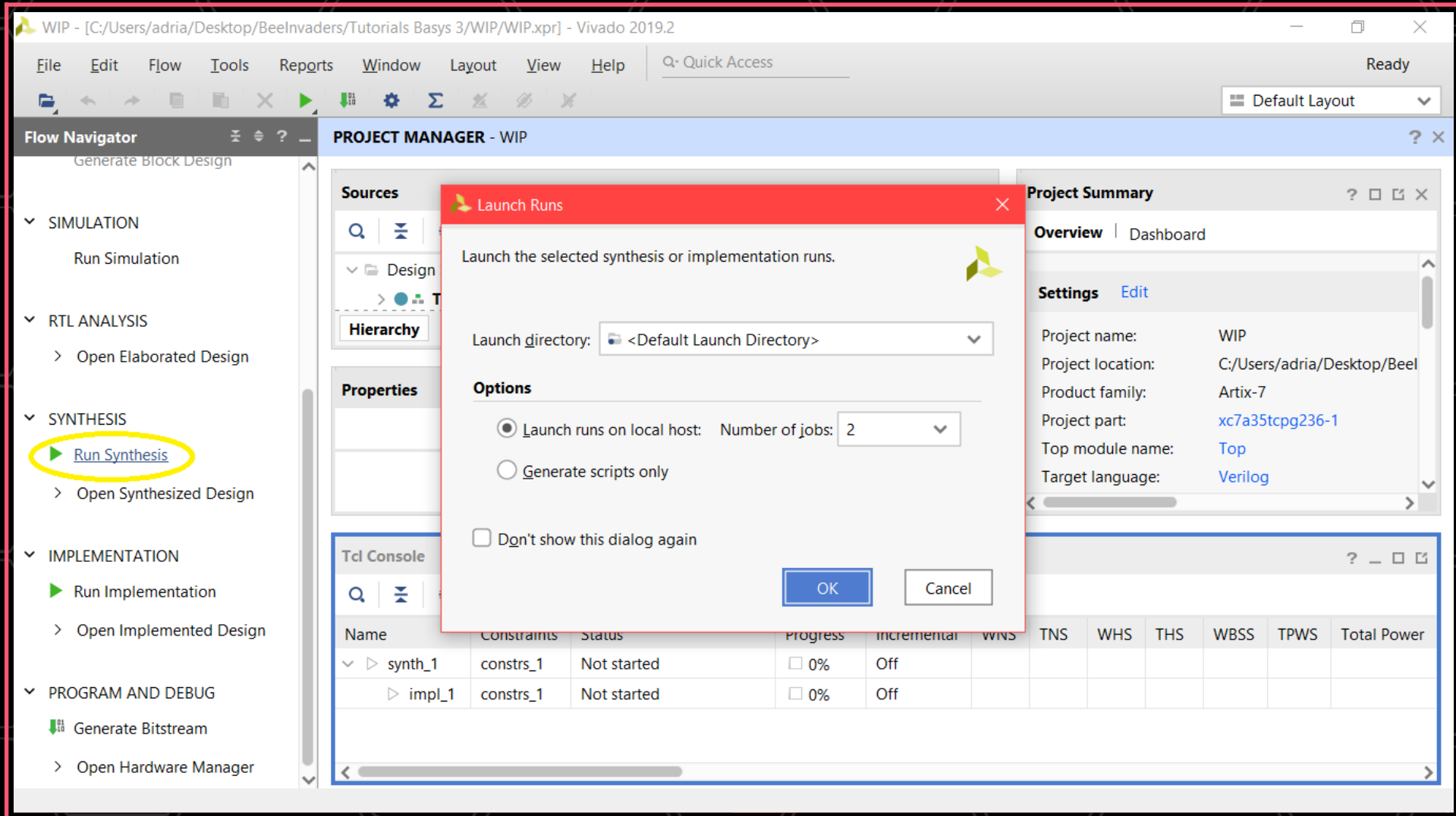
Ensure that the path is set to "WIP\WIP.srcs\sources_1\new" and select "pal24bit.mem" and "OK"

Select "Finish" and this will put the file under "Simulation Sources: Memory File"



24

Click on "Run Synthesis" (if a window pops up asking if you would like to save the project click "Save") and when the "Launch Runs" window appears click "OK"



25

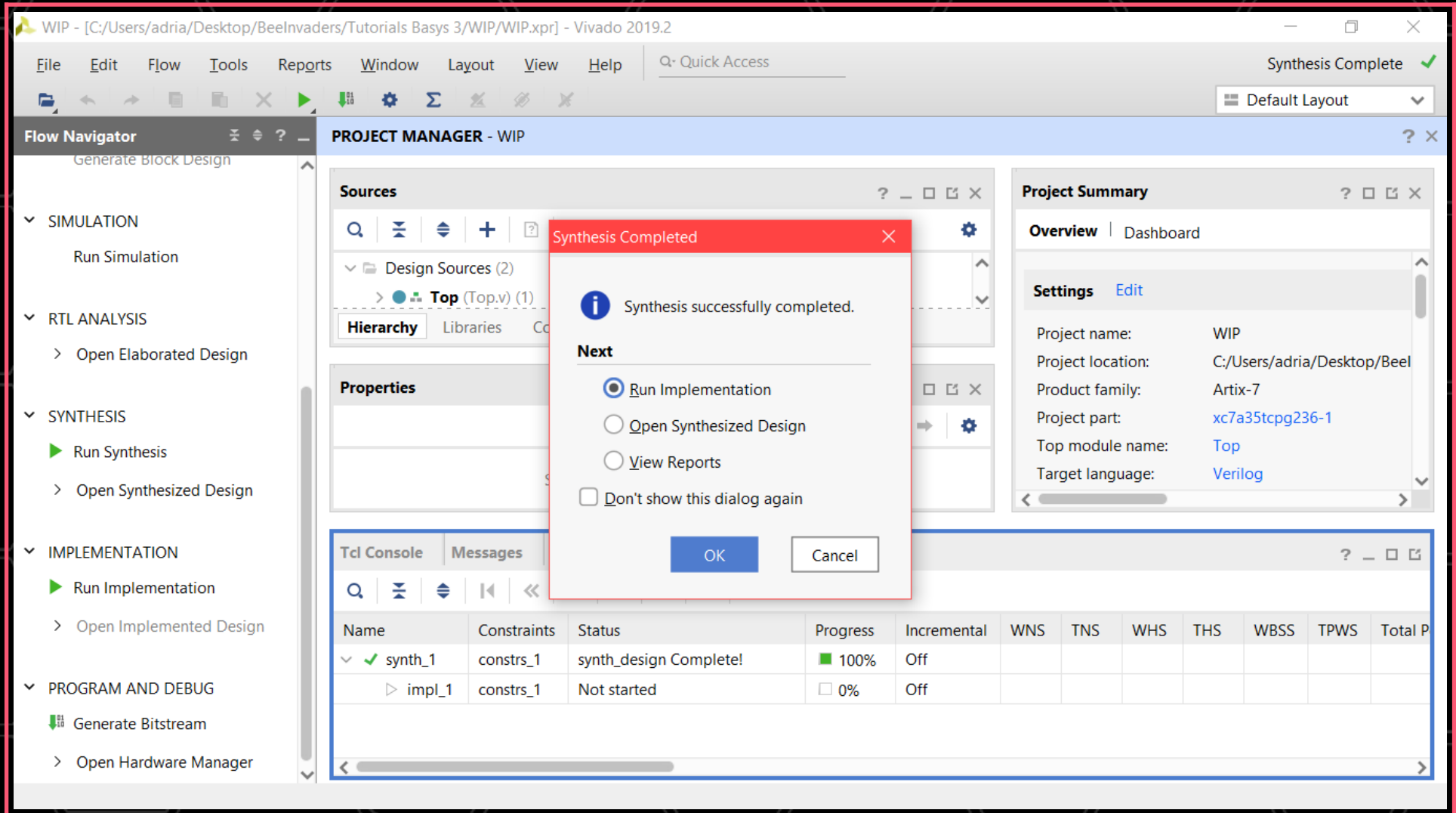
At this point you will see "Running Synth_design: Cancel"

The screenshot shows the Vivado 2019.2 IDE interface. The title bar indicates the project is 'WIP' located at 'C:/Users/adria/Desktop/BeelInvaders/Tutorials Basys 3/WIP/WIP.xpr'. The menu bar includes File, Edit, Flow, Tools, Reports, Window, Layout, View, and Help. A toolbar with various icons is visible below the menu bar. The 'Flow Navigator' on the left lists project steps: SIMULATION, RTL ANALYSIS, SYNTHESIS (highlighted), and IMPLEMENTATION. Under SYNTHESIS, 'Run Synthesis' is selected. The 'PROJECT MANAGER - WIP' pane shows the 'Sources' tab with 'Design Sources (2)' and 'Top (Top.v) (1)'. The 'Properties' pane is empty, prompting to 'Select an object to see properties'. The 'Project Summary' pane on the right shows project details: Project name: WIP, Project location: C:/Users/adria/Desktop/Beel, Product family: Artix-7, Project part: xc7a35tcpg236-1, Top module name: Top, and Target language: Verilog. The 'Design Runs' tab at the bottom shows a table of design runs.

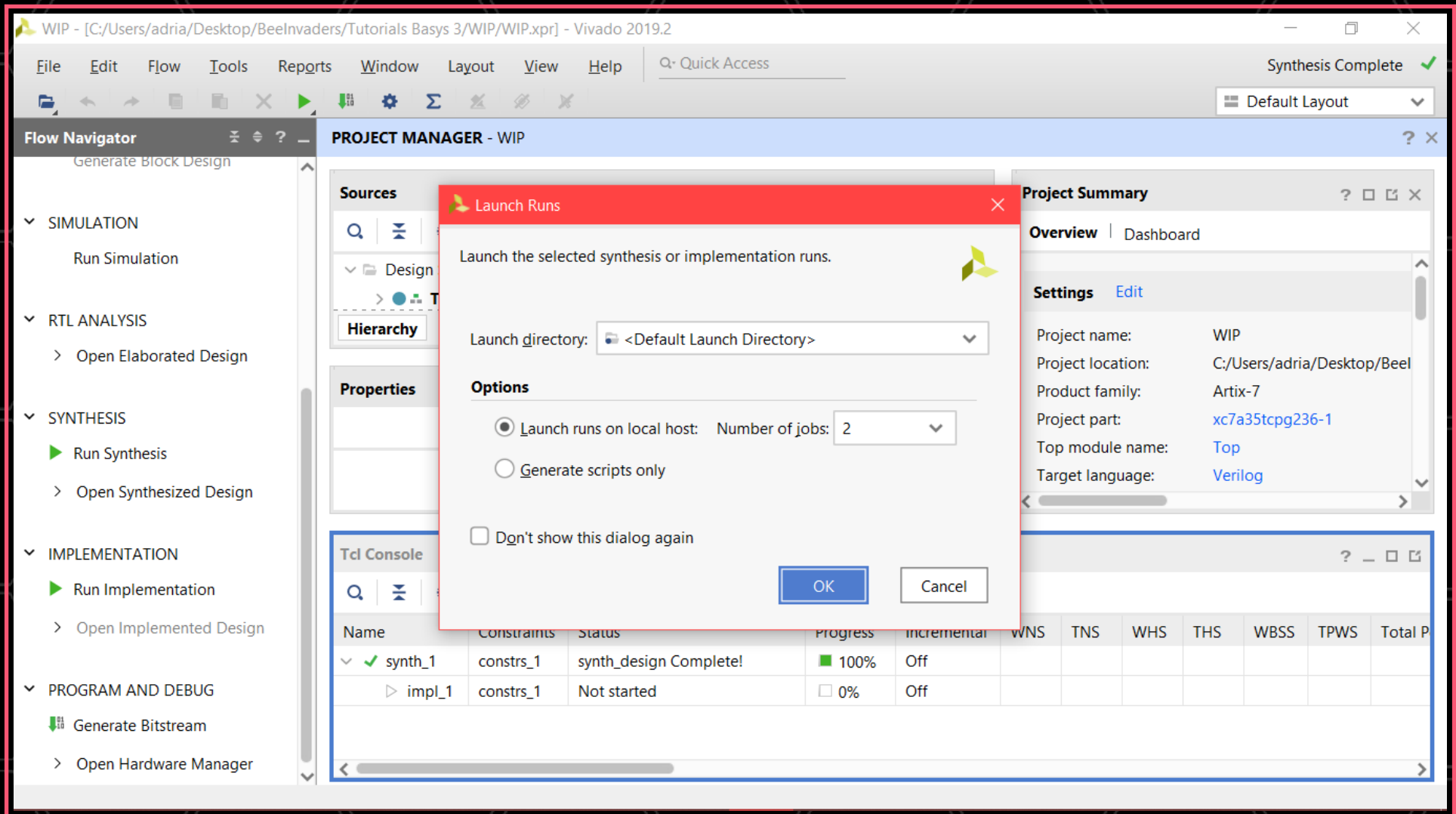
Name	Constraints	Status	Progress	Incremental	WNS	TNS	WHS	THS	WBSS	TPWS	Total Power
synth_1	constrs_1	Running synth_design...	0%	Off							
impl_1	constrs_1	Not started	0%	Off							

26

When the window "Synthesis Completed" appears ensure "Run implementation" is selected and click "OK"



27 When the "Launch Runs" window appears click "OK"



28

When the "Implementation Completed" window appears select "Generate Bitstream" and click "OK"

WIP - [C:/Users/adria/Desktop/BeelInvaders/Tutorials Basys 3/WIP/WIP.xpr] - Vivado 2019.2

File Edit Flow Tools Reports Window Layout View Help Quick Access

Implementation Complete ✓

Default Layout

Flow Navigator

- Generate Block Design
- SIMULATION
 - Run Simulation
- RTL ANALYSIS
 - Open Elaborated Design
- SYNTHESIS
 - Run Synthesis
 - Open Synthesized Design
- IMPLEMENTATION
 - Run Implementation
 - Open Implemented Design
- PROGRAM AND DEBUG
 - Generate Bitstream
 - Open Hardware Manager

PROJECT MANAGER - WIP

Sources

- Design Sources (2)
- Top (Top.v)

Hierarchy Libraries

Properties

Implementation Completed

Implementation successfully completed.

Next

- ☐ Open Implemented Design
- ☒ Generate Bitstream
- ☐ View Reports
- ☐ Don't show this dialog again

OK Cancel

Project Summary

Overview | Dashboard

Settings Edit

Project name: WIP

Project location: C:/Users/adria/Desktop/Beel

Product family: Artix-7

Project part: xc7a35tcpg236-1

Top module name: Top

Target language: Verilog

Tcl Console Messages

Name	Constraints	Status	Progress	Incremental	WNS	TNS	WHS	THS	WBSS	TPWS	Total Po
✓ synth_1	constrs_1	synth_design Complete!	100%	Off							
✓ impl_1	constrs_1	route_design Complete!	100%	Off	4.545	0.000	0.212	0.000		0.000	

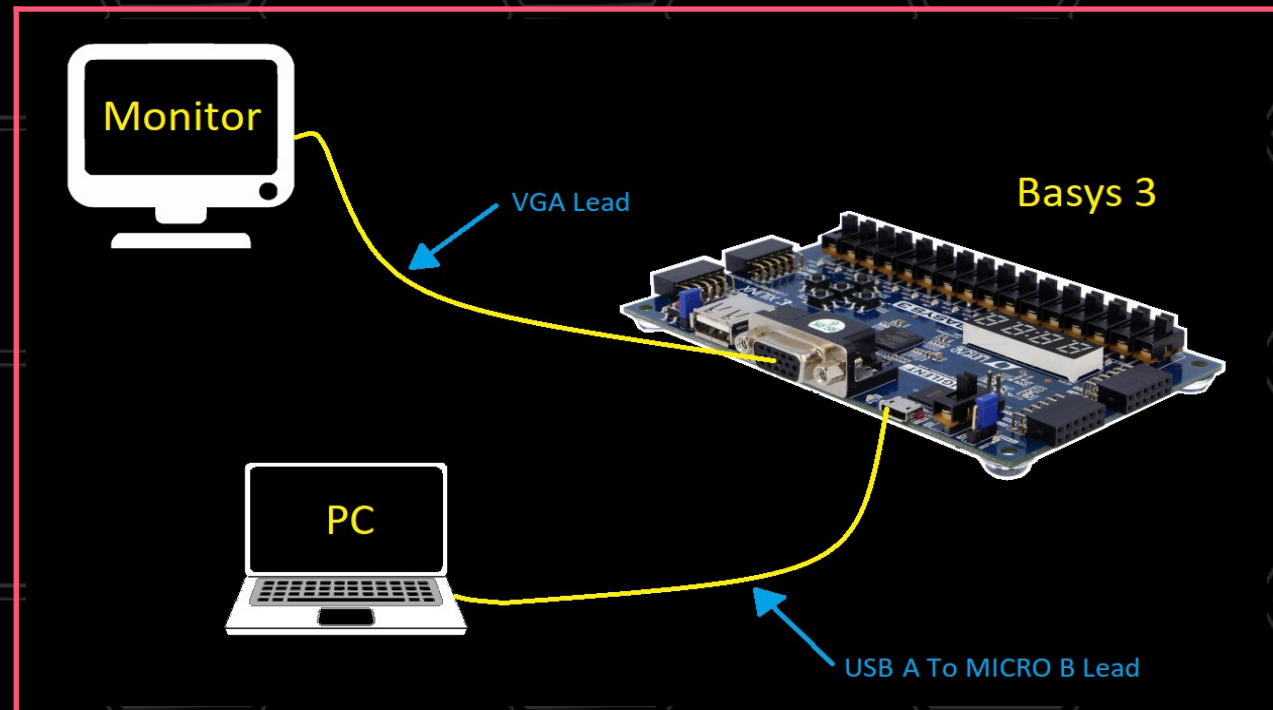
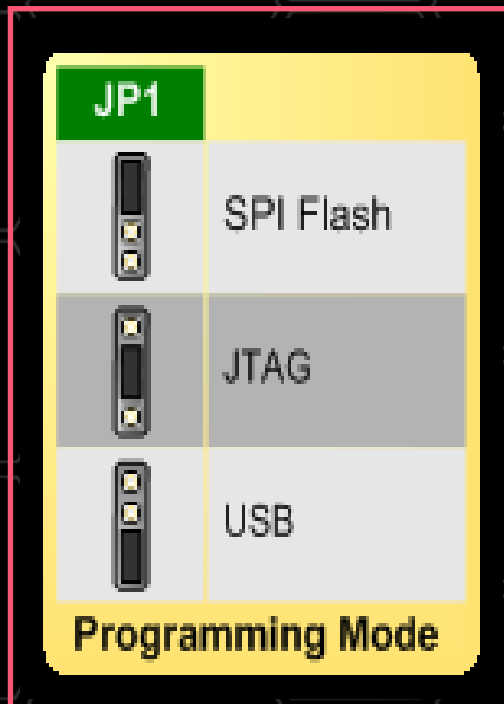
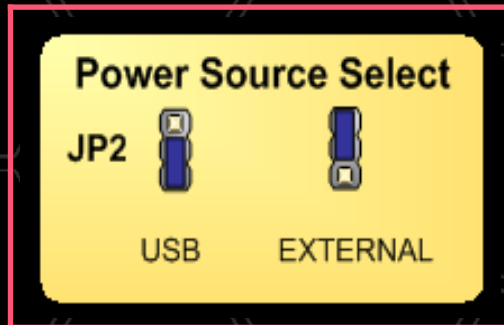
29 When the "Launch Runs" window appears click "OK"

The screenshot shows the Vivado 2019.2 IDE interface. The 'Flow Navigator' on the left lists the project steps: Generate Block Design, SIMULATION, RTL ANALYSIS, SYNTHESIS, IMPLEMENTATION, and PROGRAM AND DEBUG. The 'PROJECT MANAGER - WIP' window is open, showing the 'Sources' and 'Hierarchy' tabs. A 'Launch Runs' dialog box is displayed in the center, prompting the user to launch the selected synthesis or implementation runs. The dialog includes a 'Launch directory' dropdown set to '<Default Launch Directory>', an 'Options' section with radio buttons for 'Launch runs on local host' (selected) and 'Generate scripts only', and a 'Number of jobs' dropdown set to '2'. There are 'OK' and 'Cancel' buttons at the bottom. The 'Project Summary' window on the right shows project details: Project name: WIP, Project location: C:/Users/adria/Desktop/BeelInvaders, Product family: Artix-7, Project part: xc7a35tcbg236-1, Top module name: Top, and Target language: Verilog. The 'Tcl Console' at the bottom shows a table of implementation results.

Name	Constraints	Status	Progress	Incremental	WNS	TNS	WHS	THS	WBSS	TPWS	Total Po
✓ synth_1	constrs_1	synth_design Complete!	100%	Off							
✓ impl_1	constrs_1	route_design Complete!	100%	Off	4.545	0.000	0.212	0.000		0.000	

30

When the "Bitstream Generation Completed" window appears you will need to make sure that the Basys 3 board jumper JP2 is set to USB and JP1 is in the JTAG position, connect the Basys 3 board to your computer and your VGA screen (as shown below) and switch the board on



Now select "Open Hardware Manager" and click "OK"

WIP - [C:/Users/adria/Desktop/BeelInvaders/Tutorials Basys 3/WIP/WIP.xpr] - Vivado 2019.2

File Edit Flow Tools Reports Window Layout View Help Q Quick Access write_bitstream Complete ✓ Default Layout

Flow Navigator Generate Block Design

- SIMULATION
 - Run Simulation
- RTL ANALYSIS
 - Open Elaborated Design
- SYNTHESIS
 - Run Synthesis
 - Open Synthesized Design
- IMPLEMENTATION
 - Run Implementation
 - Open Implemented Design
- PROGRAM AND DEBUG
 - Generate Bitstream
 - Open Hardware Manager

PROJECT MANAGER - WIP

Sources

- Design Sources (2)
- Top (Top.v)

Hierarchy Libraries

Properties

Bitstream Generation Completed

Bitstream Generation successfully completed.

Next

- ☐ Open Implemented Design
- ☐ View Reports
- ☒ Open Hardware Manager
- ☐ Generate Memory Configuration File
- ☐ Don't show this dialog again

OK Cancel

Tcl Console Message

Name	Constraints	Status	Progress	Incremental	WNS	TNS	WHS	THS	WBSS	TPWS	Total Po
✓ synth_1	constrs_1	synth_design Complete!	100%	Off							
✓ impl_1	constrs_1	write_bitstream Complete!	100%	Off	4.545	0.000	0.212	0.000		0.000	

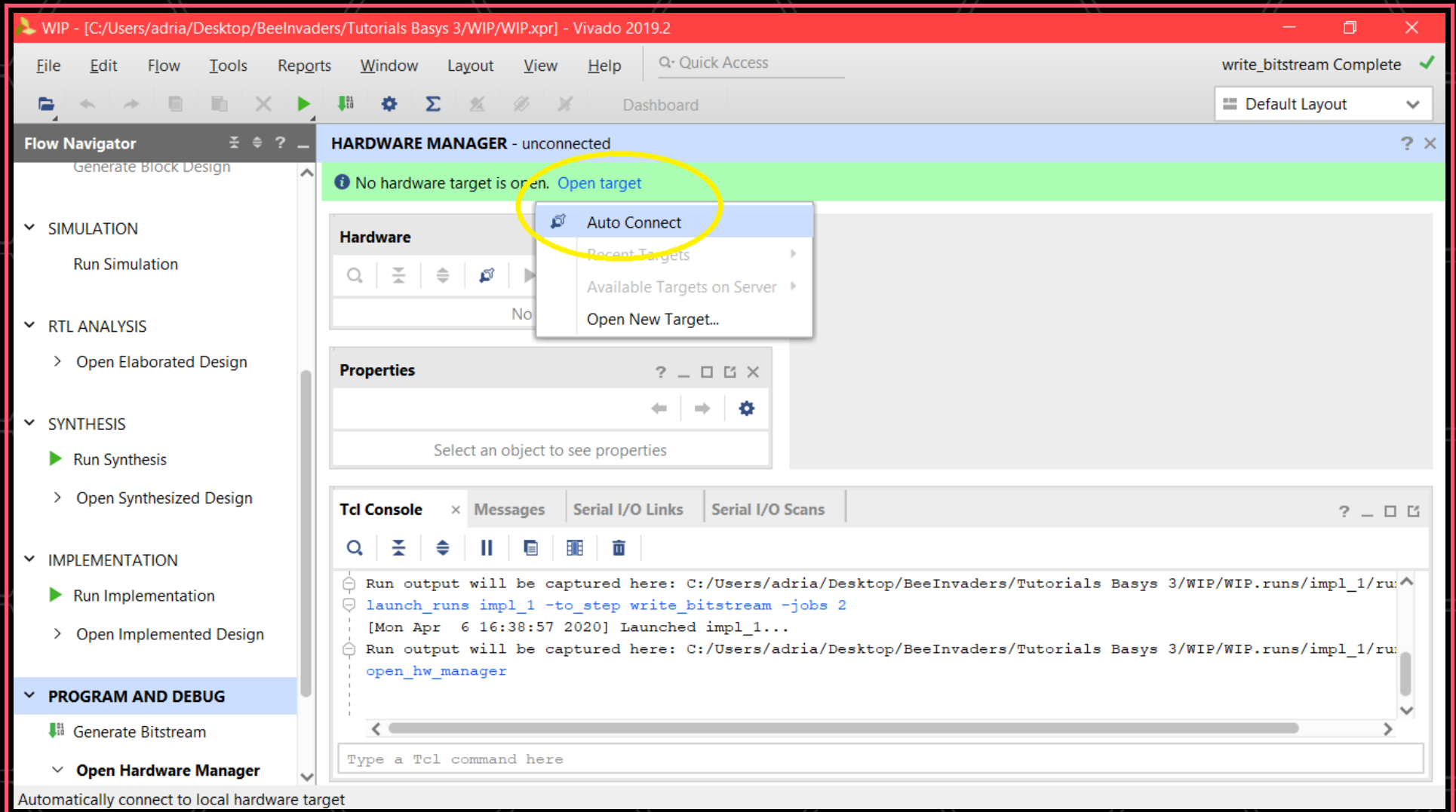
Project Summary Overview | Dashboard

Settings Edit

Project name: WIP
Project location: C:/Users/adria/Desktop/Beel
Product family: Artix-7
Project part: xc7a35tcbg236-1
Top module name: Top
Target language: Verilog

31

Next click "Open Target" and select "Auto Connect"



32 Now click "Program Device"

The screenshot shows the Vivado 2019.2 interface with the Hardware Manager window open. The title bar indicates the project is 'WIP' located at '[C:/Users/adria/Desktop/BeeInvaders/Tutorials Basys 3/WIP/WIP.xpr]'. The top menu bar includes File, Edit, Flow, Tools, Reports, Window, Layout, View, and Help. A 'Quick Access' search bar is present. The 'Flow Navigator' on the left lists various design stages: Generate Block Design, SIMULATION (Run Simulation), RTL ANALYSIS (Open Elaborated Design), SYNTHESIS (Run Synthesis, Open Synthesized Design), IMPLEMENTATION (Run Implementation, Open Implemented Design), and PROGRAM AND DEBUG (Generate Bitstream, Open Hardware Manager). The 'PROGRAM AND DEBUG' section is currently selected. The main 'HARDWARE MANAGER' window shows the target 'localhost/xilinx_tcf/Digilent/210183AA6110A'. A green status bar at the top of the hardware manager contains the message 'There are no debug cores' followed by a yellow circle around the 'Program device' button and a 'Refresh device' button. Below this, there are sections for 'Hardware' and 'Properties', both currently empty. At the bottom, the 'Tcl Console' is open, displaying a series of commands and their outputs: 'INFO: [Labtoolstcl 44-466] Opening hw_target localhost:3121/xilinx_tcf/Digilent/210183AA6110A', 'open_hw_target: Time (s): cpu = 00:00:08 ; elapsed = 00:00:10 . Memory (MB): peak = 1980.355 ; gain = 1237.47', 'set_property PROGRAM.FILE {C:/Users/adria/Desktop/BeeInvaders/Tutorials Basys 3/WIP/WIP.runs/impl_1/Top.bit}', 'current_hw_device [get_hw_devices xc7a35t_0]', 'refresh_hw_device -update_hw_probes false [lindex [get_hw_devices xc7a35t_0] 0]', and 'INFO: [Labtools 27-1435] Device xc7a35t (JTAG device index = 0) is not programmed (DONE status = 0)'. A text input field at the bottom of the console prompts 'Type a Tcl command here'.

WIP - [C:/Users/adria/Desktop/BeeInvaders/Tutorials Basys 3/WIP/WIP.xpr] - Vivado 2019.2

File Edit Flow Tools Reports Window Layout View Help Quick Access

write_bitstream Complete ✓

Default Layout

Flow Navigator

- Generate Block Design
- SIMULATION
 - Run Simulation
- RTL ANALYSIS
 - Open Elaborated Design
- SYNTHESIS
 - Run Synthesis
 - Open Synthesized Design
- IMPLEMENTATION
 - Run Implementation
 - Open Implemented Design
- PROGRAM AND DEBUG**
 - Generate Bitstream
 - Open Hardware Manager

HARDWARE MANAGER - localhost/xilinx_tcf/Digilent/210183AA6110A

There are no debug cores **Program device** Refresh device

Hardware

Name Status

Properties

Select an object to see properties

Tcl Console

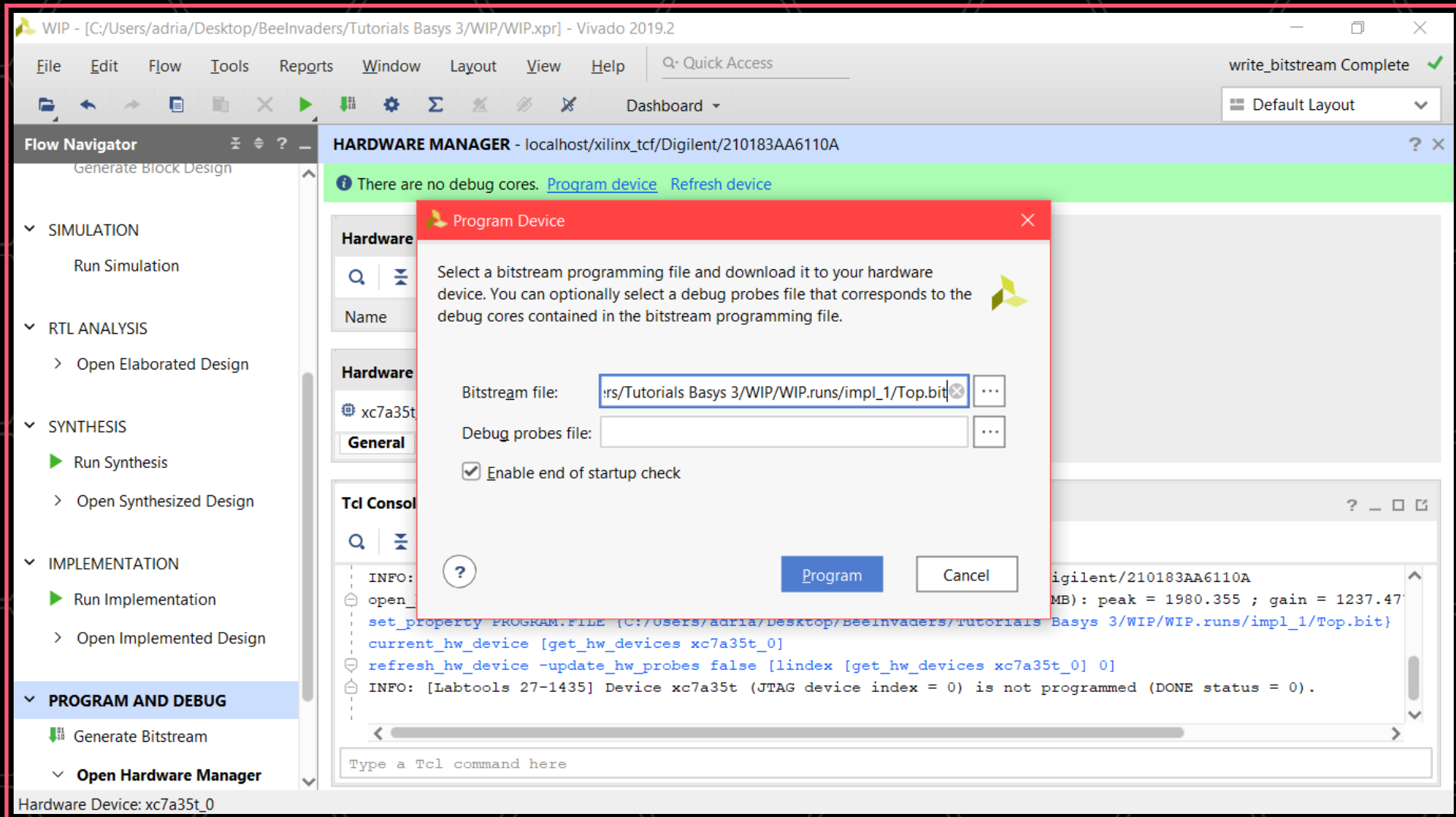
Messages Serial I/O Links Serial I/O Scans

```
INFO: [Labtoolstcl 44-466] Opening hw_target localhost:3121/xilinx_tcf/Digilent/210183AA6110A
open_hw_target: Time (s): cpu = 00:00:08 ; elapsed = 00:00:10 . Memory (MB): peak = 1980.355 ; gain = 1237.47
set_property PROGRAM.FILE {C:/Users/adria/Desktop/BeeInvaders/Tutorials Basys 3/WIP/WIP.runs/impl_1/Top.bit}
current_hw_device [get_hw_devices xc7a35t_0]
refresh_hw_device -update_hw_probes false [lindex [get_hw_devices xc7a35t_0] 0]
INFO: [Labtools 27-1435] Device xc7a35t (JTAG device index = 0) is not programmed (DONE status = 0).
```

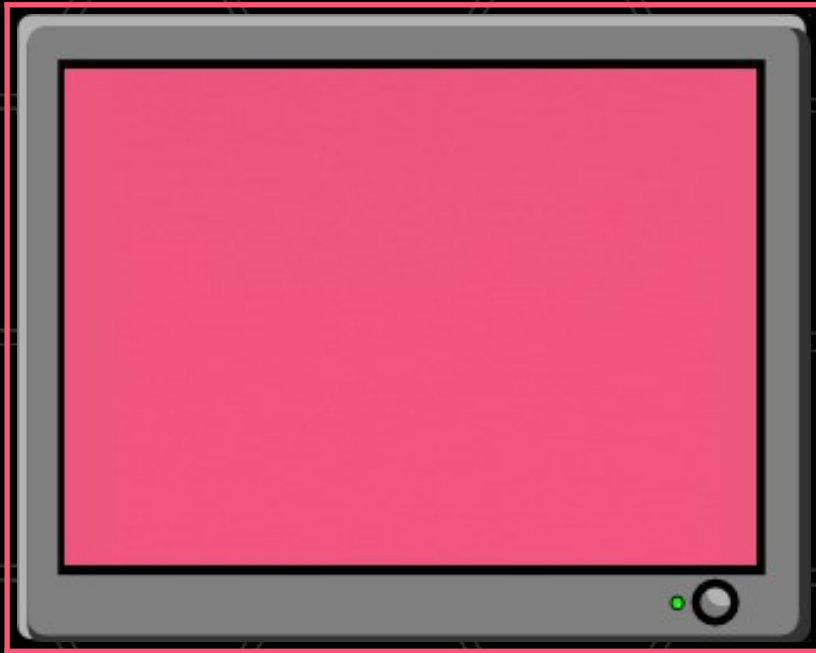
Type a Tcl command here

33

When the "Program Device" box appears click "Program"



34 You should see a pink coloured screen on your VGA monitor



Lastly, change the screen fill colour to black by changing the line of code in the "Top.v" module which states `COL <= 8'h19;` to read `COL <= 8'h0;` Then Run Synthesis etc. / program the board to test it

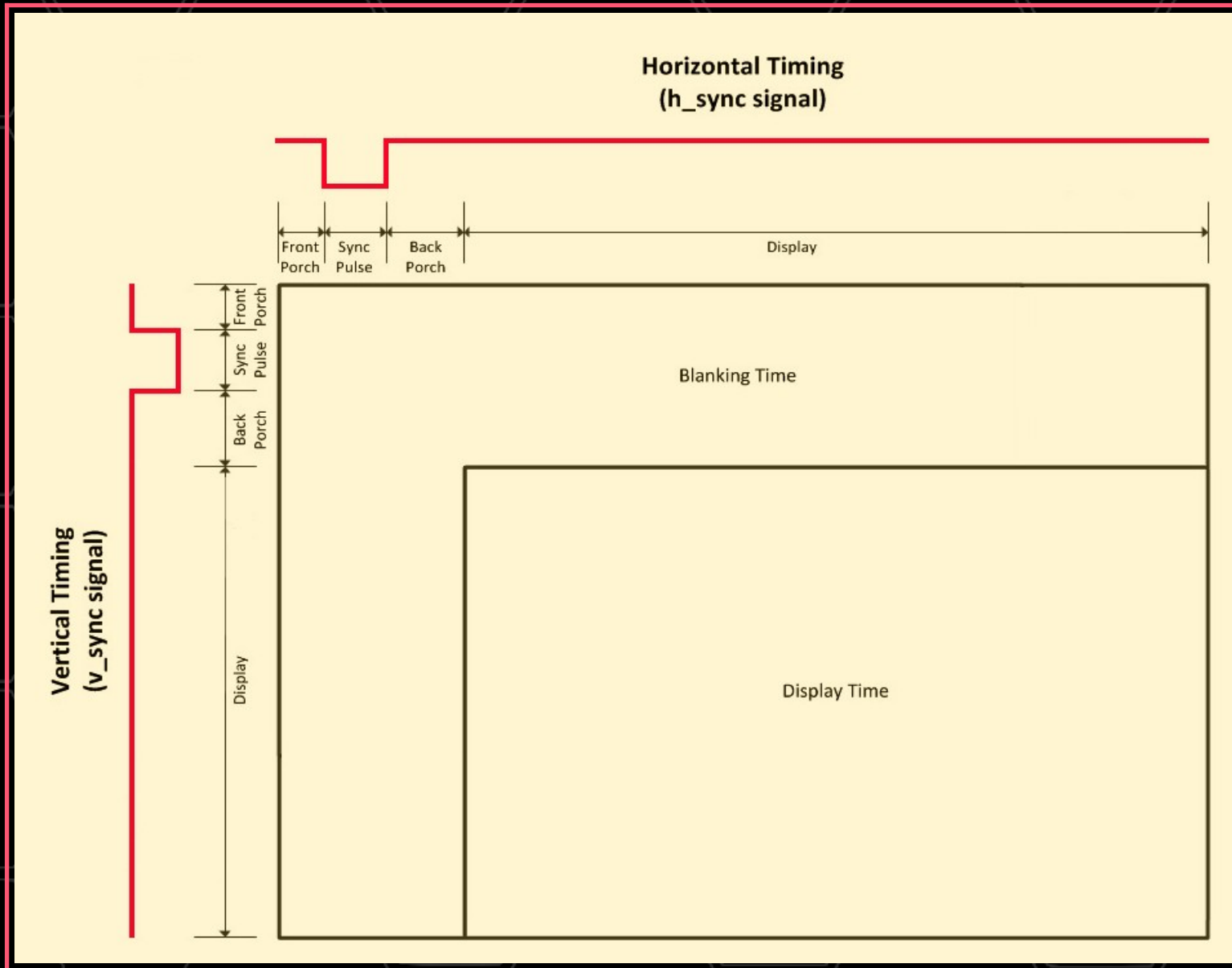
The VGA Signal Explained

VGA is an analogue video standard using a 15-pin D-sub connector. It doesn't require high clock speeds or complex encoding, so is an excellent place to begin when learning about FPGA graphics

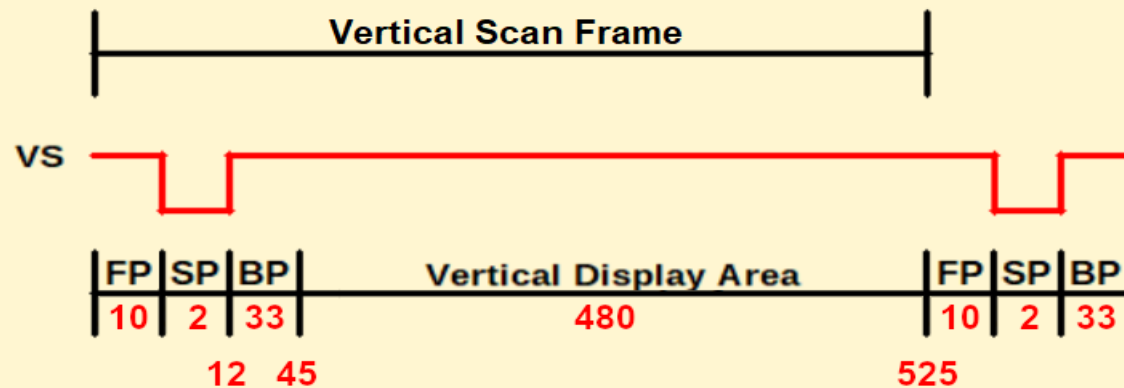
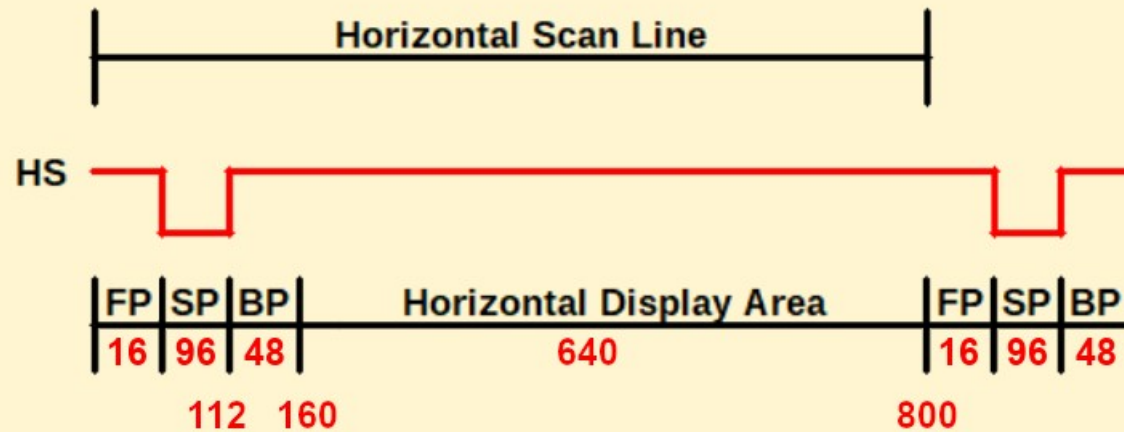
VGA has five main signal pins: one for each of red, green, and blue signals and two for sync. Horizontal sync separates a line and Vertical sync separates a screen, also known as a frame

VGA signals have two phases: drawing pixels and the blanking interval. The sync signals occur within blanking intervals; separated from pixel drawing by the front porch and back porch. When VGA was developed monitors were based on cathode ray tubes (CRTs): the blanking interval gave time for voltage levels to stabilise and for the electron gun to return to the start of a line or screen

The below diagram shows the horizontal and vertical timings for a VGA signal



Recommended Sync Pulse (SP), Back Porch (BP), Display Area and Front Porch (FP) for the vertical / horizontal timings for a 640 x 480 @ 60 Hz VGA video mode



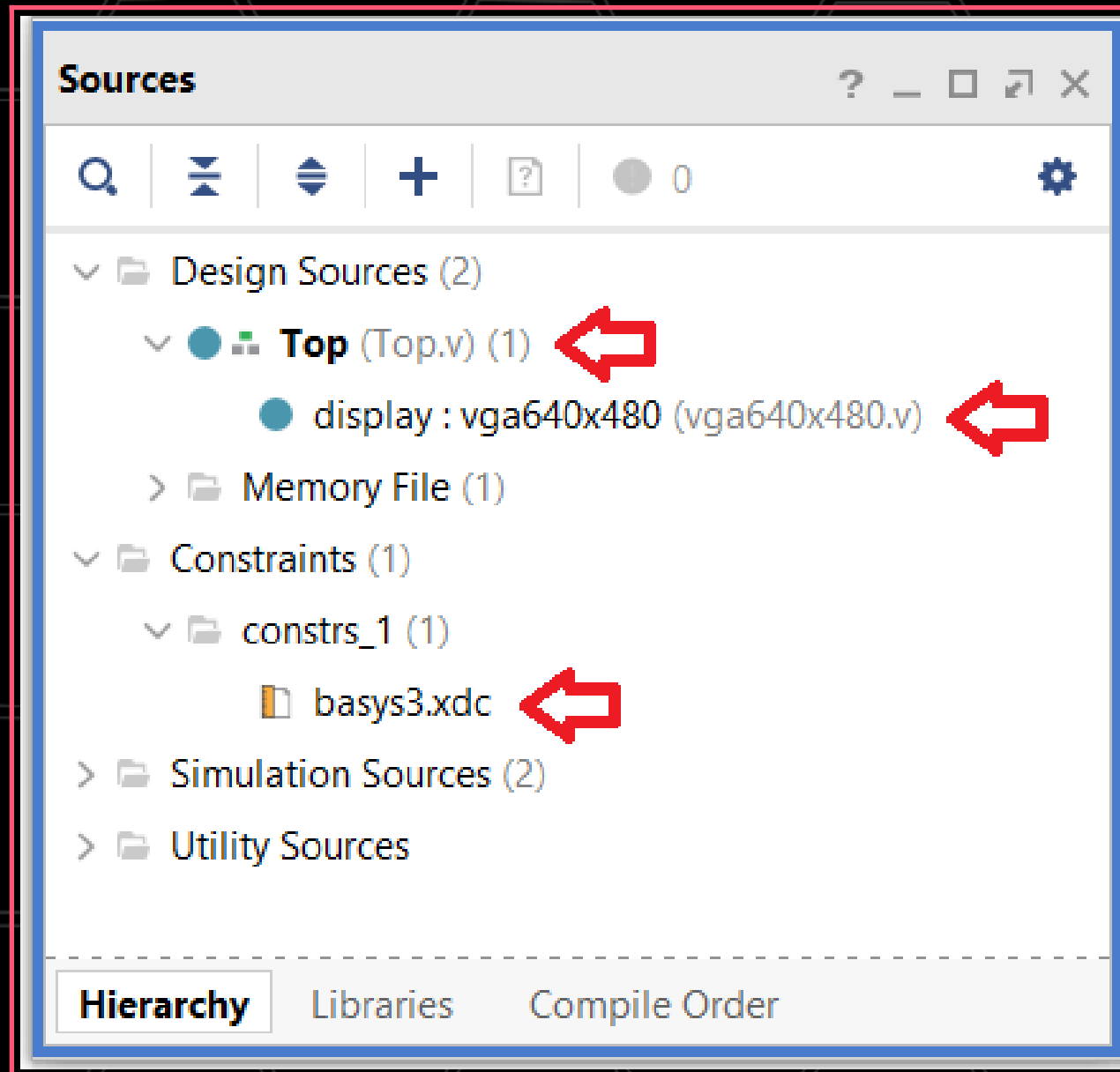
General timing: Horizontal refresh rate: 31.25 kHz or (1 / 32 μ s)
 Vertical refresh rate: 59.5238095238095 Hz or (1 / 16.8 ms)
 Pixel clock: 25 mHz

Horizontal:	Scanline part	Pixels	1 / 25 mHz Pixel Clock	Time (μ s)
	Visible area	640	0.04 μ s	25.6 μ s
	Front porch	16	0.04 μ s	0.64 μ s
	Sync pulse	96	0.04 μ s	3.84 μ s
	Back porch	48	0.04 μ s	1.92 μ s
	Whole line	800	0.04 μ s	32 μ s

Vertical:	Frame part	Lines	1 / 31.25 kHz	Time (ms)
	Visible area	480	0.032 ms	15.36 ms
	Front porch	10	0.032 ms	0.32 ms
	Sync pulse	2	0.032 ms	0.064 ms
	Back porch	33	0.032 ms	1.056 ms
	Whole frame	525	0.032 ms	16.8 ms

Pixel Clock = 800 whole line pixels * 525 vertical lines * 59.5238095238095 Hz (Vertical refresh rate: 1 / 16.8 ms) = 25,000,000 or 25MHz

Explanation Of The Code



01 Top.v module

```
//-----  
// Top Module  
// Digilent Basys 3  
// BeeInvaders Tutorial 1 : Onboard clock 100MHz  
// VGA Resolution: 640x480 @ 60Hz  
// Pixel Clock 25MHz  
//-----  
`timescale 1ns / 1ps  
  
// Setup Top Module  
module Top(  
    input wire CLK,           // Onboard clock 100MHz : INPUT Pin W5  
    input wire RESET,         // Reset button : INPUT Pin U18  
    output wire HSYNC,        // VGA horizontal sync : OUTPUT Pin P19  
    output wire VSYNC,        // VGA vertical sync : OUTPUT Pin R19  
    output reg [3:0] RED,      // 4-bit VGA Red : OUTPUT Pin G19, Pin H19, Pin J19, Pin N19  
    output reg [3:0] GREEN,    // 4-bit VGA Green : OUTPUT Pin J17, Pin H17, Pin G17, Pin D17  
    output reg [3:0] BLUE     // 4-bit VGA Blue : OUTPUT Pin N18, Pin L18, Pin K18, Pin J18  
);  
  
// Setup Reset button  
wire rst = RESET;           // reset : active high (BTNC)
```

This defines the Input / Output pins used in the module: It consists of defining the 100MHz clock (CLK wire input), the reset button (RESET wire input) and the 5 sections of the VGA on the Basys 3 board (HSYNC wire output, VSYNC wire output, RED 4 bit register output, GREEN 4 bit register output and BLUE 4 bit register output).

A wire connecting rst to RESET is defined and this is explained in the vga640x480 module

```
// generate 25MHz pixel clock using a "Fractional Clock Divider"
reg [15:0] counter1;
reg pix_clk;
always @(posedge CLK)
    // divide 100MHz by 4 = 25MHz : (2^16)/4 = 16384 decimal or 4000 hex
    {pix_clk, counter1} <= counter1 + 16'h4000;
```

The 16 bit register counter1 (incremented by 4000 hex each CLK pulse) is added to pix_clk each CLK pulse (see below). Step 5 and Step 9 show counter1 rolling over into bit 16 (setting pix_clk accordingly), this method {pix_clk, counter} is known as a Verilog concatenation operator

Bit counter1 Bits
16 <15----to----00>

Step 1	pix_clk = 0 + 0000000000000000	plus 4000h =
Step 2	pix_clk = 0 + 0100000000000000	plus 4000h =
Step 3	pix_clk = 0 + 1000000000000000	plus 4000h =
Step 4	pix_clk = 0 + 1100000000000000	plus 4000h =
Step 5	pix_clk = 1 + 0000000000000000	plus 4000h =
Step 6	pix_clk = 1 + 0100000000000000	plus 4000h =
Step 7	pix_clk = 1 + 1000000000000000	plus 4000h =
Step 8	pix_clk = 1 + 1100000000000000	plus 4000h =
Step 9	pix_clk = 0 + 0000000000000000	plus 4000h =

To reduce 100MHz clock to 25MHz you would divide 100MHz by 4. Therefore, we divide our 16 bits (2^16 or 1 0000 0000 0000 0000 binary) by 4, this equals 4000 hex or 0100 0000 0000 0000 binary. For greater accuracy you could use more than 16 bit. Notice this method produces an equal duty cycle

The code for controlling pix_clk is explained in the vga640x480 module

```
// instantiate vga640x480 code
wire [9:0] x;                                // pixel x position: 10-bit value: 0-1023 : only need 800
wire [9:0] y;                                // pixel y position: 10-bit value: 0-1023 : only need 525
wire active;                                // high during active pixel drawing
vga640x480 display (
    .i_clk(CLK),
    .i_pix_clk(pix_clk),
    .i_rst(rst),
    .o_hsync(HSYNC),
    .o_vsync(VSYNC),
    .o_x(x),
    .o_y(y),
    .o_active(active)
);
```

This part of the code links the Top module to the vga640x480 module.

An explanation of this will be given under the vga640x480 module

```

// setup palette and RGB registers
reg [7:0] palette [0:192];           // 8 bit values from the 192 hex entries in the colour palette
reg [7:0] COL;                       // holds hex colour palette value to display on the screen
reg [7:0] colourR;                   // 8 bit hex value for RED
reg [7:0] colourG;                   // 8 bit hex value for GREEN
reg [7:0] colourB;                   // 8 bit hex value for BLUE

// load colour palette
initial begin
    $readmemh("pal24bit.mem", palette); // load 192 hex values into "palette"
end

```

This sets up the following registers;

palette: holds all 192 x 8 bit values from the colour palette file (64 colours each with a Red, Green and Blue hex value)

COL: holds an 8 bit hex value (0 to 3F) which points to the colour palette (for the screen fill)

colourR: holds an 8 bit hex value for colour RED

colourG: holds an 8 bit hex value for colour GREEN

colourB: holds an 8 bit hex value for colour BLUE

Load into palette the 192 x 8 bit hex RGB values from the "pal24bit.mem" file


```

// fill the active area of the screen
always @ (posedge CLK)
begin
    COL <= 8'h19;                // set colour to pink (decimal 25)
    if (active)
        begin
            colourR <= palette[(COL*3)];    // retrieve RED palette hex value
            colourG <= palette[(COL*3)+1];  // retrieve GREEN palette hex value
            colourB <= palette[(COL*3)+2];  // retrieve BLUE palette hex value
            RED <= colourR[7:4];             // output 4 left hand bits of the 8 bit RED value retrieved
            GREEN <= colourG[7:4];          // output 4 left hand bits of the 8 bit GREEN value retrieved
            BLUE <= colourB[7:4];           // output 4 left hand bits of the 8 bit BLUE value retrieved
        end
    else
        begin
            RED <= 0;                    // set RED, GREEN & BLUE
            GREEN <= 0;                  // to "0" when x,y outside of
            BLUE <= 0;                  // the active display area
        end
    end
end
endmodule

```

Set COL to 19 hex (25 decimal) which points to the colour pink in the colour palette

When the x,y pixel positions are within the active area (640 x 480);

Retrieve colourR / colourG / colourB values from palette register pointed to by COL

Put the 4 left bits of colourR / colourG / colourB into VGA wires Red / Green / Blue

(the reason being, RGB wires are only 4 bits long and our colour palette values are 8 bit long)

When the x,y pixel positions are outside of the active area set Red / Green / Blue = 0

02 vga640x480.v module

```
//-----  
// vga640x480 Module  
// Digilent Basys 3  
// BeeInvaders : Onboard clock 100MHz  
// VGA Resolution: 640x480 @ 60Hz  
// Pixel Clock 25MHz  
//-----  
`timescale 1ns / 1ps  
  
// Setup vga640x480 Module  
module vga640x480(  
    input wire i_clk,           // 100MHz onboard clock  
    input wire i_pix_clk,       // 25MHz pixel clock  
    input wire i_rst,           // reset  
    output wire o_hsync,        // horizontal sync  
    output wire o_vsync,        // vertical sync  
    output wire o_active,       // high during active pixel drawing  
    output wire [9:0] o_x,      // current pixel x position  
    output wire [9:0] o_y,      // current pixel y position  
);
```

This defines the Input / Output pins used in the module: It consists of defining the 100MHz clock (as `i_clk` a wire input), 25MHz pixel clock (as `i_pix_clk` a wire input), reset button (as `i_rst` a wire input), horizontal sync (as `o_hsync` a wire output), vertical sync (as `o_vsync` a wire output), active pixel area (as `o_active` a wire output), current x position (as `o_x` a wire output) and current y position (as `o_y` a wire output)

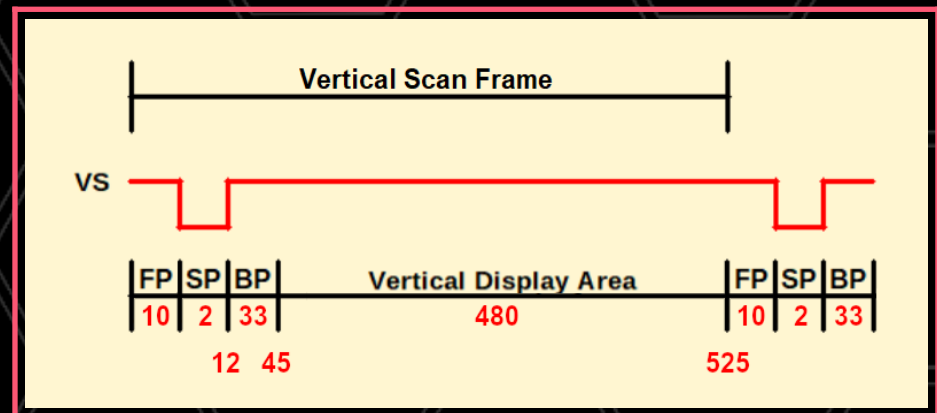
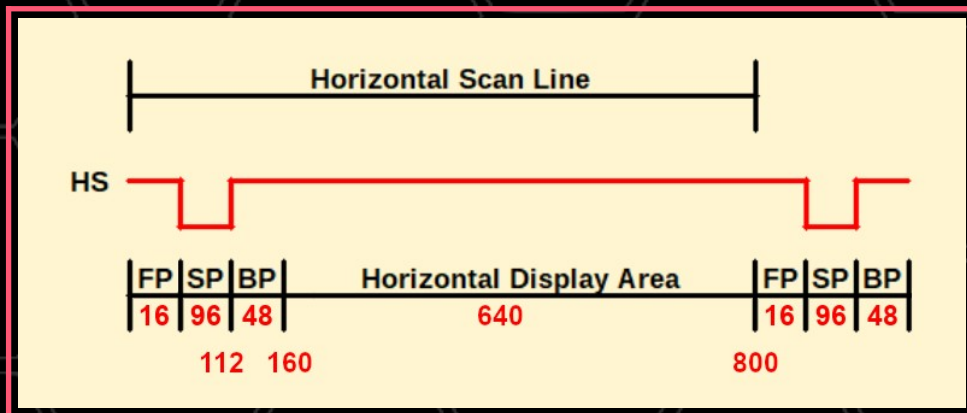
```

// setup VGA timings
//-----
// VGA 640x480 Horizontal Timing (line)
localparam HSYNCSTART = 16;           // horizontal sync start
localparam HSYNCEND = 16 + 96;         // horizontal sync end
localparam HACTIVESTART = 16 + 96 + 48; // horizontal active start
localparam HACTIVEEND = 16 + 96 + 48 + 640; // horizontal active end
reg [9:0] H_SCAN;                       // line position

// VGA 640x480 Vertical timing (frame)
localparam VSYNCSTART = 10;           // vertical sync start
localparam VSYNCEND = 10 + 2;         // vertical sync end
localparam VACTIVESTART = 10 + 2 + 33; // vertical active start
localparam VACTIVEEND = 10 + 2 + 33 + 480; // vertical active end
reg [9:0] V_SCAN;                       // screen position

```

This defines the Start and End of the horizontal sync, vertical sync and active area



Two registers H_SCAN and V_SCAN are also created in respect of the x,y positions

```
// set sync signals to low (active) or high (inactive)
assign o_hsync = ~((H_SCAN >= HSYNCSTART) & (H_SCAN < HSYNCEND));
assign o_vsync = ~((V_SCAN >= VSYNCSTART) & (V_SCAN < VSYNCEND));

// set x and y values
assign o_x = (H_SCAN < HACTIVESTART) ? 0 : (H_SCAN - HACTIVESTART);
assign o_y = (V_SCAN < VACTIVESTART) ? 0 : (V_SCAN - VACTIVESTART);

// set active high during active area
assign o_active = ~((H_SCAN < HACTIVESTART) | (V_SCAN < VACTIVESTART));
```

If H_SCAN falls within the horizontal sync period set output o_hsync = 0 (invert the logical result)

If V_SCAN falls within the vertical sync period set output o_vsync = 0 (invert the logical result)

If H_SCAN falls outside of the active area set output o_x = 0

else H_SCAN falls within the active area and set output o_x = current active x position

If V_SCAN falls outside of the active area set output o_y = 0

else V_SCAN falls within the active area and set output o_y = current active y position

If H_SCAN or V_SCAN fall outside of the active area set output o_active = 0 (invert the logical result of 1). Therefore, if they fall inside the active area set output o_active = 1 (invert the logical result of 0)

```

// check for reset / create frame loop
always @ (posedge i_clk)
begin
    // check for reset button pressed
    if (i_rst)                                // jump to start of a frame and reset registers
    begin
        H_SCAN <= 0;
        V_SCAN <= 0;
    end

    // loop through a full screen
    if (i_pix_clk)
    begin
        if (H_SCAN == HACTIVEEND)            // if at the end of a line update registers
        begin
            H_SCAN <= 0;
            V_SCAN <= V_SCAN + 1;
        end
        else
            H_SCAN <= H_SCAN + 1;              // else increment horizontal counter
        if (V_SCAN == VACTIVEEND)            // if at the end of a screen reset vertical counter
            V_SCAN <= 0;
        end
    end
end
endmodule

```

Check if the Reset button has been pressed and if true jump to the start of a frame

Every 25MHz (i_pix_clk) check;

if H_SCAN has reached the end of a line (HACTIVEEND)

if it has reset H_SCAN to zero and increment V_SCAN

else increment H_SCAN

if V_SCAN has reached the end of a frame (VACTIVEEND) reset V_SCAN to zero

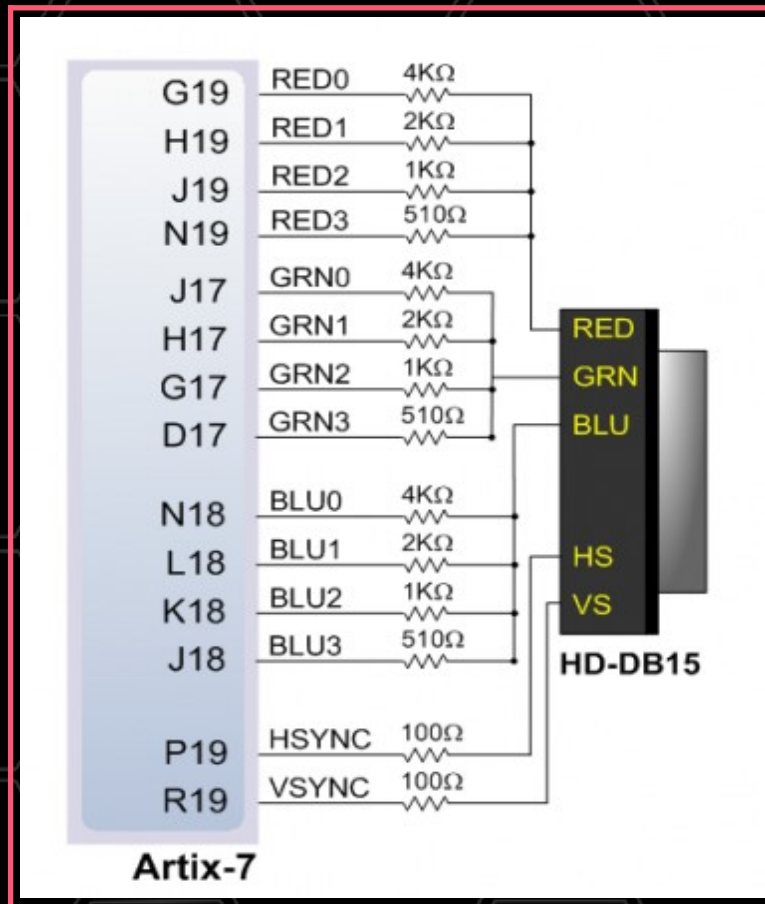
```
##-----  
## Constraints Module  
## Digilent Basys 3  
## BeeInvaders : Onboard clock 100MHz  
## VGA Resolution: 640x480 @ 60Hz  
## Pixel Clock 25MHz  
##-----  
  
## Clock  
set_property -dict {PACKAGE_PIN W5 IOSTANDARD LVCMOS33} [get_ports {CLK}];  
create_clock -add -name sys_clk_pin -period 10.00 \  
    -waveform {0 5} [get_ports {CLK}];  
  
## Use BTNC as Reset Button (active high)  
set_property -dict {PACKAGE_PIN U18 IOSTANDARD LVCMOS33} [get_ports {RESET}];  
  
## VGA Connector  
set_property -dict {PACKAGE_PIN G19 IOSTANDARD LVCMOS33} [get_ports {RED[0]}];  
set_property -dict {PACKAGE_PIN H19 IOSTANDARD LVCMOS33} [get_ports {RED[1]}];  
set_property -dict {PACKAGE_PIN J19 IOSTANDARD LVCMOS33} [get_ports {RED[2]}];  
set_property -dict {PACKAGE_PIN N19 IOSTANDARD LVCMOS33} [get_ports {RED[3]}];  
set_property -dict {PACKAGE_PIN N18 IOSTANDARD LVCMOS33} [get_ports {BLUE[0]}];  
set_property -dict {PACKAGE_PIN L18 IOSTANDARD LVCMOS33} [get_ports {BLUE[1]}];  
set_property -dict {PACKAGE_PIN K18 IOSTANDARD LVCMOS33} [get_ports {BLUE[2]}];  
set_property -dict {PACKAGE_PIN J18 IOSTANDARD LVCMOS33} [get_ports {BLUE[3]}];  
set_property -dict {PACKAGE_PIN J17 IOSTANDARD LVCMOS33} [get_ports {GREEN[0]}];  
set_property -dict {PACKAGE_PIN H17 IOSTANDARD LVCMOS33} [get_ports {GREEN[1]}];  
set_property -dict {PACKAGE_PIN G17 IOSTANDARD LVCMOS33} [get_ports {GREEN[2]}];  
set_property -dict {PACKAGE_PIN D17 IOSTANDARD LVCMOS33} [get_ports {GREEN[3]}];  
set_property -dict {PACKAGE_PIN P19 IOSTANDARD LVCMOS33} [get_ports {HSYNC}];  
set_property -dict {PACKAGE_PIN R19 IOSTANDARD LVCMOS33} [get_ports {VSYNC}];  
  
## Configuration options, can be used for all designs  
set_property CONFIG_VOLTAGE 3.3 [current_design]  
set_property CFGBVS VCCO [current_design]
```


The constraints file should set up all the Input/s and Output/s used in the code

Each type of input or output has a designated pin number and the name used in the code for each input or output is included in the constraints file

PIN	NAME	PIN NAME	DESCRIPTION
W5	CLK	IO_L12P_T1_MRCC_34	Onboard 100MHz clock
U18	RESET	IO_L18N_T2_A11_D27_14	Reset Button
P19	HSYNC	IO_L10P_T1_D14_14	VGA horizontal sync
R19	VSYNC	IO_L10N_T1_D15_14	VGA vertical sync
G19	RED[0]	IO_L4N_T0_D05_14	VGA Red Bit 0
H19	RED[1]	IO_L4P_T0_D04_14	VGA Red Bit 1
J19	RED[2]	IO_L6N_T0_D08_VREF_14	VGA Red Bit 2
N19	RED[3]	IO_L9N_T1_DQS_D13_14	VGA Red Bit 3
J17	GREEN[0]	IO_L7P_T1_D09_14	VGA Green Bit 0
H17	GREEN[1]	IO_L5P_T0_D06_14	VGA Green Bit 1
G17	GREEN[2]	IO_L5N_T0_D07_14	VGA Green Bit 2
D17	GREEN[3]	IO_0_14	VGA Green Bit 3
N18	BLUE[0]	IO_L9P_T1_DQS_14	VGA Blue Bit 0
L18	BLUE[1]	IO_L8P_T1_D11_14	VGA Blue Bit 1
K18	BLUE[2]	IO_L8N_T1_D12_14	VGA Blue Bit 2
J18	BLUE[3]	IO_L7N_T1_D10_14	VGA Blue Bit 3

The VGA section of the Basys 3 board consists of 4 bit Red, 4 bit Green and 4 bit Blue (12 bit or 4096 colours) as shown below



Suggestions

1. Code improvements

Any improvements in the code used are most welcome. Please provide details of this for consideration in using in this tutorial

2. Errors or Mistakes

Any errors or mistakes spotted are most welcome, including incorrect explanations

3. Testbenches

I would like to include Testbenches in the tutorial. It would be most helpful to receive details / explanations of the following (including steps taken to arrive at the results);

- a) 100MHz clock compared to 25MHz clock
- b) Loading of the palette file hex values

Tutorial 2

The next tutorial will include;

1. Using "Gimp" (a free image editor) to obtain the RAW image data / Palette data from a graphics image and how to convert these to hex data files
2. How to display the Bee character at the bottom of the screen
3. A Sprite sheet showing the images / characters used in the game, along with hex data values for each character which can be imported into Gimp
4. As in tutorial 1, suggestions will be welcome / considered, including suggestions on the graphics used

Appendix

01 Basys 3 Board Pins

PIN	BASYS 3	PIN NAME
A1		GND
A10		MGTREFCLK1N_216
A11		DXP_0
A12		VP_0
A13		VREFN_0
A14	JB[0]	IO_L6P_TO_16
A15	JB[4]	IO_L6N_TO_VREF_16
A16	JB[1]	IO_L12P_T1_MRCC_16
A17	JB[5]	IO_L12N_T1_MRCC_16
A18	RsTx	IO_L19N_T3_VREF_16
A19		GND
A2		MGTPTXN1_216
A3		GND
A4		MGTPRXNO_216
A5		GND
A6		MGTPRXN1_216
A7		GND
A8		MGTREFCLKON_216
A9		GND
B1		MGTAVTT
B10		MGTREFCLK1P_216
B11		DXN_0
B12		VREFP_0
B13		VN_0

B14		GND
B15	JB[2]	IO_L11N_T1_SRCC_16
B16	JB[3]	IO_L13N_T2_MRCC_16
B17	PS2Data	IO_L14N_T2_SRCC_16
B18	RsRx	IO_L19P_T3_16
B19		VCCO_16
B2		MGTPTXP1_216
B3		GND
B4		MGTPRXP0_216
B5		GND
B6		MGTPRXP1_216
B7		GND
B8		MGTREFCLKOP_216
B9		GND
C1		MGTAVCC
C10		GND
C11		CCLK_0
C12		GNDADC_0
C13		VCCADC_0
C14		VCCO_16
C15	JB[6]	IO_L11P_T1_SRCC_16
C16	JB[7]	IO_L13P_T2_MRCC_16
C17	PS2Clk	IO_L14P_T2_SRCC_16
C18		VCCO_16
C19		GND
C2		GND
C3		GND
C4		GND
C5		MGTAVTT
C6		GND
C7		MGTREF_216
C8		TCK_0
C9		VCCBATT_0

D1		MGTPTXN0_216
D17	vgaGreen[3]	IO_0_14
D18	QspiDB[0]	IO_L1P_T0_D00_MOSI_14
D19	QspiDB[1]	IO_L1N_T0_D01_DIN_14
D2		MGTPTXP0_216
D3		GND
E1		MGTAVCC
E17		GND
E18		IO_L3P_T0_DQS_PUDC_B_14
E19	led[1]	IO_L3N_T0_DQS_EMCCLK_14
E2		MGTAVTT
E3		GND
F1		GND
F17		VCCO_14
F18	QspiDB[3]	IO_L2N_T0_D03_14
F19		GND
F2		GND
F3		MGTAVCC
G1		GND
G10		VCCINT
G11		GND
G12		VCCO_0
G13		VCCO_16
G17	vgaGreen[2]	IO_L5N_T0_D07_14
G18	QspiDB[2]	IO_L2P_T0_D02_14
G19	vgaRed[0]	IO_L4N_T0_D05_14
G2	JA[3]	IO_L1N_T0_AD4N_35
G3	JA[7]	IO_L1P_T0_AD4P_35
G7		MGTAVTT
G8		GND
G9		MGTAVCC
H1	JA[4]	IO_L3P_T0_DQS_AD5P_35
H10		VCCINT

H11		GND
H12		GND
H13		VCCAUX
H17	vgaGreen[1]	IO_L5P_T0_D06_14
H18		GND
H19	vgaRed[1]	IO_L4P_T0_D04_14
H2	JA[6]	IO_L2P_T0_AD12P_35
H3		VCCO_35
H7		GND
H8		GND
H9		MGTAVCC
J1	JA[0]	IO_L3N_T0_DQS_AD5N_35
J10		VCCINT
J11		GND
J12		GND
J13		VCCAUX
J17	vgaGreen[0]	IO_L7P_T1_D09_14
J18	vgaBlue[3]	IO_L7N_T1_D10_14
J19	vgaRed[2]	IO_L6N_T0_D08_VREF_14
J2	JA[2]	IO_L2N_T0_AD12N_35
J3	JXADC[0]	IO_L7P_T1_AD6P_35
J7		VCCO_35
J8		GND
J9		GND
K1		VCCO_35
K12		VCCO_14
K13		VCCO_14
K17	JC[0]	IO_L12N_T1_MRCC_14
K18	vgaBlue[2]	IO_L8N_T1_D12_14
K19	QspiCSn	IO_L6P_T0_FCS_B_14
K2	JA[5]	IO_L5P_T0_AD13P_35
K3	JXADC[4]	IO_L7N_T1_AD6N_35
K7		VCCO_35

K8		GND
L1	led[15]	IO_L6N_T0_VREF_35
L10		VCCINT
L11		GND
L12		VCCO_14
L13		VCCO_14
L17	JC[4]	IO_L12P_T1_MRCC_14
L18	vgaBlue[1]	IO_L8P_T1_D11_14
L19		GND
L2	JA[1]	IO_L5N_T0_AD13N_35
L3	JXADC[1]	IO_L8P_T1_AD14P_35
L7		VCCO_35
L8		GND
L9		GND
M1	JXADC[6]	IO_L9N_T1_DQS_AD7N_35
M10		VCCINT
M11		VCCBRAM
M12		VCCO_14
M13		GND
M17		VCCO_14
M18	JC[1]	IO_L11P_T1_SRCC_14
M19	JC[5]	IO_L11N_T1_SRCC_14
M2	JXADC[2]	IO_L9P_T1_DQS_AD7P_35
M3	JXADC[5]	IO_L8N_T1_AD14N_35
M7		VCCO_35
M8		VCCO_34
M9		GND
N1	JXADC[7]	IO_L10N_T1_AD15N_35
N10		VCCINT
N11		VCCBRAM
N12		GND
N13		GND
N17	JC[2]	IO_L13P_T2_MRCC_14

N18	vgaBlue[0]	IO_L9P_T1_DQS_14
N19	vgaRed[3]	IO_L9N_T1_DQS_D13_14
N2	JXADC[3]	IO_L10P_T1_AD15P_35
N3	led[13]	IO_L12P_T1_MRCC_35
N7		VCCO_34
N8		VCCO_34
N9		GND
P1	led[14]	IO_L19N_T3_VREF_35
P17	JC[6]	IO_L13N_T2_MRCC_14
P18	JC[3]	IO_L14P_T2_SRCC_14
P19	Hsync	IO_L10P_T1_D14_14
P2		GND
P3	led[12]	IO_L12N_T1_MRCC_35
R1		VCCO_34
R17		VCCO_14
R18	JC[7]	IO_L14N_T2_SRCC_14
R19	Vsync	IO_L10N_T1_D15_14
R2	sw[15]	IO_L1P_T0_34
R3	sw[11]	IO_L2P_T0_34
T1	sw[14]	IO_L3P_T0_DQS_34
T17	btnR	IO_L17P_T2_A14_D30_14
T18	btnU	IO_L17N_T2_A13_D29_14
T19		GND
T2	sw[10]	IO_L1N_T0_34
T3	sw[9]	IO_L2N_T0_34
U1	sw[13]	IO_L3N_T0_DQS_34
U10		M2_0
U11		INIT_B_0
U12		DONE_0
U13		VCCO_14
U14	led[6]	IO_25_14
U15	led[5]	IO_L23P_T3_A03_D19_14
U16	led[0]	IO_L23N_T3_A02_D18_14

U17	btnD	IO_L18P_T2_A12_D28_14
U18	btnC	IO_L18N_T2_A11_D27_14
		IO_L15P_T2_DQS_RDWR_B_1
U19	led[2]	4
U2	an[0]	IO_L9N_T1_DQS_34
U3	led[11]	IO_L9P_T1_DQS_34
U4	an[1]	IO_L11P_T1_SRCC_34
U5	seg[4]	IO_L16P_T2_34
U6		GND
U7	seg[6]	IO_L19P_T3_34
U8	seg[2]	IO_L14P_T2_SRCC_34
U9		GND
V1		VCCO_34
V10		PROGRAM_B_0
V11		CFGBVS_0
V13	led[8]	IO_L24P_T3_A01_D17_14
V14	led[7]	IO_L24N_T3_A00_D16_14
V15	sw[5]	IO_L21P_T3_DQS_14
V16	sw[1]	IO_L19P_T3_A10_D26_14
		IO_L19N_T3_A09_D25_VREF
V17	sw[0]	_14
V18		GND
		IO_L15N_T2_DQS_DOUT_CS
V19	led[3]	O_B_14
V2	sw[8]	IO_L5P_T0_34
V3	led[9]	IO_L6P_T0_34
V4	an[2]	IO_L11N_T1_SRCC_34
V5	seg[5]	IO_L16N_T2_34
V6		VCCO_34
V7	dp	IO_L19N_T3_VREF_34
V8	seg[3]	IO_L14N_T2_SRCC_34
V9		VCCO_0
W1		GND

W10		TDI_0
W11		M1_0
W12		GND
W13	sw[7]	IO_L22P_T3_A05_D21_14
W14	sw[6]	IO_L22N_T3_A04_D20_14
		IO_L21N_T3_DQS_A06_D22_14
W15	sw[4]	IO_L20P_T3_A08_D24_14
W16	sw[2]	IO_L20N_T3_A07_D23_14
W17	sw[3]	IO_L16P_T2_CSI_B_14
W18	led[4]	IO_L16N_T2_A15_D31_14
W19	btnL	IO_L5N_T0_34
W2	sw[12]	IO_L6N_T0_VREF_34
W3	led[10]	IO_L12N_T1_MRCC_34
W4	an[3]	IO_L12P_T1_MRCC_34
W5	clk	IO_L13N_T2_MRCC_34
W6	seg[1]	IO_L13P_T2_MRCC_34
W7	seg[0]	TDO_0
W8		TMS_0
W9		

02

VGA Timing Specifications for Various VGA Modes

Resolution (pixels)	Refresh Rate (Hz)	Pixel Clock (MHz)	Horizontal (pixel clocks)				Vertical (rows)				h_sync Polarity	v_sync Polarity
			Display	Front Porch	Sync Pulse	Back Porch	Display	Front Porch	Sync Pulse	Back Porch		
640x350	70	25.175	640	16	96	48	350	37	2	60	p	n
640x350	85	31.5	640	32	64	96	350	32	3	60	p	n
640x400	70	25.175	640	16	96	48	400	12	2	35	n	p
640x400	85	31.5	640	32	64	96	400	1	3	41	n	p
640x480	60	25.175	640	16	96	48	480	10	2	33	n	n
640x480	73	31.5	640	24	40	128	480	9	2	29	n	n
640x480	75	31.5	640	16	64	120	480	1	3	16	n	n
640x480	85	36	640	56	56	80	480	1	3	25	n	n
640x480	100	43.16	640	40	64	104	480	1	3	25	n	p
720x400	85	35.5	720	36	72	108	400	1	3	42	n	p
768x576	60	34.96	768	24	80	104	576	1	3	17	n	p
768x576	72	42.93	768	32	80	112	576	1	3	21	n	p
768x576	75	45.51	768	40	80	120	576	1	3	22	n	p
768x576	85	51.84	768	40	80	120	576	1	3	25	n	p
768x576	100	62.57	768	48	80	128	576	1	3	31	n	p
800x600	56	36	800	24	72	128	600	1	2	22	p	p
800x600	60	40	800	40	128	88	600	1	4	23	p	p
800x600	75	49.5	800	16	80	160	600	1	3	21	p	p
800x600	72	50	800	56	120	64	600	37	6	23	p	p
800x600	85	56.25	800	32	64	152	600	1	3	27	p	p
800x600	100	68.18	800	48	88	136	600	1	3	32	n	p
1024x768	43	44.9	1024	8	176	56	768	0	8	41	p	p

1024x768	60	65	1024	24	136	160	768	3	6	29	n	n
1024x768	70	75	1024	24	136	144	768	3	6	29	n	n
1024x768	75	78.8	1024	16	96	176	768	1	3	28	p	p
1024x768	85	94.5	1024	48	96	208	768	1	3	36	p	p
1024x768	100	113.31	1024	72	112	184	768	1	3	42	n	p
1152x864	75	108	1152	64	128	256	864	1	3	32	p	p
1152x864	85	119.65	1152	72	128	200	864	1	3	39	n	p
1152x864	100	143.47	1152	80	128	208	864	1	3	47	n	p
1152x864	60	81.62	1152	64	120	184	864	1	3	27	n	p
1280x1024	60	108	1280	48	112	248	1024	1	3	38	p	p
1280x1024	75	135	1280	16	144	248	1024	1	3	38	p	p
1280x1024	85	157.5	1280	64	160	224	1024	1	3	44	p	p
1280x1024	100	190.96	1280	96	144	240	1024	1	3	57	n	p
1280x800	60	83.46	1280	64	136	200	800	1	3	24	n	p
1280x960	60	102.1	1280	80	136	216	960	1	3	30	n	p
1280x960	72	124.54	1280	88	136	224	960	1	3	37	n	p
1280x960	75	129.86	1280	88	136	224	960	1	3	38	n	p
1280x960	85	148.5	1280	64	160	224	960	1	3	47	p	p
1280x960	100	178.99	1280	96	144	240	960	1	3	53	n	p
1368x768	60	85.86	1368	72	144	216	768	1	3	23	n	p
1400x1050	60	122.61	1400	88	152	240	1050	1	3	33	n	p
1400x1050	72	149.34	1400	96	152	248	1050	1	3	40	n	p
1400x1050	75	155.85	1400	96	152	248	1050	1	3	42	n	p
1400x1050	85	179.26	1400	104	152	256	1050	1	3	49	n	p
1400x1050	100	214.39	1400	112	152	264	1050	1	3	58	n	p
1440x900	60	106.47	1440	80	152	232	900	1	3	28	n	p
1600x1200	60	162	1600	64	192	304	1200	1	3	46	p	p
1600x1200	65	175.5	1600	64	192	304	1200	1	3	46	p	p

1600x1200	70	189	1600	64	192	304	1200	1	3	46	p	p
1600x1200	75	202.5	1600	64	192	304	1200	1	3	46	p	p
1600x1200	85	229.5	1600	64	192	304	1200	1	3	46	p	p
1600x1200	100	280.64	1600	128	176	304	1200	1	3	67	n	p
1680x1050	60	147.14	1680	104	184	288	1050	1	3	33	n	p
1792x1344	60	204.8	1792	128	200	328	1344	1	3	46	n	p
1792x1344	75	261	1792	96	216	352	1344	1	3	69	n	p
1856x1392	60	218.3	1856	96	224	352	1392	1	3	43	n	p
1856x1392	75	288	1856	128	224	352	1392	1	3	104	n	p
1920x1200	60	193.16	1920	128	208	336	1200	1	3	38	n	p
1920x1440	60	234	1920	128	208	344	1440	1	3	56	n	p
1920x1440	75	297	1920	144	224	352	1440	1	3	56	n	p