*NOTE*

If you wish to migrate a design from an old ISE project, please see the document at https://www.xilinx.com/support/documentation/sw_manuals/xilinx2012_2/ug911-vivado-migration.pdf

To create a new project in Vivado, select the “Create Project” option in the Quick Start menu.

Follow the prompts as shown below, selecting a location and name for the new project, selecting “RTL Project” as the type.

Pick a name for your project,
If you already have source files, or do not want to create them now, click the Do not specify sources button below the RTL Project selection.

If you have an old ISE project, you can also import that here with the Imported Project option, however you may need to make manual changes to the project and ucf files to ensure compatibility with new tools. See document at beginning of tutorial for more information.
For our demonstration project, select Next, and follow the dialog,

Continue on to select the board to be used in the project, follow the options below to select the ARTIX-7 chip we will be using. The Part number is xc7a100tcsg324-1.

You now see a summary for the created project, go through it and make any changes if needed. Note, changes to these settings can be made half way through the project too,
You now see the project open, on a screen similar to this,

The sources are available on the right hand side of the page. Go ahead and create a new design source by right clicking on the design sources in the sources field and then new source. The following window will be prompted, click on the create sources.
Choose a name for the source and a location for the source.

The created source is here, hit finish to complete creating a project,
You can now choose to define your i/o ports, or declare them internally on the project, for now we will declare them internally. Click either OK or Cancel, and we will define our module with the test code provided.

The source files created can be found in the Sources menu at the top left, double click on any file to open it for editing. For now, select the Verilog file you created.

Copy the test code from Appendix 1 into the .v file. Ensure no errors appear due to improper copying (check newlines and ; terminations)
Do the same for the xdc file created earlier, with the generic XDC provided in Appendix 2.

Click the green play button on the menu, Press F11, or click Run Synthesis on the left navigator bar to begin project synthesis. This may take some time, progress can be monitored from either the Project Summary screen, or the Design Runs menu at the bottom of screen.

Once Synthesis is complete, the following window will pop up. Select Run Implementation and press ok to finish implementation of the synthesized code. This may take some time, progress can be monitored from either the Project Summary screen, or the Design Runs menu at the bottom of screen.
Once Implementation is complete, the following window will appear, select Generate Bitstream to create the programming file to send to the board.
Once the bitstream file has been generated, open the hardware manager to program the FPGA. Ensure the board is connected to the computer and powered on.
Click “Auto Connect” in the top green bar, or Open Target > Auto Connect in the left navigation bar.

Click Program Device Either at bottom of left menu, or in top green bar.
Ensure program works properly, 3 LEDs on bottom right side of board should count light up in order from right to left.

To reset, move SW0 on far right to UP position, then return to DOWN to allow counter to run again.

Appendix 1

```
`timescale 1ns / 1ps

/////////////////////////////////////////////////////////////////////////////
// Company: UMBC
// Engineer: Uttej Kallakuri
//
// Create Date: 15:52:27 01/18/2018
// Design Name:
// Module Name: counter
// Project Name:
// Target Devices:
// Tool versions:
```
module counter(clk, rst, count1, count2, count3);
input clk, rst;
output count1, count2, count3;
reg countout1, countout2, countout3;
reg[32:0] count_temp;
always @(posedge clk)
begin
if(rst) begin
    count_temp<= 32'd0;
    countout1<= 1'b0;
    countout2<= 1'b0;
    countout3<= 1'b0;
end
else if(count_temp == 32'b0000_0011_1111_1111_1111_1111_1111_1111) begin
    count_temp<= 32'd0;
    countout1<= countout1 + 1;
    countout2<= countout2 + 1;
end
else if(count_temp == 32'b0000_0000_0000_1111_1111_1111_1111_1111) begin
    count_temp<= 32'd0;
    countout1<= countout1 + 1;
    countout2<= countout2 + 1;
    countout3<= countout3 + 1;
end
endmodule
countout2 <= countout2 + 1;
count_temp <= count_temp + 1;
end
else if(count_temp == 32'b0000_0111_1111_1111_1111_1111_1111_1111) begin
  count_temp <= 32'd0;
countout3 <= countout3 + 1;
count_temp <= count_temp + 1;
end
else
  count_temp <= count_temp + 1;
end
assign count1 = countout1;
assign count2 = countout2;
assign count3 = countout3;
endmodule

Appendix 2

### This file is a general .xdc for the Nexys4 DDR Rev C board
### To use it in a project:
### - uncomment the lines corresponding to used pins
### - rename the used signals according to the project
### Clock signal
set_property LOC E3 [get_ports clk];
set_property IOSTANDARD LVCMOS33 [get_ports clk];
# I think this is the proper "new" way to instantiate the clock, but i am not 100% sure
#create_clock -name clk100mhz -period 10ns [get_ports clk]

### Switches
set_property LOC J15 [get_ports rst];
set_property IOSTANDARD LVCMOS33 [get_port rst];

# for future use
#set_property LOC L16 [get_ports r<1>];
#set_property IOSTANDARD LVCMOS33 [get_ports r<1>];
#set_property LOC M13 [get_ports rst];
#set_property IOSTANDARD LVCMOS33 [get_ports r<2>];
#set_property LOC R15 [get_ports r<3>];
#set_property IOSTANDARD LVCMOS33 [get_ports r<3>];
#set_property LOC R17 [get_ports g<0>];
#set_property IOSTANDARD LVCMOS33 [get_ports g<0>];
#set_property LOC T18 [get_ports g<1>];
#set_property IOSTANDARD LVCMOS33 [get_ports g<1>];
#set_property LOC U18 [get_ports g<2>];
#set_property IOSTANDARD LVCMOS33 [get_ports g<2>];
#set_property LOC R13 [get_ports g<3>];
#set_property IOSTANDARD LVCMOS33 [get_ports g<3>];
#set_property LOC T8 [get_ports b<0>];
#set_property IOSTANDARD LVCMOS33 [get_ports b<0>];
#set_property LOC U8 [get_ports b<1>];
#set_property IOSTANDARD LVCMOS33 [get_ports b<1>];
#set_property LOC R16 [get_ports b<2>];
#set_property IOSTANDARD LVCMOS33 [get_ports b<2>];
#set_property LOC T13 [get_ports b<3>];
#set_property IOSTANDARD LVCMOS33 [get_ports b<3>];
#set_property LOC H6 [get_ports sw<12>];
#set_property IOSTANDARD LVCMOS33 [get_ports sw<12>];
#set_property LOC U12 [get_ports sw<13>];
#set_property IOSTANDARD LVCMOS33 [get_ports sw<13>];
#set_property LOC U11 [get_ports sw<4>];
#set_property IOSTANDARD LVCMOS33 [get_ports sw<4>];
#set_property LOC V10 [get_ports sw<15>];
#set_property IOSTANDARD LVCMOS33 [get_ports sw<15>];

##VGA Connector

#set_property LOC A3 [get_ports red<0>];
#set_property IOSTANDARD LVCMOS33 [get_ports red<0>];
#set_property LOC B4 [get_ports red<1>];
#set_property IOSTANDARD LVCMOS33 [get_ports red<1>];
#set_property LOC C5 [get_ports red<2>];
#set_property IOSTANDARD LVCMOS33 [get_ports red<2>];
#set_property LOC A4 [get_ports red<3>];
#set_property IOSTANDARD LVCMOS33 [get_ports red<3>];
#set_property LOC C6 [get_ports green<0>];
#set_property IOSTANDARD LVCMOS33 [get_ports green<0>];
#set_property LOC A5 [get_ports green<1>];
#set_property IOSTANDARD LVCMOS33 [get_ports green<1>];
#set_property LOC B6 [get_ports green<2>];
#set_property IOSTANDARD LVCMOS33 [get_ports green<2>];
#set_property LOC A6 [get_ports green<3>];
#set_property IOSTANDARD LVCMOS33 [get_ports green<3>];
#set_property LOC B7 [get_ports blue<0>];
#set_property IOSTANDARD LVCMOS33 [get_ports blue<0>];
#set_property LOC C7 [get_ports blue<1>];
#set_property IOSTANDARD LVCMOS33 [get_ports blue<1>];
#set_property LOC D7 [get_ports blue<2>];
#set_property IOSTANDARD LVCMOS33 [get_ports blue<2>];
#set_property LOC D8 [get_ports blue<3>];
## LEDs

set_property LOC H17 [get_ports count1];
set_property IOSTANDARD LVCMOS33 [get_ports count1];
set_property LOC K15 [get_ports count2];
set_property IOSTANDARD LVCMOS33 [get_ports count2];
set_property LOC J13 [get_ports count3];
set_property IOSTANDARD LVCMOS33 [get_ports count3];

# For future

#set_property LOC N14 [get_ports led<3>];
#set_property IOSTANDARD LVCMOS33 [get_ports led<3>];
#set_property LOC R18 [get_ports led<4>];
#set_property IOSTANDARD LVCMOS33 [get_ports led<4>];
#set_property LOC V17 [get_ports led<5>];
#set_property IOSTANDARD LVCMOS33 [get_ports led<5>];
#set_property LOC U17 [get_ports led<6>];
#set_property IOSTANDARD LVCMOS33 [get_ports led<6>];
#set_property LOC U16 [get_ports led<7>];
#set_property IOSTANDARD LVCMOS33 [get_ports led<7>];
#set_property LOC V16 [get_ports led<8>];
#set_property IOSTANDARD LVCMOS33 [get_ports led<8>];
#set_property LOC T15 [get_ports led<9>];
#set_property IOSTANDARD LVCMOS33 [get_ports led<9>];
#set_property LOC U14 [get_ports led<10>];
#set_property IOSTANDARD LVCMOS33 [get_ports led<10>];
#set_property LOC T16 [get_ports led<11>];
#set_property IOSTANDARD LVCMOS33 [get_ports led<11>];
#set_property LOC V15 [get_ports led<12>];
#set_property IOSTANDARD LVCMOS33 [get_ports led<12>];
#set_property LOC V14 [get_ports led<13>];
#set_property IOSTANDARD LVCMOS33 [get_ports led<13>];
#set_property LOC V12 [get_ports led<14>];
#set_property IOSTANDARD LVCMOS33 [get_ports led<14>];
#set_property LOC V11 [get_ports led<15>];
#set_property IOSTANDARD LVCMOS33 [get_ports led<15>];