These instructions are a combination of “MicroBlaze_Install_Short_Version” and “Lec19_Install_Short_Version”, tailored for lab3.

==========

1. Creating New Project

1.1) Click on **Create New Project** (i.e,. Lab3)
Be sure there are no spaces in the folder name or path
(i.e., don’t use “Lab 3”, but instead use “Lab3”)
Otherwise, the tool chain will crash later on.
and click **Next**.
1.2) Choose Project Type as **RTL Project**. Leave the “Do not specify sources…” box unchecked and click **Next**. On the Add Sources screen, don’t add sources, but ensure the Target Language and Simulator Language are both VHDL, not Verilog. **Next**. Don’t add constraints. **Next**
1.3) select **Nexys Video** board. **Next**.
1.4) A summary of the new project design sources and target device is displayed. Click **Finish**.

2. Creating New Block Design

2.2) On the left you should see the Flow Navigator. Select **Create Block Design** under the IP Integrator. Keep the name as **design_1**, click **OK**.
2.3) Click the **Add IP button (plus sign)**. Search for “**Microblaze**” and double click on it to add the IP block to your empty design.

3. Adding Microblaze IP and Customization

3.1) click **Run Block Automation**
3.3) Change default settings in the block options
- Local Memory = **32KB**
- Cache Configuration = **16KB**
- Interrupt Controller = **Do not check**

and click **OK**.

4. Customization of Clock Wizard IP Block

4.1) **Double click** on the **Clock Wizard**, clk_wiz_1, IP block.
4.2) Choose **sys clock** for **CLK_IN1**.
Choose **reset** for **EXT_RESET_IN**.
4.3) Select the **Output Clocks** tab.
4.4) **Check the box** next to **clk_out2**, then select clk_out2 output frequency as **200.000** (Mhz) and set **Reset Type** as **Active Low**. (scroll to bottom of window)
4.5) click **OK** to finish block automation of Clock Wizard.

5. Adding UART IP Block

5.1) Go to **Add IP (plus sign)** and search for “**UART**”. Select the **AXI Uartlite** IP block.

6. Running Connection Automation for the First Time

6.1) Now select the **Run Connection Automation** from the Designer Assistance bar message prompt. Select **axi_uartlite_0**, **clk_wiz_1**, and **rst_clk_wiz_1_100M**. Do NOT select **microblaze_0**.

7. Adding and Customizing Memory Interface Generator IP Block

7.1) Click **Add IP (plus sign)** and search for "**Memory Interface Generator**", then double click the result.
7.2) click **Run Block Automation**. Click **OK**.
7.4) If you see this one error message [BD 41-1273]. You can ignore this. Click **OK** to dismiss this. If you have more than one error message, redo. Delete the MIG block from the schematic, and redo step 7. Sometimes several MIG errors occur when the path name for your repo is too long. In this case, move your repo to a higher level directory.

8. Running Connection Automation for the Second Time

8.1) Now click on **Run Connection Automation**
8.2) **Select** only the **mig_7series_0** in the connection automation list.
- click on **sys_clk_i** and change **clock source** to **clock_out2 200MHz**.
Do not select **Microblaze_0** section.
Click **OK**.

10. Make DDR3 Signal External

10.1) The MIG block should be named **mig_7series_0**. Place your cursor on this symbol **||** next to the **DDR3+** port name. Your cursor will change to look like a pencil. **Right click** here and in the drop down list select **Make External**

11. Validate Design

11.1) Select **Validate Design** (**check box symbol or F6**).
If you get an error message, see the original tutorial.
11.2) Success? Click **OK**.

From "Lec19_Install_Short_Version"

1. Open your Lab3 Vivado project (if not already opened)

Go to **Tools → Create and package IP → next**

2. Create your custom IP project

2.1) Select **Create a new AXI4 peripheral** and click **Next**
2.2) Input "**My_Lab2**" and click **Next**
2.3) Change the number of Registers to **32** on the AXI interface and click **Next**
2.4) Select **Edit IP** and click **Finish**

3. Designing the IP core

3.1) A new instance of Vivado will open up for the new IP core. Expand the top level file **My_Lab2_IP_v1_0** to see **My_Lab2_IP_v1_0_S00_AXI**

6. Modify **My_Lab2_IP_v1_0.vhd**

5.1) add **Lines 19, 59, 93** as needed, similar as you did in HW#10, adding ports for all the signals that need to go to your lab2.xdc ports

5.2) Looking at the lab3 block diagram, you'll see that the "**ready**" bit (or FlagQ) will need to pushed out to a higher level for the interrupt, similar to the 3 lines you added in HW#11 for "**roll**"

5. Modify **My_Lab2_IP_v1_0_S00_AXI.vhd**

Similar to HW#10 and HW#11, you will need to add all the slv_regs you are mapping to your lab2 ports.

Add **Lines 20 (**add **lab2.xdc ports,** plus a similar line for **ready/flaqQ),**

Add lines **112-122, (**put **ready/flagQ** in entity).

Also add needed signal wires (usually for lab2 ports microblaze will want to read on slv_regs.)

Near line **671,** connect the lab2 ports microblaze will want to read on slv_regs, similar to what you did for HW#10, for example,

```
case loc_addr is
  when b"00000" =>
    reg_data_out <= X"000000" & std_logic_vector(Q);
  when b"00001" =>
    reg_data_out <= slv_reg1;  -- keep as is
```

Add lines like **759-766 in HW#10,** instantiating lab2 datapath, and hooking up the microblaze slv_reg for it to write to

May need CSA statements to hook lab2 datapath signals, like **clearFlag** to the appropriate bit of a **slv_reg,** or to hook top level **ready** to **flagQ**

4. Add Lab2 files to the My_Lab2_IP_v1_0

4.1) "**Add Sources**" → **lab2_dp.vhd** file [and many more files supporting lab2]

You will need to recreate clk_wiz_0 and clk_wiz_1 (from lab#2)… Select "Global" synthesis option when "generating"

7. Packaging the IP core

7.0) Now click on **Package IP** in the Flow Navigator and you should see the Package IP tab.

7.1) Select **Compatibility** (under Packaging Steps) and make sure "Artix7" are present. If those are not there, you can add them by clicking the plus button. The Life Cycle does not matter at this point.

7.2) Select **Customization Parameters** and select the line for **Merge Changes from Customization Parameters Wizard**.

7.3) Select **Customization GUI**. This is where we get to change our graphical interface. No changes at this time.

7.3) Select **File Groups**.

For Lab#3, you need to move synthesis files from advanced to standard (so can compile mixed vhdl and Verilog…)

- Your .vhd and .v files are probably inside Advanced->Synthesis
- Click "+" and select Standard->Synthesis to create this new File Group
- Open Advanced->Synthesis and select all these files and drag them into Standard->Synthesis
- Then delete the files inside Advanced->Synthesis
- If in the line for Standard->Synthesis the Model Name does not say the name of your IP (such as My_Lab2_v1_0), then type your IP name in this square

and when done, select the line for **Merge Changes..**.

7.4) select **Review and Package** and click the **Re-package IP** button.

7.5) A popup will ask if you want to close the project, Select **Yes**.

8. Add Custom IP to your design

8.1) In the project manager page of the original window, click **Open Block Design**.
8.2) Use the **Add IP** (**plus sign**) button to add your **My_Lab2** IP you just created
8.3) Find your **My_Lab2_ip_v1.0** block in the circuit diagram. Right click on output pins (like **LEDs** in HW#10, which should be your lab2.xdc signals) and select **Make External**
8.4) Notice the **ready/flagQ** signal is exposed. We need to manually connect it to MicroBlaze **interrupt** pin.
- Click the **'+' sign** by the MicroBlaze **Interrupt,** and it will expand to 3 pins
- Click on the My_Lab2's **Ready/FlagQ** Signal and drag to the MicroBlaze's **Interrupt** pin and release.
8.5) run **Connection Automation**
8.5) Now you need to add a constraints file to add the LED net to the pins on the Artix 7 chip by adding the constraints **Lab2.xdc** file. (Ensure the names of the output pins match the diagram… Mine were called **xxxxx_0**. For Lab#3, almost all the xdc signals needed "_0" added)
Add sources → add or create constraints → Lab3.xdc

10. Verify Addressing Design

10.1) Click the **Address Editor** tab (next to **Diagram** tab)
10.2) Verify the addresses and range for the components match that of slide#15
Uart is at 0x4060_0000; my_counter is at 0x44A0_0000

11. Validate Design

11.1) Select **Validate Design** (**check box symbol or F6**).

12. Creating or Regenerate the HDL System Wrapper

12.1) right click on ***design_1*** and select **Create HDL Wrapper**.
Let Vivado manage the wrapper.

13. Generating Bit File

13.1) In the **Flow Navigator panel** on the left, under **Program and Debug** select the **Generate Bitstream** option.
13.3) After the bitstream has been generated, a message prompt will pop-up on the screen. You don't have to open the Implemented Design for this demo. Just click on **Cancel**.
[Note: one MIG error [BD 41-1273] is okay]

14. Exporting Hardware Design to SDK

14.1) On the top left corner of the window, from the tool bar click on **File** and select **Export Hardware**. Make sure the generated **bitstream** is included by **checking the box**.

15. Launching SDK

15.1) Go to **File** and select **Launch SDK** and click **OK**.

17. Creating New Application Project in SDK

17.1) Go to **File** in the main tool bar and select **New → Application Project**.
Project Name = **Lab3**
Create New under **Board Support Package**.
Click **Next**.

18. Selecting Hello World Application from available templates

18.1) Select **Hello World** under *Available Templates* on the left panel and click **Finish**.

18.2) **Lab3** is our main working source folder.
18.3) Replace with our C-code (similar to lec19.c)
- I just opened **hello_world.c** and cut-n-pasted the code from **lec19.c** over the code in hello_world.c, and then modified this to create my lab3.c code.

19. Verify Linker Script File for Memory Region Mapping & Stack/Heap

19.0) **Double click** on the ***lscript.ld*** to open.
19.1) In the linker script, take a look at the **Section to Memory Region Mapping** box. If you did the *Make DDR3 External* step then the target memory region **must** read **mig_7series_0**. Scroll down to check if this applies to all rows. If for any region it does not say **mig_7series_0**, then click on the row under the **Memory Region** column and select **mig_7series_0**.
19.2) My stack size = **0x400** and Heap size = **0x800** [we made need to increase this when our C program gets larger]
19.3) microblaze…bram… size = **0x7FB0**

20. Programming FPGA with Bit File

20.1) Make sure that the Nexys Video board is turned on and connected to the host PC with the provided micro USB cable. Then click on the **Program FPGA** button to open the Program FPGA window. Make sure that the ***Hardware Platform*** is selected as **design_1_wrapper_hw_platform_0**.
In the software configuration box, under *ELF File to Initialize in Block RAM ()* column, the row option must read **bootloop**. If not, click on the row and select **bootloop**.
Now click on **Program**.

21. Run Configuration Settings for STDIO Connection

21.1) From the *Project Explorer* panel, **right click** on the **Lab3** project folder. At the bottom of the drop down list, select **Run As** and then select **Run Configurations**.

The Run Configurations window is divided into two main sections. In the left panel, click on **Xilinx C/C++ application(GDB)** and then select **Lab3.elf**. *Note: In case you see* **Lab3** ***Debug*** *instead of lab3.elf in this step, you can still run it without any issues.*

Now click on **Apply** and **Run**.

22. Use Tera Term Terminal Emulator

Note: SDK appears to no longer support UART messages in its console, so we will need to use an external terminal emulator like Tera Term. http://en.wikipedia.org/wiki/Tera_Term (http://en.wikipedia.org/wiki/Tera_Term) to know what Tera Term is. You can download and install Tera Term from this link http://ttssh2.sourceforge.jp/index.html.en

Establish a serial connection with the correct communication port inside Tera Term. Tera Term may find your COM port your USB is using automatically. If not you can find the COM port your USB is using my going to windows **Device Manager** and clicking on **Ports (COM & LPT)**. Mine is sometimes **COM3** and sometimes **COM5**. Typically the settings will be **8 Data Bits**, **No Parity Bit**, **1 Stop Bit**.

23.1) "**Welcome to Lab3**" will be displayed on the Console tab
Type "?" to see list of commands

If the ISR is working, every time the counter rolls over, the isr count should increase.