

## EE3220 System-on-Chip Design - 2024/25 Spring

Assignment 3 - Due Date: April 21 2025 - 11:59pm

Submission Method - Canvas Online Assignment

Name: \_\_\_\_\_

Date: \_\_\_\_\_

Student ID: \_\_\_\_\_

Mark: \_\_\_\_\_

---

Submission guidelines:

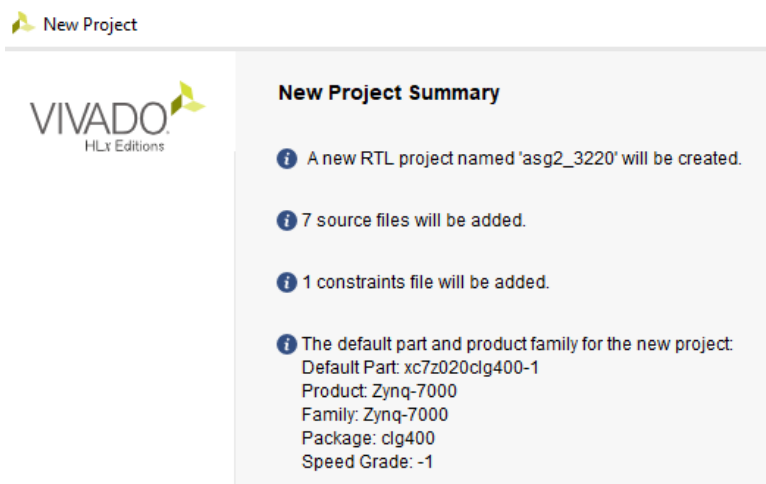
- Please prepare your assignment in "PDF" format. (1%)
- Please rename your file to "EE3220 – assignment 3 – studentID.pdf". (1%)
- For late submission, 10% deduction per day.
- Checklist for the report

|                                                                                                                                                                |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1. screenshot on the revised ssn_1 report                                                                                                                      |
| 2. screenshot on the Schematic Diagram generated in the Synthesis Design                                                                                       |
| 3. screenshot on the utilization report generated in the Synthesis Design showing the submodule of U0 and U1                                                   |
| 4. answer to Question 1                                                                                                                                        |
| 5. screenshots on 1) the Source Clock Path summary in the timing report and 2) the Source Clock Path in schematic form                                         |
| 6. screenshot on the utilization report generated in the Implementation Design showing the submodule of U0 and U1                                              |
| 7. answer to Question 2                                                                                                                                        |
| 8. screenshot on the timing summary report generated in the revised Implementation Design showing that the timing violations of the Intra-clock paths are gone |
| 9. analysis and other record for relative steps                                                                                                                |

Please review the tutorials and use Xilinx Vivado to complete the task about synthesizing an RTL design, and submit a report to record relative steps and answer the questions. Please make sure **every item in the above checklist** is included in your report before submission.

**a. Create a New Vivado Project using IDE:**

1. The new Vivado Project should be selected as an **RTL Project targeting the XC7Z020clg400-1 device**. We would use **Verilog** as the *Target Language* and *Simulator Language*.
2. In the *Add Sources* form, add all the Verilog files provided (*led\_ctl.v*, *meta\_harden.v*, *uart\_baud\_gen.v*, *uart\_led.v*, *uart\_rx.v*, *uart\_rx\_ctl.v* and *uart\_top.v*) into the project.
3. In the *Add Constraints* form, add *uart\_led\_timing\_pynq.xdc* into the project (Please leave *uart\_led\_pins\_pynq.xdc* file **NOT** included for a while).
4. Please make sure that you have created the project similar with Tutorial 7. The Summary of New Project is attached here for you to double check.



5. Go to the provided Tcl script - *ps7\_create\_pynq.tcl*. In the line 7 – 8, please input the Project Name (*prjName*) and the Project Directory (*prjDir*) in the command below (Hint: The Project Directory is where the .xpr file is placed).

Example:

```
set prjName systh_3220
set prjDir C:/Teaching/systh_3220
```

This script will create a block design called system, instantiate ZYNQ PS with one GPIO channel (GPIO14) and one EMIO channel. It will then create a top-level wrapper file called *system\_wrapper.v* which will instantiate the *system.bd* (the block design). You can check the contents of the TCL files to confirm the commands that are being run.

6. Generate the PS design by executing the revised Tcl script. You can click **Tools > Run TCL Script** to execute the revised Tcl script.
7. Analyze the design source files hierarchy.

- a) In the *Sources* pane, expand the `uart_led` entry and notice hierarchy of the lower-level modules.
- b) Double-click on the `uart_led` entry to view its content.

```

23
24 `timescale 1ns/1ps
25 module uart_led (
26     // Write side inputs
27     input      clk_pin,      // Clock input (from pin)
28     input      rst_pin,      // Active HIGH reset (from pin)
29     input      btn_pin,      // Button to swap high and low bits
30     input      rxd_pin,      // RS232 RXD pin - directly from pin
31     output [7:0] led_pins    // 8 LED outputs
32 );
33
34 //*****
35 // Parameter definitions
36 //*****
37 parameter BAUD_RATE      = 115_200;
38 parameter CLOCK_RATE     = 125_000_000;
39

```

Notice in the Verilog code, the `BAUD_RATE` and `CLOCK_RATE` parameters are defined to be 115200 and 125 MHz respectively. Also notice that the lower level modules are instantiated. The `meta_harden` modules are used to synchronize the asynchronous reset and push-button inputs.

- c) Expand `U0` and `uart_rx_i0` instance to see its hierarchy.  
This module uses the baud rate generator and a finite state machine. The `rxd_pin` is sampled at a x16 the baud rate

## 8. Open the `uart_led_timing_pynq.xdc` source and analyze the content

- a) In the *Sources* pane, expand the *Constraints* folder and double-click the `uart_led_timing_pynq.xdc` entry to open the file in text mode.

```

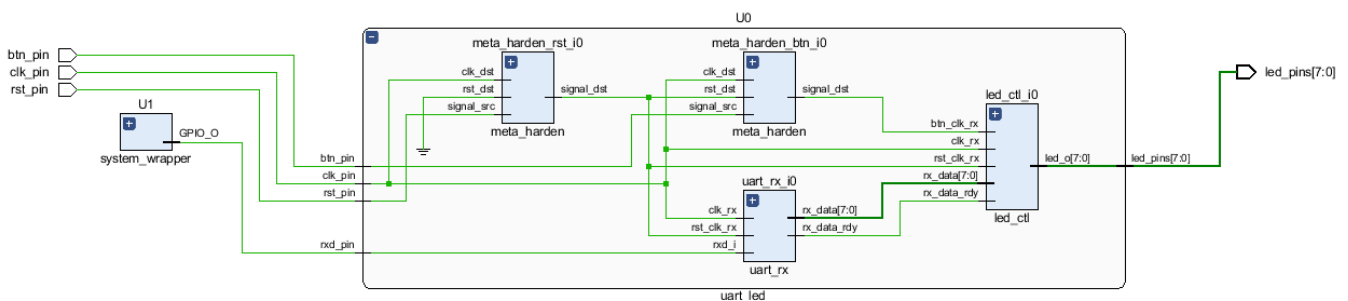
1  ##pynq Timing
2
3  # define clock and period
4  create_clock -period 8.000 -name clk_pin -waveform {0.000 4.000} [get_ports clk_pin]
5
6  # Create a virtual clock for IO constraints
7  create_clock -period 12.000 -name virtual_clock
8
9  #input delay
10 set_input_delay -clock virtual_clock -max 0.000 [get_ports rst_pin]
11 set_input_delay -clock virtual_clock -min -0.500 [get_ports rst_pin]
12
13 set_input_delay -clock virtual_clock -max 0.000 [get_ports btn_pin]
14 set_input_delay -clock virtual_clock -min -0.500 [get_ports btn_pin]
15
16 #output delay
17 set_output_delay -clock virtual_clock 0.000 [get_ports {led_pins[*]}]

```

Line 4 creates the period constraint of 8ns with a duty cycle of 50%. Line 7 creates a virtual clock of 12 ns. This clock can be viewed as the upstream device is generating its output with respect to its clock and outputs data with respect to it. The btn\_pin is constrained with respect to the upstream clock (lines 13, 14). The led\_pins are constrained with respect to the upstream clock as the downstream device may be using it.

## b. Elaborate the Design:

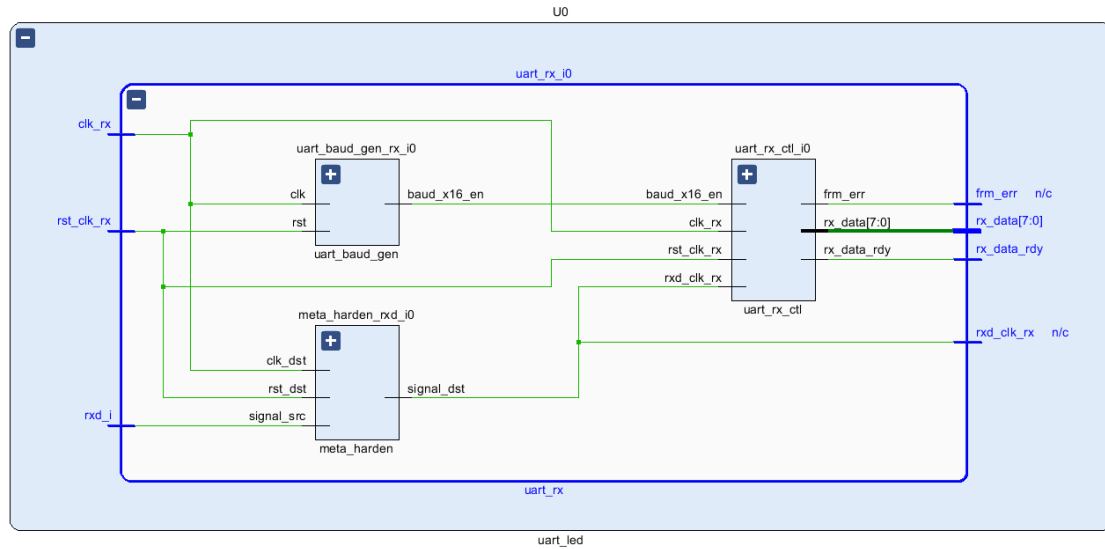
1. Elaborate and perform the RTL analysis on the source file. Click **Flow Navigator > RTL ANALYSIS > Open Elaborated Design > Schematic**. Click **OK**. The model (design) will be elaborated and a logical view of the design is displayed.



You will see two components at the top-level; going down one level in component U0 shows 2 instances of meta\_harden, one instance of uart\_rx, and one instance of led\_ctl.

2. To see where the uart\_rx\_i0 gets generated, right-click on the uart\_rx\_i0 instance and select **Go To Source** and see that line 84 in the source code is generating it.

- Double-click on the `uart_rx_i0` instance in the schematic diagram to see the underlying components.



- Click on **Flow Navigator > RTL Analysis > Open Elaborated Design > Report Noise**. Click **OK** to generate the report named `ssn_1`.
- View the `ssn_1` report and observe the unplaced ports, Summary, and I/O Bank Details are highlighted in red because the pin assignments were not

Tcl ConsoleMessagesLogReportsDesign RunsNoise x

Summary

Messages (2)

IO Bank Details

Links

Q

⚙

⌕

IO Bank Details

| Name               | Port        | IO Std    | Vcco | Slew | Drive Strength (... | Off-Chip Termina... | Remaining Margin ... | Notes                            |
|--------------------|-------------|-----------|------|------|---------------------|---------------------|----------------------|----------------------------------|
| IO Bank 0 (0)      |             |           |      |      |                     |                     |                      |                                  |
| IO Bank 13 (0)     |             |           |      |      |                     |                     |                      |                                  |
| IO Bank 34 (0)     |             |           |      |      |                     |                     |                      |                                  |
| IO Bank 35 (0)     |             |           |      |      |                     |                     |                      |                                  |
| Unplaced Ports (8) |             |           |      |      |                     |                     |                      |                                  |
| led_pins[0]        | led_pins[0] | LVC MOS18 | 1.80 | SLOW | 12                  | FP_VTT_50           |                      | CRITICAL WARNING - Unplaced port |
| led_pins[1]        | led_pins[1] | LVC MOS18 | 1.80 | SLOW | 12                  | FP_VTT_50           |                      | CRITICAL WARNING - Unplaced port |
| led_pins[2]        | led_pins[2] | LVC MOS18 | 1.80 | SLOW | 12                  | FP_VTT_50           |                      | CRITICAL WARNING - Unplaced port |
| led_pins[3]        | led_pins[3] | LVC MOS18 | 1.80 | SLOW | 12                  | FP_VTT_50           |                      | CRITICAL WARNING - Unplaced port |
| led_pins[4]        | led_pins[4] | LVC MOS18 | 1.80 | SLOW | 12                  | FP_VTT_50           |                      | CRITICAL WARNING - Unplaced port |
| led_pins[5]        | led_pins[5] | LVC MOS18 | 1.80 | SLOW | 12                  | FP_VTT_50           |                      | CRITICAL WARNING - Unplaced port |
| led_pins[6]        | led_pins[6] | LVC MOS18 | 1.80 | SLOW | 12                  | FP_VTT_50           |                      | CRITICAL WARNING - Unplaced port |
| led_pins[7]        | led_pins[7] | LVC MOS18 | 1.80 | SLOW | 12                  | FP_VTT_50           |                      | CRITICAL WARNING - Unplaced port |

done. Note that only output pins are reported as the noise analysis is done on the output pins.

- Click on **Add Sources** under the Project Navigator, select **Add or create constraints option** and click **Next**.
- Click on the **Blue Plus button**, then the **Add Files...** button and browse to the provided directory in canvas, select the `uart_led_pins_pynq.xdc` file, click **OK**, and then click **Finish** to add the pins location constraints. [Notice that the sources are modified and the tools detect it, showing a warning status bar to **re-load the design**.]
- Click on the **Reload** link. The constraints will be processed.
- Click on **Report Noise** and click **OK** to generate the report named `ssn_1`. Observe that this time it does not show any errors.

Please take a screenshot on the revised ssn\_1 report, and attach it to your assignment report.

**c. Synthesize the Design:**

1. Click on **Flow Navigator > SYNTHESIS > Run Synthesis**.
2. Click **Save** if the Save Project dialog box is displayed.  
The synthesis process will be run on the *uart\_top.v* and all its hierarchical files. When the process is completed a *Synthesis Completed dialog* box with three options will be displayed.
3. Select the **Open Synthesized Design** option and click **OK** as we want to look at the synthesis output.  
Click **Yes** to close the elaborated design if the dialog box is displayed.
4. Click on **Flow Navigator > SYNTHESIS > Open Synthesized Design > Schematic** to view the synthesized design in a schematic view.  
Notice that IBUF and OBUF are automatically instantiated (added) to the design as the input and output are buffered. There are still four lower level modules instantiated.  
Please take a screenshot on the Schematic Diagram generated in the Synthesis Design, and attach it to your assignment report.
5. Double-click on U0 and uart\_rx\_i0 instances in the schematic view to see the underlying instances.
6. Select the uart\_baud\_gen\_rx\_i0 instance, right-click, and select **Go To Source**.  
Notice that line 86 is highlighted. Also notice that the CLOCK\_RATE and BAUD\_RATE parameters are passed to the module being called.
7. Double-click on the meta\_harden\_rxd\_i0 instance to see how the synchronization circuit is being implemented using two FFs. This synchronization is necessary to reduce the likelihood of meta-stability.

**Generate the utilization and power reports.**

8. Click **Flow Navigator > SYNTHESIS > Open Synthesized Design > Report Report Utilization**, and click **OK** to generate the utilization report.
9. Click on **Hierarchy** in the left pane to see the complete hierarchy utilization.

Please take a screenshot on the utilization report generated in the Synthesis Design showing the submodule of U0 and U1, and attach it to your assignment report.

10. Click **Flow Navigator > SYNTHESIS > Open Synthesized Design > Report Power**, and click **OK** to generate the estimated power consumption report using default values.

Note that this is just an estimate as no simulation run data was provided and no accurate activity rate, or environment information was entered.

#### Question 1

From the power report, find the % power consumption used by each of the following (You can move the mouse on the boxes which do not show the percentage to see the consumption):

| Resource | Number |
|----------|--------|
| Clocks:  | _____  |
| Signals: | _____  |
| Logic:   | _____  |
| I/O:     | _____  |
| PS7:     | _____  |

#### Analyze the timing report

11. Click on **Flow Navigator > SYNTHESIS > Open Synthesized Design > Report Timing Summary**.
12. Click **OK** to generate the Timing\_1 report.



Notice that the Design Timing Summary and Inter-Clock Paths entry in the left pane is highlighted in red indicating timing violations. In the right pane, the information is grouped in Setup, Hold, and Width columns.

Under the Setup column Worst Negative Slack (WNS) is linked indicating that clicking on it can give us insight on how the failing path has formed. The Total Negative Slack (TNS) is highlighted in red indicating the total amount of violations in the design and the Number of Failing Endpoints indicate total number of failing paths.

13. Click on **the link beside the Worst Negative Slack (WNS)** and see the 8 failing paths. Note that this is an estimate only. The nets are specified as

unplaced and have all been allocated default values. No actual routing delays are considered.

14. Double-click on the **Path 25** to see a detailed view of the path. The path report shows four sections: (i) Summary, (ii) Source Clock Path, (iii) Data Path, and (iv) Destination Clock Path.
15. Select Path 25 in the timing summary panel, or the Path summary view, right-click, and select **Schematic**. The schematic for the output data path will be displayed.
16. In order to see how the Source Clock Path is made up in schematic form, double-click on **left end of the C pin** of the FDRE in the schematic. This will show the net between the BUFG and C port of the FDRE.
17. Similarly, double-click on **the left end of the BUFG** to see the path between IBUF and BUFG.
18. Finally, double-click on **the input pin of IBUF** to see the path between the clock input pin and the IBUF.

Please take a screenshots on 1) the Source Clock Path summary in the timing report (step 14) and 2) the Source Clock Path in schematic form (step 16-18 above), and attach them to your assignment report.

**Since the virtual clock is slower (12 ns) than the clk\_pin period (8 ns), the data path delay includes the clock period of the clk\_pin clock source.**

1. Click **Edit Timing Constraints** under the Synthesized Design. The Timing Constraints GUI will appear, showing the design has two create clocks, four inputs, and one output constraints. It also shows the constraints in the text form in the All Constraints section.
2. Click in the Period cell of the virtual\_clock and change the period **from 12 to 8**. Click **Apply**. Click on **Rerun**. Notice that setup timing violations are gone. However, there are still 2 failing paths for the Hold.
3. Click on the **Worst Hold Slack (WHS) link** to see the paths.
4. Double-click on the first path to see the timing compositions. Notice that the clock path delay does not include the entire clock period.
5. Select **File > Constraints > Save...**
6. Click **Update**.

7. Click **OK** and then **Yes** to save the synthesized design. Notice that the **Synthesis Out-of-Date** status is displayed on the top-right corner.

**d. Implement the Design:**

1. Click the **Close Design** link in the status bar. If prompted, do not save anything.
2. Click on the **Run Implementation** in the Flow Navigator pane.
3. Click **OK** when prompted to run the synthesis first before running the implementation process.  
When the implementation is completed, a dialog box will appear with three options.
4. Select the **Open Implemented Design** option and click OK.

**View the amount of FPGA resources consumed by the design using Report Utilization.**

1. In the Flow Navigator pane, select **IMPLEMENTATION > Open Implemented Design > Report Utilization**. The Report Utilization dialog box opens.
2. Click **OK**. The utilization report is displayed at the bottom of the Vivado IDE. You can select any of the resources on the left to view its corresponding utilization.

Please take a screenshot on the utilization report generated in the Implementation Design showing the submodule of U0 and U1, and attach it to your assignment report.

**Question 2**

Is there any difference between utilization reports generated in the Synthesis and Implementation? Please give your explanation.

**Generate a timing summary report**

3. Click **Flow Navigator > IMPLEMENTATION > Open Implemented Design > Report Timing Summary**. The Report Timing Summary dialog box opens.
4. Leave all the settings unchanged and click **OK** to generate the report.

- Click on the **WNS link** to see a detailed report to determine the failing path entries.

| Source Clock Path              |           |          |                       |                              |
|--------------------------------|-----------|----------|-----------------------|------------------------------|
| Delay Type                     | Incr (ns) | Path ... | Location              | Netlist Resource(s)          |
| (clock clk_pin rise edge)      | (r) 0.000 | 0.000    |                       |                              |
|                                | (r) 0.000 | 0.000    | Site: H16             | clk_pin                      |
| net (fo=0)                     | 0.000     | 0.000    |                       | clk_pin                      |
|                                |           |          | Site: H16             | clk_pin_IBUF_inst/I          |
| IBUF (Prop ibuf I O)           | (r) 1.451 | 1.451    | Site: H16             | clk_pin_IBUF_inst/O          |
| net (fo=1, routed)             | 2.522     | 3.973    |                       | clk_pin_IBUF                 |
|                                |           |          | Site: BUF...TRL_X0Y16 | clk_pin_IBUF_BUFG_inst/I     |
| BUFG (Prop bufg I O)           | (r) 0.101 | 4.074    | Site: BUF...TRL_X0Y16 | clk_pin_IBUF_BUFG_inst/O     |
| net (fo=59, routed)            | 2.057     | 6.131    |                       | U0/led_ctl_i0/CLK            |
| FDRE                           |           |          | Site: SLICE_X113Y104  | U0/led_ctl_i0/led_o_reg[3]/C |
| Data Path                      |           |          |                       |                              |
| Delay Type                     | Incr (ns) | Path ... | Location              | Netlist Resource(s)          |
| FDRE (Prop fdre C Q)           | (r) 0.456 | 6.587    | Site: SLICE_X113Y104  | U0/led_ctl_i0/led_o_reg[3]/Q |
| net (fo=1, routed)             | 1.526     | 8.112    |                       | led_pins_OBUF[3]             |
|                                |           |          | Site: M14             | led_pins_OBUF[3]_inst/I      |
| OBUF (Prop obuf I O)           | (r) 3.581 | 11.693   | Site: M14             | led_pins_OBUF[3]_inst/O      |
| net (fo=0)                     | 0.000     | 11.693   |                       | led_pins[3]                  |
|                                |           |          | Site: M14             | led_pins[3]                  |
| Arrival Time                   |           | 11.693   |                       |                              |
| Destination Clock Path         |           |          |                       |                              |
| Delay Type                     | Incr (ns) | Path ... | Loca...               | Netlist Resourc...           |
| (clock virtual...ck rise edge) | (r) 8.000 | 8.000    |                       |                              |
| ideal clock network latency    | 0.000     | 8.000    |                       |                              |
| clock pessimism                | 0.000     | 8.000    |                       |                              |
| clock uncertainty              | -0.025    | 7.975    |                       |                              |
| output delay                   | -0.000    | 7.975    |                       |                              |
| Required Time                  |           | 7.975    |                       |                              |

Compared to delays from the synthesis report, the net delays are actual delays (rather than an estimated figure). The data path delay is longer than the destination clock path delay giving a negative slack (violation). The data path delay is 11.693 ns for the PYNQ-Z2, the destination clock path is 7.975 ns and the negative slack is -3.718 ns.

At this point we can ignore this violation as the LED display change by a few nanoseconds won't be observable by human eyes. We can also change the output delay by -4 ns and make the timings meet.

- Change the output delay by -4 ns and add the following min delay constraint in the *uart\_led\_timing\_pynq.xdc* file.

```
set_output_delay -clock virtual_clock -4.000 [get_ports {led_pins[*]}]
set_output_delay -clock virtual_clock -min -1.000 [get_ports {led_pins[*]}]
```

- Click on **Save File** button to save the modified constraint file and then click on the **Run Implementation** in the Flow Navigator pane to implement the project with new constraint file.

8. Click **OK** when prompted to run the synthesis first before running the implementation process.
9. When the implementation is completed, a dialog box will appear with three options.
10. Select the **Open Implemented Design** option and click **OK**.
11. Click **Flow Navigator > IMPLEMENTATION > Open Implemented Design > Report Timing Summary**

Please take a screenshot on the timing summary report generated in the revised Implementation Design showing that the timing violations of the Intra-clock paths are gone, and attach it to your assignment report.

-----END-----