Exercise 4:
Synthesizing your design

Submission Deadline: 17/02/2024 23:59

1 Exercise Description

At this point, we have finished writing, simulating, and debugging our small test design and now it's time to start the backend flow. The first task is to synthesize our design and see how the RTL we wrote turns into a sea of gates.

1.1 Step 1: Cloning the Synthesis Environment

Clone the git repo of the homework, create a branch and check it out:

```
tsmc65
cd DVD
git clone https://gitlab.com/dvd24/dvd24_hw/hw4.git
cd hw5
git checkout v1.0 -b <ID>
```

The synthesis directory includes the following folders:

- `sourcecode`: This folder will link to the RTL files of your design.
- `scripts`: This folder will store your scripts, such as the main synthesis TCL script.
- `inputs`: This folder will store your project definitions, SDC files, and others.
- `reports`: This folder will store the output reports from your synthesis run.
- `exports`: This folder will include the exported netlist and any other files you save.
- `docs`: This folder has the tutorial documentation for this exercise.

1.2 Step 2: Set up your design for synthesis

Copy your RTL files into the `sourcecode` directory, modify the RTL file list and the `.defines` file in `inputs` directory according to your design.

Run **Genus** and follow the flow in `scripts/genus.tcl`, as described in class:

- **Set up your libraries**, and make sure you understand all **warnings**.
- **Read your RTL** and fix any syntax errors.
- **Elaborate** your design, and check that the generic netlist correctly represents your behavioral code, i.e., the **correct number of registers** are inferred, there are **no latches inferred**, etc.

1.3 Step 3: Define your design constraints

Edit your constraints file (`.sdc`) and then **read in** your constraints. Your constraints file should have, at the minimum, the following features:

- Clock definition
- Input/Output delays defined for all ports
- Input drivers on all inputs and output loads on the output
- Clock and reset networks should be defined as ideal
- Optionally, add jitter (clock uncertainty)

The following specifications should be parameterized, using a TCL variable:

- Clock period
- Input and output delays
- Name of clock and reset ports
- Driving cell name and output load. Note that it is recommended to derive the output load from the library information in your design.

After reading in your SDC file, you should run `check_timing_intent` and ensure that you have no timing problems, due to mistakes in your SDC.

1.4 Step 4: Synthesis and Timing

At this point, you are ready to synthesize your design. Run synthesis with the constraints you have set and report the timing of the design. Try and understand the timing report and make sure that the worst path is, in fact, real. If not, define a timing exception for this path (e.g., `set_false_path`).

Now, try and ***push*** your design. In other words, change your clock period to try and achieve the fastest design possible. You can do this by changing your clock period definition and then running incremental synthesis.

After reaching the fastest design possible, remove your design (or restart **Genus**) and re-run synthesis from the beginning with your new constraint. See if you can push it a bit more.

1.5 Step 5: Reporting and Design Export

After synthesis, write out the reports and carefully look at them. You can also use the GUI to look at your design, even down to the schematic level. See that what you got actually is what you wanted. If so, you are ready to export your design.

Run `write_hdl` to write out your synthesized netlist to the `exports` directory.

2 Homework Submission Instructions

2.1 Homework Parameters

As usual, there is a table with parameters according to your ID. Let us define the 9-digit ID number of each student as **ABCDEFGHI**. With this definition, find the values of ***DONT_USE***, ***COUNT***, and ***AREA***, in the table below:

C	*DONT_USE*	E	*COUNT*	H	*AREA*
0	AO21B_X2M_A9TR	0	INV	0	NOR2
1	AO21_X3M_A9TR	1	BUF	1	NAND2
2	INV_X0P5B_A9TR	2	DFF	2	DFF
3	LATRPQN_X3M_A9TR	3	NAND2	3	BUF
4	MX2_X3B_A9TR	4	NOR2	4	INV
5	MXT2_X4M_A9TR	5	INV	5	NOR2
6	NAND2XB_X2M_A9TR	6	BUF	6	NAND2
7	NAND3XXB_X1M_A9TR	7	DFF	7	DFF
8	NOR2B_X4M_A9TR	8	NAND2	8	BUF
9	OAI21B_X1P4M_A9TR	9	NOR2	9	INV

2.2 What to do with the parameters

Now that you know your parameters, follow these instructions for your synthesis:

1) Make sure your gatelevel implementation does not use the standard cell defined by the ***DONT_USE*** parameter.

To do this, use the avoid database attribute of the cell, for example

```
set_db $my_library_cell .avoid true
```

2) Count the number of instances of the standard cell of the type defined by the ***COUNT*** parameter appear in your gatelevel implementation.

To do this, we recommend writing a simple script that finds all the cells of this type and counts them.

3) Calculate the total area consumed by standard cells instances of the type defined by the ***AREA*** parameter.

To do this, we recommend making a bit more fancy script than above that iterates over the cells you found, reads their area attribute, and accumulates it.

2.3 Homework submission

To submit your homework, please **commit** and **push** your reports and export directories to your design branch. In addition, add a text file in the `reports` directory, called `my_summary.txt`. In this file, write the following data in the given order:

- Name and ID – (The first and last name must be one name each!)
- Name of TOPMODULE
- Smallest clock period that your design was able to meet with <5ps negative slack.
- The ***COUNT*** parameter and the number of instances counted
- The ***AREA*** parameter and the total calculated area

For example, your my_summary.txt file should look something like this:

```
John Smith 012345678
TOPMODULE:    my_top
Period:       1.500 ns
COUNT:        INV    2345
AREA:         DFF    1234.78
```

Add to the git all the files that you have in hw4 directory, including:

```
reports/my_summary.txt
scripts/genus.tcl
inputs/average.postsyn.v

exports/
inputs/
reports/
scripts/
sourcecode/
```

2.4 A reminder how to submit your homework with git:

```
git add <filename1> <filename2> <sub_dir_name> ...
git commit -m "Some meaningful message, which describes the commit"
git push origin <ID>
```

Remember – any questions that you have regarding the reports and the data extraction can easily be answered by opening the documentation and the support site!