ECC Encoder & Decoder

Digital Design and Logical Synthesis for Electric Computer Engineering (36113611) Course Project

Synthesis

Version 1.0

Revision History

Rev	Description	Done By	Date
1.0	Initial document	Yuval Yoskovits, Roy Shor	18-Dec-2021
2.0	Final Report	Yuval Saar, Katrin Nechin	30-Dec-2021
3.0			

Contents

LIST OF TABLES

1. INTRODUCTION

In the third part of the project, you will perform the synthesis flow as taught in Lab3 for your own Hardware (Verilog design). You are required to convert your design from Verilog code to Gate Level code (Netlist) using the synthesis tool "Design Compiler", then analyze the result using the extracted reports.

Note: In this part of the project your design is supposed to be working.

1.1 Design Constraints Requirements

Try to increase the clock frequency as high as you can achieve. Then put other some constraints to your DC run.

The system constraints that you should adjust and use in the synthesis flow are as follows:

Constraint Type	Value
Clock Period	At least 100KHz
Clock uncertainty	Maximum 0.1 ns
Clock transition time	Maximum 0.1 ns
Input delay	Maximum 0.2 ns
Input transition	Maximum 0.1 ns
Output delay	Maximum 0.5 ns
Design area	Smaller than 50,000,000
Wire load model	tsmc18_wl50

Table 1: Design Constraints Requirements.

1.2 The Used Constraints

Fill the following table with the design constrains you used in this part:

If may add or remove rows to table if you added more constraints to your synthesis.

False Path (if there are any)			
From Pin/Port	Through Pin	To Pin/Port	Comments
Clock Definitions			

Constraint Name	clk	Comment
Period [Nano Seconds]	15	370MHz
Pin/Port name	clk	
Uncertainty	0.1ns	
Transition	0.1ns	
External Delay (input/output delay)		
Pin name	Value	Comment
Input delay	0.2ns	
Input transition	0.1ns	
Output delay	0.5ns	

Table 2: Used Constraints

1.3 Synthesis Flow

Apply the Synthesis flow as explained in Lab3 using the given Library and given TCL scripts:

- Add your own design files (Verilog codes) to the RTL directory (folder).
- Update the Synthesis scripts according to your own module's names (file name).
- Update the Constrains scripts according to Table 2 you created.
- Upload the DC directory (folder) to your server (as shown in Lab3).
- Run dc_shell command from inside the DC directory using the terminal (also shown in Lab3).
- Use the updated scripts to run the synthesis flow on your design.

2. REPORTS & RESULTS

After completing the synthesis flow, download the "output" and "report" directory to your own workstation. Then open the reports and perform the needed analysis, show, and explain them in this part report.

2.1 Area Report

The design area report includes a total area of 1456854 micro-meter squared.
If we had more time for our project we could save space by removing gates which are not used for every time of parameter. (For example, the decoder keeps a multiplication result vector of size 5 even when it needs to be just of size 4.

```
****************************************
Report : area
Design : ECC_ENC_DEC
Version: O-2018.06-SP4
Date   : Thu Dec 30 20:16:37 2021
****************************************

Library(s) Used:

    slow (File: /users/agnon/pp2022/katrinn/lab3/DC/LibraryFiles/db/slow.db)

Number of ports:                         2480
Number of nets:                          5041
Number of cells:                         2845
Number of combinational cells:           2466
Number of sequential cells:               226
Number of macros/black boxes:               0
Number of buf/inv:                        937
Number of references:                      28

Combinational area:              65556.691875
Buf/Inv area:                    17666.510588
Noncombinational area:           15790.420725
Macro/Black Box area:                0.000000
Net Interconnect area:         1375506.948959

Total cell area:                 81347.112600
Total area:                    1456854.061559
1
```

2.2 Timing Report

From the timing report we can see that the longest path enforces a time from the exit of a FF to the entrance of another of 15.33ns, which exceeds our constraint. It seems like the longest critical path goes from the decoder control register to the data out register. In order to make it shorter we would need to rethink our implementation of the decoder. Maybe save the matrices in-memory?

```
Information: Updating design information... (UID-85)

****************************************
Report : timing
         -path full
         -delay max
         -max_paths 1
Design : ECC_ENC_DEC
Version: O-2018.06-SP4
Date   : Thu Dec 30 20:16:37 2021
****************************************

Operating Conditions: slow   Library: slow
Wire Load Model Mode: top

  Startpoint: u_enc_dec_rgf/ctrl_reg[0]
              (rising edge-triggered flip-flop clocked by clk)
  Endpoint: dec_data_out_reg_reg[1]
            (rising edge-triggered flip-flop clocked by clk)
  Path Group: clk
  Path Type: max

  Des/Clust/Port     Wire Load Model       Library
  ------------------------------------------------
  ECC_ENC_DEC        tsmc18_wl50           slow

  --------------------------------------------------------------------------
  data required time                                                 15.33
  data arrival time                                                 -15.33
  --------------------------------------------------------------------------
  slack (MET)                                                        0.00
```

2.3 Constraints Violations

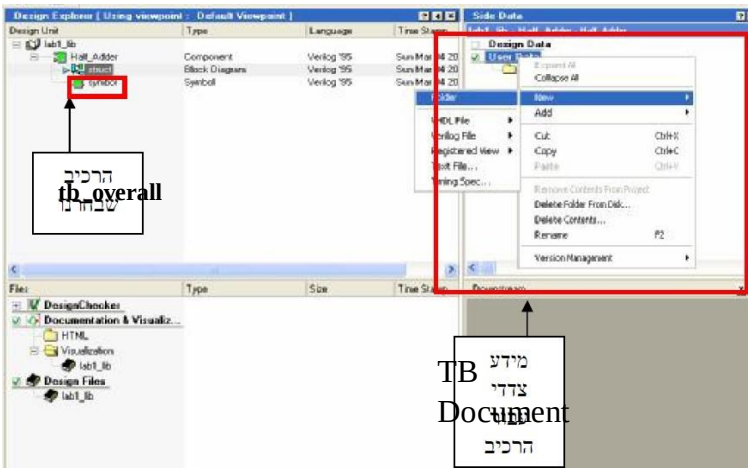
Our design has no constraint violations ☺

```
1
2  ****************************************
3  Report : constraint
4           -all_violators
5  Design : ECC_ENC_DEC
6  Version: O-2018.06-SP4
7  Date   : Thu Dec 30 20:16:37 2021
8  ****************************************
9
10 This design has no violated constraints.
11
12 1
13
```

3. SUBMISSION REQUIREMENTS

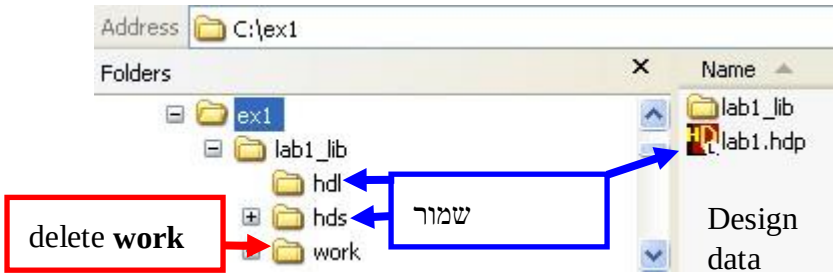
For the second (Verification) part of the project you must submit the following:

1) Gate Level Files that contain the following units:
   - Netlist – File name: top.v
   - SDF – File name: top.sdf
2) **Test Bench Files** that containing the following units: Stimulus ,Interface , Golden-Model, Overall TB and any other file that you added to your verification testbench.
3) **HDL Designer library including project files**, hdl and hds libraries. Top level module should be named **ecc_enc_dec**, and the file should be named **ecc_enc_dec.v**. Verilog 2005 syntax (or systemverilog)
4) **Short Synthesis Guide** containing the requirement described and shown in previous sections (in pdf format).
5) **Synthesis Document** should be attached in the **ecc_enc_dec (Gate Level)** design in HDL designer using side data library



6) **Delete** the **work** library

7) Compress **Project file**, **hds** and **hdl** libraries into a ZIP file named ***<ID1>_<ID2>.zip***, where ID1 and ID2 are the ID (*teudat zeut*) numbers of the submitters.



8) All the library files should be zipped to a file named **id1_id2.zip** and submitted to Moodle.

9) One of the Student's Pair Should Submit Their work to Moodle.

SUBMISSION DATE: 30/12/2021 AT 23:55

4. PROJECT GRADE EVALUATION

This part is **30 points** **from the project grade.** Preliminary checks must pass before your work is checked. **Failure in the preliminary checks means grade of 0 for this part of the project.**

Task	Description
Synthesis passes successfully.	Your design is synthesized successfully with Design Compiler. Reports are created. No timing violations. Netlist (Gate level) is generated.

Table 3: Preliminary tests

After passing the preliminary tests the system's performance will be checked with additional tests.

grade	Task	Description
30%	Implementation	Gate level simulation reaches the same logic results as RTL simulation, comparison with golden model
30%	Document	Documents contain all requirement and reports, **easy to read and proper diagrams**, right explanations, and answers.
40%	Reports & Results	Timing analysis, Power analysis, Area analysis .

Table 4: Grading policy

Late submission will degrade the project score by 5*days