Group #:
G

Design Project 2

ENSC 350 1257

Project Submitted in Partial Fulfillment of the
Requirements for Ensc 350
Towards a Bachelor Degree in Engineering Science.

Last Name:	Last Name:	Last Name:
SID:	SID:	SID:

Intentionally left blank

Table of Contents:

Design Project 2

Main Objectives:

1) To create additional design candidates for a 64-bit adder.
2) To update the procedure to include both functional and timing verification.
3) To verify both functional behaviour and timing performance of all design candidates.
4) To analyse all design candidates, comparing relative cost and timing performance.
5) To find optimal solutions from the results of earlier design candidates.
6) To document all activities performed in accomplishing the project tasks.

Sub-Objectives:

1) To setup an organised procedure for Synthesis & Functional & Timing Verification.

2) To create additional design candidates for the DUT, employing additional advanced design principles.

3) To create one set of test vectors for verifying all DUT implementations with reasonable coverage, and including estimated worst timing cases.

4) To generate timing models from Quartus.

5) To employ VHDL configurations and VHDL components to maintain design hierarchy.

6) To create ONE testbench used for both functional and timing verification of all DUT implementations.

7) To create individual scripts to make the design flow efficient and easily reproduce results.

8) To synthesis, realise and optimize relative cost and performance for all design candidates on both a Cyclone IV and a Arria II FPGAs.

9) To improve your ability to maintain engineering notebooks and fully log a record of R&D activity.

10) To improve your ability to write progress reports that document R&D activity and technical reports that document actual design details.

Specifications - Device Under Test: (DUT)

The Adder Entity:

The Adder should

- be a purely combinational circuit.
- VHDL code should describe an N-bit adder but may assume that N is a power of 2.
- The circuit should include both incoming and outgoing carries.
- The circuit should include an overflow output for the case of signed operands.

```
Entity Adder is
Generic ( N : natural := 64 );
Port ( A, B : in std_logic_vector( N-1 downto 0 );
              S : out std_logic_vector( N-1 downto 0 );
           Cin : in std_logic;
  Cout, Ovfl : out std_logic );
End Entity Adder;
```

Specifications – Verification:

Testbench Specification:

The Testbench should cycle through sufficiently many test vectors,

- Test vectors may be created using external tools and stored in a text file,
- The test vectors in the file should be formatted with one line for each measurement; with values in the order { A, B, Cin, S, Cout, Ovfl }
  - ➢ A B & S are 64-bits in Hexadecimal, all other signals are in binary.

Example:

```
0AAAAAAAAAAAAAAA 0555555555555555 1 1000000000000000 0 0
```

For functional verification, test vectors should be selected to ensure correct operation of each output bit individually for both a rising event and also a falling event.

For Timing verification, test vectors should be selected to cover sufficiently many critical paths of each design candidate.

For each test vector, the testbench should report single strings to the output that should be captured into a transcript file.

Each string should contain the measurement index, the input values, the output results and additional information about the status of the measurement.

General Tasks:

1) **Design Project Setup**:

Refer to SW-Install-Setup-350-xxxx.pdf

- Prepare your filing structure. (§Filing Structure)
- Prepare ModelSim project. (§Setting up ModelSim) – DP2.mpf
- Prepare a Quartus Project. (§Setting up Quartus) – DP2.qpf

Refer to Project Documentation Guidelines-350-xxxx.pdf

- Prepare the Activity Log file.(§Activity Logs)
- Prepare an Engineering Notebook.(§Engineering Notebooks)

2) **Baseline Device**: (using a Cyclone IV FPGA, - EP4C-E-115-F29-C7)

Create a VHDL description of the baseline device. (same as for DP1)

The Baseline Device is defined as
a 64-bit structural ripple adder implemented on a Cyclone IV FPGA (EP4C-E-115-F29-C7).

Update the log file and your notebook.

General Tasks: Design Candidates

3a) Design Candidates - topological considerations:

- In addition to the two topologies of DP1, write at least three more VHDL architectures, each exploiting different design principles.
  - Check that your code compiles with both Quartus and ModelSim/Questa.
  - Check that your code can be loaded by ModelSim/Questa and synthesised by Quartus.
  - Add comprehensive comments to the sourcecode.
- Establish a naming convention to easily identify the VHDL architectures.
- Create configurations to manage a hierarchy for bindings of Entity/Architecture to components.

3a) Design Candidates - implementation considerations:

- Each design candidate uses a specific adder topology and is implemented on a specific FPGA device. You should have at least ten design candidates; the baseline device and nine additional design candidates.
- Use Quartus to Synthesise an implementation for each topology using

  1) a Cyclone IV FPGA and

  2) an ARRIA II FPGA.

  You should select a version that fits your design and avoids inclusion of unnecessary additional hardware resources. You may wish to consider (as a weak recommendation) (EP4C-E-115-F29-C7, and EP2A-GX-45-D-F29-C6)
- Update your naming convention to allow identifiers that distinguish between individual design candidates; namely that identify the VHDL architectures and the target FPGA implementation.
- For each design candidate, rename the timing models produced by Quartus. Store these files logically in your project file hierarchy.
- Create individual configurations and scripts to distinguish between functional and timing VHDL models. The naming convention will apply to Your VHDL sourcecode filenames, Your VHDL architecture names, timing-model filenames, Configuration names, script filenames. (and possibly also timing model architecture names)

Update the log file and your notebook.

General Tasks: Testbench Design

3b) **Testbench for the DUT**:
refer to LWS03.x-350-xxxx.pdf, LWS04.x-350-xxxx.pdf, and LWS05.x-350-xxxx.pdf

➢ Write a single testbench for the Adder based on the testbenches of DP1. The testbench should
- ✓ Read test vectors from a file.
- ✓ Perform all measurements until the last test vector
- ✓ Provide information describing the results of individual measurements.
- ✓ Provide a message indicating that the last test vector has been processed.

➢ The test vector file should be formatted as given in the specifications.

➢ Update the test vectors from DP1 to include worst timing cases for each design candidate. Consider the critical timing paths associated with the topology and also the critical timing paths that arise once the design is fitted on a specific FPGA.

➢ Include an index that keeps track of the measurements. (LWS03.x-350-xxxx.pdf)
Each measurement should
- ✓ begin with the stimuli ‘X’ for all input bits, held for a PreStimTime,
- ✓ apply stimuli until all outputs are stable,
- ✓ measure the two propagation delays; one at the sum output, S, & one at Status formed by combining both Cout and Ovfl.
- ✓ verify that the result is correct,
- ✓ report a message containing useful information. (index, true values, actual values)

➢ The testbench may easily find the wort-case propagation delay of all measurements. This value should be reported to the transcript once all measurements have finished.

➢ The DUT should be instantiated as a component. You should create many separate configurations;
- ✓ for each design candidate,
- ✓ for functional verification and timing verification.
- ✓ and possibly for simulation and for synthesis.

➢ Include useful information in a well formatted wave window. Index, propagation delays etc.

➢ Add extensive comments to the testbench so that anyone can understand the code in the future.

Update the log file and your notebook.

General Tasks: Iterative Experiments

4) Synthesis & Simulation of the DUT:

- Synthesize all design candidates.
  - ✓ Record all information from Quartus, that relates to the cost of the implemented device.
  - ✓ Rename and store the timing models.
  - ✓ Review RTL and post-fit netlists to ensure that your VHDL coding methods have successfully produced your desired topological structures.

- Using a ModelSim/Questa project, run both, a functional simulation and a timing simulation for each design candidate.
  - ✓ Setup a well formatted wave window and save the script.
  - ✓ You may wish to add specific signals that are unique to a given topology for diagnosics.

- Create multiple scripts; each for design candidate, both a functional simulation and also a timing simulation. Prepend the prefixes "**FS**" and "**TS**" to the script filenames, to distinguish between the two types of simulation runs.

Each script should,

- ✓ name a unique transcript file,
- ✓ turn on/off re-direction to this transcript file when you wish to capture the output,
- ✓ compile all relevant sourcecode, including the configuration,
- ✓ start a simulation of the appropriate configuration,
- ✓ setup an appropriate wave window, and
- ✓ run the simulation.

Update the log file and your notebook.

General Tasks: Analysis of Results

5) Analysis of Cost & Performance:

- Repeat the analysis of cost from DP1 to include the new design candidates.
  - ✓ Update the summary table of costs.

- Using the measured delays of the Baseline device, estimate a bound on the propagation delay the LUTs in both a Cyclone IV and an ARRIA II.

- Analyse each candidate an make a manual estimate of the worst-case delays.
  - ✓ record the predicted & actual measured delays.

- Create a concise table of performance to easily compare, predicted results, measured results and derived parameters for each design candidate.

- If your experimental process is implemented efficiently, it should not be too difficult to repeat all measurements and collect (timing & cost) results for different values of N for each design candidate. Worst-Case Test Vectors would need to be re-constructed. If test vector filenames include the vector size, the single testbench may be modified to use a generic parameter to construct a test vector filename and read the appropriate test vector file.
  - It may be the case that the optimal cost topology depends on the value of N.
  - It may also be that the optimal performance topology depends on the value of N.
  - Record the cost and performance for various circuit sizes, in a convenient table.
  - You may also wish to plot a graph to find break-point values of N.

- Using your results, devise some simple experiments that combine topologies to find a design candidate that optimizes the cost-performance ratio. Create final summary table that conveniently provides cost, performance and cost-performance ratio.

- You will need to document all of your design attempts, including design candidates that were not optimal.

Update the log file and your notebook.

5% BONUS
for DP2
but required for FP

Design Project 2

Deliverables: Files to be Submitted for Assessment

DP2.0: Organisation and Report Structures

Begin by reading Project Documentation Guidelines-350-xxxx.pdf for the general rules for project organisation and documentation.

Your project documentation will contain many files. These files are to be zipped into a single archive called "DP2-Gxx-350-1257.zip"

- The Archive file should contain **ONE ROOT FOLDER** named **DP1**.
- Each group is to submit one archive file.
- The root folder, **DP2** in the archive must be organised with the specified filing Structure.
  - ✓ for DP2.0, most folders will be empty. (include the outline PDFs & Project Log Files)

Refer to the section (Project Documentation Guidelines-350-xxxx.pdf§Filing Structure)

This project requires **two** separate written documents;

- A summary report, being a business style letter that documents your R&D activities, and
- A design project report, that documents technical details specific to the project.

Refer to the section (Project Documentation Guidelines-350-xxxx.pdf§Reports)

- ✓ DP2.0 requires that you submit two document outlines.

1) Submit an archive file
   - ✓ A root project directory with all sub-directories forming the specified hierarchy
   - ✓ All directories should be **empty** except for the Documentation directory.

2) Submit an outline of the Summary Report, placed in the Documentation directory.
   - ✓ Filename = "DP2-Summary-Gxx-350-1257.pdf"
   - ✓ No Cover Page.
   - ✓ No Table of Contents.
   - ✓ At most two pages, containing content rectangles.(§Report Outlines)

3) Submit an outline of the Project Report, placed in the Documentation directory.
   - ✓ Filename = "DP2-Report-Gxx-350-1257.pdf"
   - ✓ Cover Page. – Given at the beginning of this document.
   - ✓ Table of Contents – titles but leave page numbers vacant.
   - ✓ Multiple pages containing content rectangles.(§Report Outlines)

4) The log files for each group member, placed in the Documentation directory.(§Activity Logs)
   - ✓ Completed entries with activities associated with the **DP2.0** deadline.
   - ✓ The same log file is used for DP1.0, DP1.1, DP1.2, DP2.0, DP2.1 & FP.
   - ✓ The workbook contains separate worksheets for each stage of Ensc 350 project work.

Design Project 2

DP2.1: Full Documentation Requirements

Your project documentation will contain many files. These files are to be zipped into a single archive called "DP2-Gxx-350-1257.zip"

- The Archive file should contain **ONE** root folder named **DP2**.
- Each group is to submit one archive file.
- The root folder, **DP2** in the archive must be organised with the specified filing Structure.

Refer to the section (Project Documentation Guidelines-350-xxxx.pdf§Filing Structure).

This project requires **ONE** written document;

- A design project report, that documents technical details specific to the project.

Refer to the section (Project Documentation Guidelines-350-xxxx.pdf§Reports)

- **DP2.1** requires that you submit complete final documents.

The Archive Contents:

1) Submit an archive file

- ✓ A root project directory with all sub-directories forming the specified hierarchy.
- ✓ All required files should be filed in the correct sub-directories.

The file structure is defined in "Project Documentation Guidelines-350-xxxx.pdf§Filing Structure"

- ✓ **Delete all unnecessary files**. Marks will be subtracted if you include useless files such as database files generated by Quartus and ModelSim/Questa

2) Submit an completed Project Report, placed in the Documentation directory.

- ✓ Filename = "DP2-Report-Gxx-350-1257.pdf"
- ✓ Cover Page.
- ✓ Table of Contents – with page numbers.
- ✓ Refer to "Project Documentation Guidelines-350-xxxx.pdf§A Design Project Report"
- ✓

3) Submit an completed Summary Report, placed in the Documentation directory.

- ✓ Filename = "DP2-Summary-Gxx-350-1257.pdf"
- ✓ No Cover Page.
- ✓ No Table of Contents.
- ✓ At most two pages.(§A Summary Report)

It is obvious to the reader when a report has been constructed with minimal thought and effort.
*A report that fills space with source code listings, screen captures produced from various tools, accompanied by a few redundant sentences is **not** considered acceptable for upper division university students*.

4) The log files for each group member, placed in the Documentation directory.(§Activity Logs)
- ✓ Completed entries with activities associated with the **DP2.1** deadline.
- ✓ The same log file is used for DP1.0, DP1.1, DP1.2, ~~DP2.0~~, DP2.1 & FP.
- ✓ The workbook contains separate worksheets for each stage of Ensc 350 project work.

5) Sourcecode:(§Documenting VHDL Sourcecode)
- ✓ Machine-readable Text files containing all original sourcecode, filed in the appropriate sub-directories.
- ✓ Human-readable PDF Appendices containing files sourcecode listings with syntax highlighting, filed in the same location as the project report.

6) Testvectors:
- ✓ Machine-readable Text files, containing the test vectors used to produce the results given in the report, filed in the appropriate sub-directory.

7) Execution Scripts:
- ✓ Machine-readable Text files, containing scripts that you created to produce results from ModelSim (and/or Quartus).

8) ModelSim Transcripts: (Timing simulation results)
- ✓ Human-readable Text files, containing the output from the simulation runs.
- ✓ The scripts and VHDL sourcecode should be created to generate comprehensive, well-organised transcripts.
- ✓ A human reader should be able to easily and quickly be able to find useful information.

9) Quartus Output Reports:(§Documenting Quartus Output Reports)
- ✓ A single Human-readable Text file, created from Quartus summary files.
- ✓ Merge the individual **summary files** for each design candidate into a single file.
- ✓ Insert Section Titles that identify the information relevant to each design candidate.
- ✓ Do Not include other reports generated by Quartus.

10) Quartus and ModelSim Project Files:
- ✓ Include the **.QPF**, **.QSF** and **.MPF** files containing Quartus/ModelSim configurations.
- ✓ Do not include any other superfluous settings files.

11) Document Name Summary Table:
- ✓ Filename = “DP2-FileList-Gxx-350-1257.pdf”
- ✓ A one-page PDF document containing a table that lists the mapping of Directories/Sub-Directories to Filenames.
- ✓ Add a brief description of file content.