Group #:

G

Design Project 1

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:

SFU

Ensc 350-1257
Fall 2025

Design Project 1

OBJ

OBJ

Main Objectives:

1) To create multiple design candidates for a 64-bit adder.
2) To establish a systematic procedure for formal functional verification.
3) To verify correct functional behaviour of all design candidates.
4) To synthesise all design candidates and analyse relative cost.
5) To document all activities performed in accomplishing the project tasks.

Sub-Objectives:

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

2) To create a ripple adder to be used as a baseline circuit for calibrating observed cost and performance metrics.

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

4) To create one set of test vectors for verifying all DUT implementations.

5) To create testbenches used for functional verification of all DUT implementations.

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

7) To synthesis, realise and compare relative cost for all topologies on both a Cyclone IV and a Arria II FPGAs.

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

9) 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 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(Qn?)-xxxx.pdf

- Prepare your filing structure. (§Filing Structure)
- Prepare ModelSim project. (§Setting up ModelSim) – DP1.mpf
- Prepare a Quartus Project. (§Setting up Quartus) – DP1.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)

An 64-bit Ripple adder implemented on a Cyclone IV FPGA (EP4C-E-115-F29-C7) will be used as a standard circuit representing the scale of the design, referred to as the baseline design.

- The cost, and performance of the baseline design will be used to normalize the respective metrics of all candidate implementations.

Write two VHDL descriptions for an N-bit ripple adder. Name the architectures "FastRipple" and "Baseline"

Baseline: Create a structural description that explicitly instantiates full adders, or iteratively generate the single bit carry and sum expressions, or use array operations.

FastRipple: Quartus naturally synthesizes a ripple adder from the "+" operator.

A (N+1)-bit addition can be implemented by increasing the length of the operands with a VHDL concatenation operator, "&".

- 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.

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:

- Write an additional VHDL architecture for an N-bit conditional-sum adder.
  - 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.

3a) Design Candidates - implementation considerations:

- Individual design candidates exploit an abstract concept (design principle) to derive a circuit structure (design topology). The circuit structure is then revised (mapped) to use the primitive elements of a target gate array and wiring paths are selected. (design implementation)
  - Example: carry-select is a principle that may be exploited resulting in a topology called a conditional-sum adder. This topology may them be mapped to LEs or ALMs depending on the target gate array.
  - The cost & performance depend on both the topology and the implementation)

- Each design candidate is a {topology/implementation} pair. A Design Candidate is the use of a specific adder topology that is implemented on a specific FPGA device.

- Thus, your task is to create four design candidates; the baseline device and three 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.

Update the log file and your notebook.

General Tasks: Testbench Design

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

- Write two specific testbenches; one for each design topology. Each 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.

- Use the following constants,

  Constant TestVectorFile : string := "Adder00.tvs";
  Constant PreStimTime : time := 1 ns;
  Constant PostStimTime : time := **??** ns; (*Experiment to choose a suitable value.*)

- 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,
  - ✓ verify that the result is correct,
  - ✓ report a message containing useful information. (index, true values, actual values)

- The DUT should be instantiated directly as an entity. Each testbench should be identical except for the statement that instantiates the DUT.

- 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.

- Using a ModelSim/Questa project, run a functional simulation of both topologies.
  - ✓ 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 two scripts; each for functional verification of the two topologies.

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,
- ✓ start a simulation of the appropriate testbench,
- ✓ 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:

- Analyse the Baseline device and make a manual estimate of the required FPGA resources.
  - ✓ Record the predicted & actual resources required.

- Analyse the three remaining design candidates and make an estimate of the required FPGA resources. (Ripple-Arria, CSA-Cyclone, CSA-Arria)
  - ✓ Record the predicted & actual resources required.

- Determine a reasonable method to estimate the cost of an Arria II ALM in comparison to the LE of a Cyclone IV. (Suggestion, it is easy to create and download a .csv file from Digikey containing relevant information for Cyclone and Arria devices, including the cost.)

- The Baseline device is used to normalize parameters derived from measurements on the design candidates. Create a concise table of costs to easily compare, predicted results, measured results and derived parameters for each design candidate.

Update the log file and your notebook.

Deliverables: Files to be Submitted for Assessment

DP1.0: 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 "DP1-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, **DP1** in the archive must be organised with the specified filing Structure.
  - ✓ for DP1.0, most folders will be empty.

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)

- ✓ DP1.0 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 = "DP1-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 = "DP1-Report-Gxx-350-1257.pdf"
   - ✓ Cover Page.
   - ✓ 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 **DP1.0** 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.

DP1.1: Full Documentation Requirements

Your project documentation will contain many files. These files are to be zipped into a single archive called "DP1-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, **DP1** 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 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)

- DP1.1 requires that you submit complete final documents.

The Archive Contents:

Do NOT include unnecessary files/folders.
- delete folders created by Quartus/ModelSim

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 = "DP1-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 = "DP1-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 **DP1.1** 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/Questa (and/or Quartus).

8) ModelSim/Questa Transcripts: (functional 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/Questa Project Files:

- ✓ Include the **.QPF**, **.QSF** and **.MPF** files containing Quartus/ModelSim/Questa tool configuration settings. Do not include any other superfluous settings files.

11) Document Name Summary Table:

- ✓ Filename = "DP1-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.