# ECE 385 Lab 7 Report Outline

- ❏ **Introduction**
  - ❏ Summarize the basic functionality of the NIOS-II processor running on the Cyclone IV FPGA.
- ❏ **Written Description and Diagrams of NIOS-II System**
  - ❏ Summary of Operation
    - ❏ Describe in words the hardware component of the lab (Describe what IP blocks are used beyond the ones necessary for the NIOS to run. In this lab, the only additional IP blocks used are the PIO blocks).
    - ❏ Describe in words the software component of the lab. One of the INQ questions asks about the blinker code, but you must also describe your accumulator.
  - ❏ Written Description of all .sv Modules
    - ❏ A guide on how to do this was shown in the Lab 5 report outline. Do not forget to describe the Qsys generated file lab7_soc.v!
  - ❏ Top Level Block Diagram
    - ❏ This diagram should represent the placement of all your modules in the top level. *Please only include the top level diagram and not the RTL view of every module*. The Qsys view of the NIOS processor is not necessary for this portion.
- ❏ **Answers to all 11 INQ Questions**
  - ❏ The Prelab question (part A) is one of these 11 questions. The questions are reproduced at the end of this document for convenience.
- ❏ **Postlab Question**
  - ❏ Document the Design Resources and Statistics in following table.

| | |
|---|---|
| LUT | |
| DSP | |
| Memory (BRAM) | |
| Flip-Flop | |
| Frequency | |
| Static Power | |
| Dynamic Power | |
| Total Power | |

- ❑ **Conclusion**
  - ❑ Discuss functionality of your design. If parts of your design didn’t work, discuss what could be done to fix it
  - ❑ Was there anything ambiguous, incorrect, or unnecessarily difficult in the lab manual or given materials which can be improved for next semester?  You can also specify what we did right so it doesn’t get changed.

# APPENDIX

**INQ Questions**

1. What advantage might on-chip memory have for program execution?

2. Note the bus connections coming from the NIOS II; is it a Von Neumann, “pure Harvard”, or “modified Harvard” machine and why?

3. Note that while the on-chip memory needs access to both the data and program bus, the led peripheral only needs access to the data bus. Why might this be the case?

4. Why does SDRAM require constant refreshing?

5. Make sure this is consistent with your above numbers; you will need to justify how you came up with 1 Gbit to your TA.

| SDRAM Parameter | Short Name | Parameter Value |
|---|---|---|
| Data Width | [width] | |
| # of Rows | [nrows] | |
| # of Columns | [ncols] | |
| # of Chip Selects | [ncs] | |
| # of Banks | [nbanks] | |

6. What is the maximum theoretical transfer rate to the SDRAM according to the timings given?

7. The SDRAM also cannot be run too slowly (below 50 MHz). Why might this be the case?

8. Make another output by clicking clk c1, and verify it has the same settings, except that the phase shift should be -3ns. This puts the clock going out to the SDRAM chip (clk c1) 3ns ahead of the controller clock (clk c0). Why do we need to do this? Hint, check Altera Embedded Peripheral IP datasheet under SDRAM controller.

9. What address does the NIOS II start execution from? Why do we do this step after assigning the addresses?

10. You must be able to explain what each line of this (very short) program does to your TA. Specifically, you must be able to explain what the volatile keyword does (line 8), and how the set and clear functions work by working out an example on paper (lines 13 and 16). *This question is referring to the blinker code.*

11. Look at the various segment (.bss, .heap, .rodata, .rwdata, .stack, .text), what does each section mean? Give an example of C code which places data into each segment, e.g. the code: **const int my_constant[4] = {1, 2, 3, 4}** will place 1, 2, 3, 4 into the .rodata segment.