ENGR 210 / CSCI B441

Introduction

Andrew Lukefahr

Introduction

- Topics covered:
  - Boolean algebra and logic gates
  - Sequential Logic
  - State Machines
  - Serial Communication
    - Buses
    - Protocols

Prof. Lukefahr

Andrew Lukefahr, Assistant Professor

Office: 2032 Luddy Hall

Email: *lukefahr@iu.edu*

Research work on security for FPGA-based systems.

Submit Your Own Work

- All submitted work must be your own
  - Not your buddy's
  - Not last semester's
  - Not the internet's

- I've done this before. I will catch you.

Logic Gates

Review

- Ask a series of (hopefully) review questions.
- If you have never seen this before, that’s ok, but let me know
- I am happy to help review after class / in office hours.

Review Questions

- What is 0x42 in binary? In decimal?
- What is -5 in 8-bit binary?
- Can you draw an AND gate? OR? NOT? NAND? NOR? XOR?
- Can you draw this circuit: $D = A \cdot B + C$?
- What is the truth table for this: $D = A \cdot B + C$?

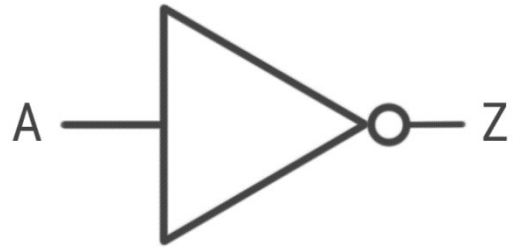
Review: Numbers

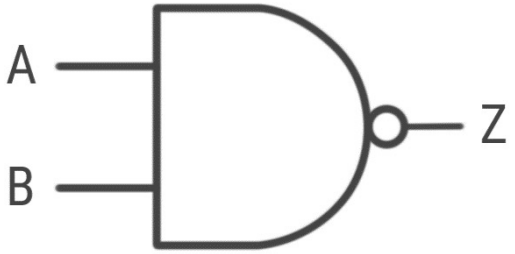
- What is 0x42 in binary? In decimal?

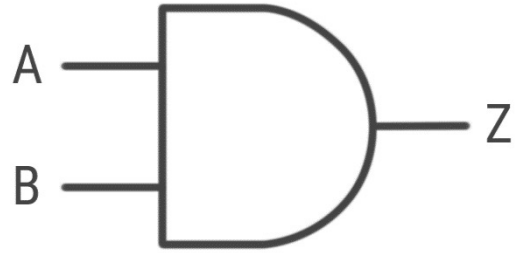
Review: 2’s Compliment

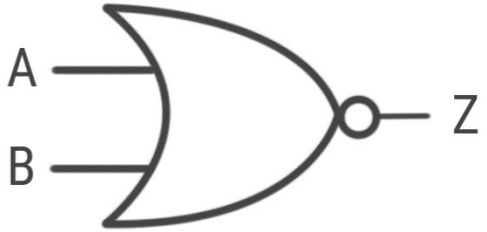
- What is -5 in binary?

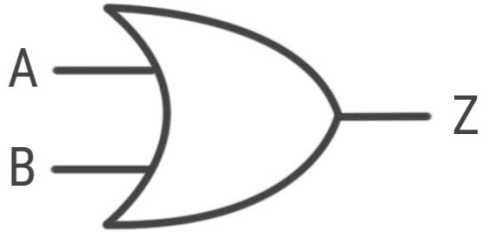
Review: Logic Gates

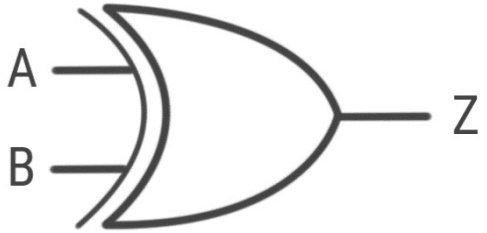












Review: Boolean Equation

- What circuit is this?

$$D = A \cdot B + C$$

Review: Truth Table

- What is the truth table for this?

$$D = A \cdot B + C$$

NOT

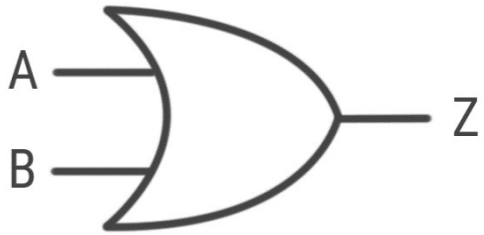
- Math:
  - $Z = \bar{A}$
- `Code:`
  - `Z = ~A`
- *Schematic*

A	Z

- Math:
  - $Z = \bar{A}$
- `Code:`
  - `Z = ~A`
- *Schematic*

OR

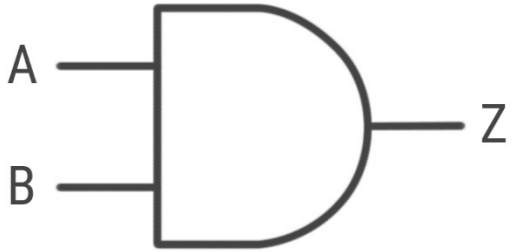
- Math:
  - $Z = A + B$
- `Code:`
  - `Z = A | B`
- *Schematic*



A	B	Z

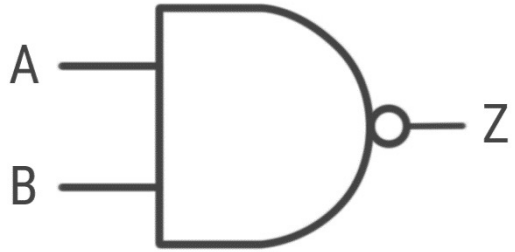
AND

- Math:
  - $Z = A \cdot B$
- `Code:`
  - `Z = A & B`
- *Schematic*



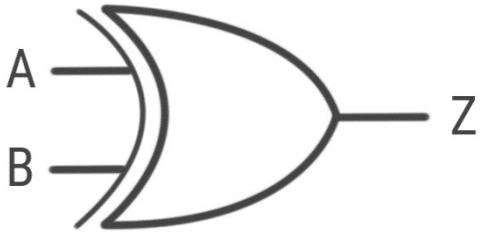
A	B	Z

Other Gates: NAND



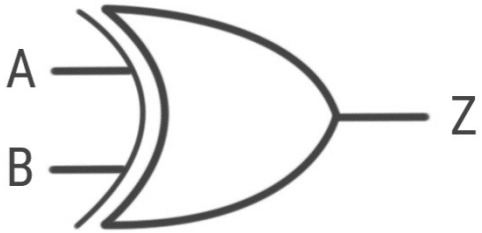
A	B	Z

Other Gates: XOR



A	B	Z

Other Gates: XOR



A	B	Z

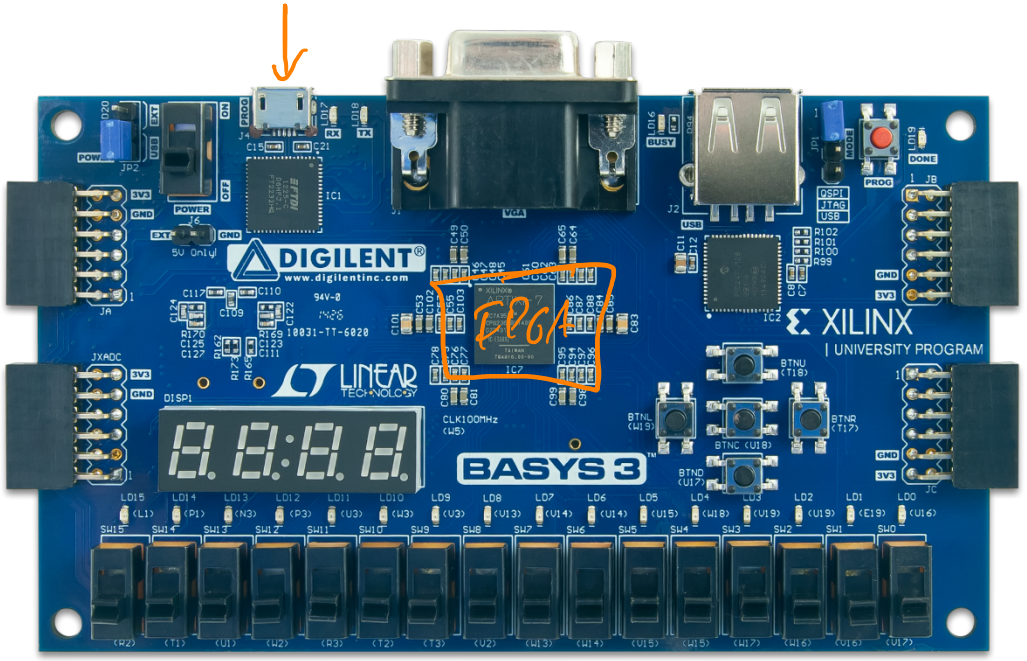
$Z = A \wedge B$

↑

shift + 6

$Z = A \oplus B$

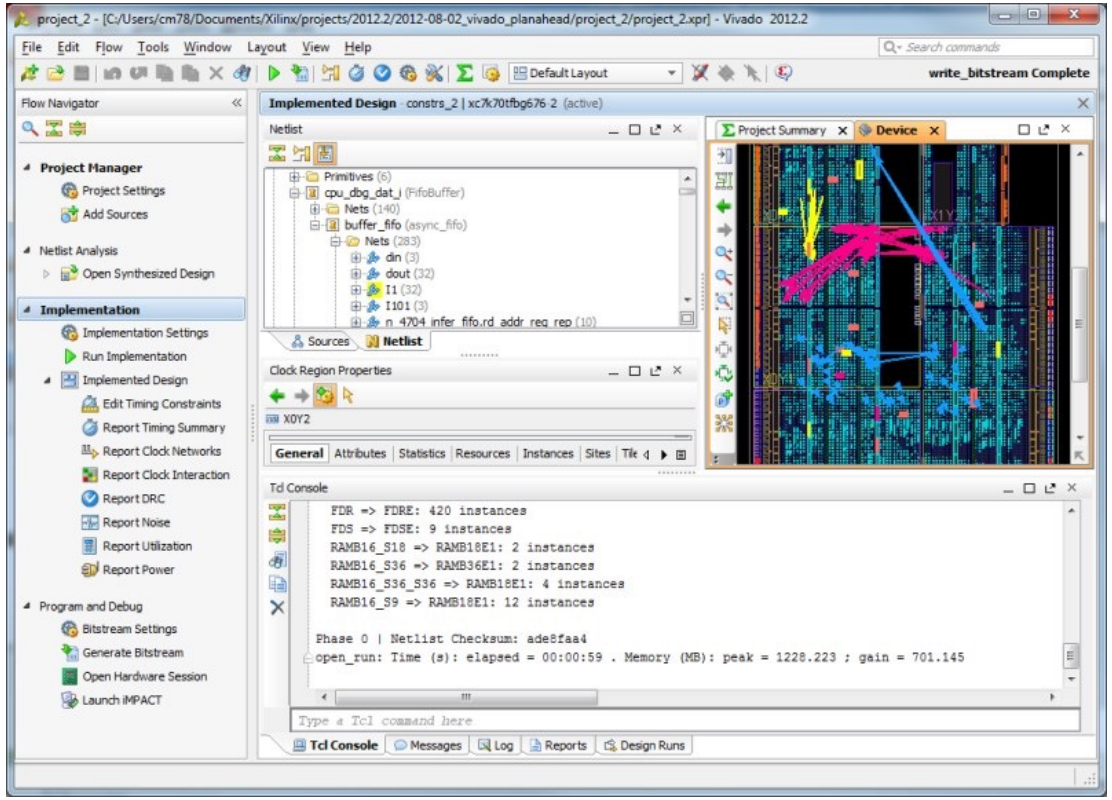
Basys3 Boards



- Checkout board for semester next time
- Programmed with Xilinx Vivado
  - Available in Luddy 4111 or download yourself

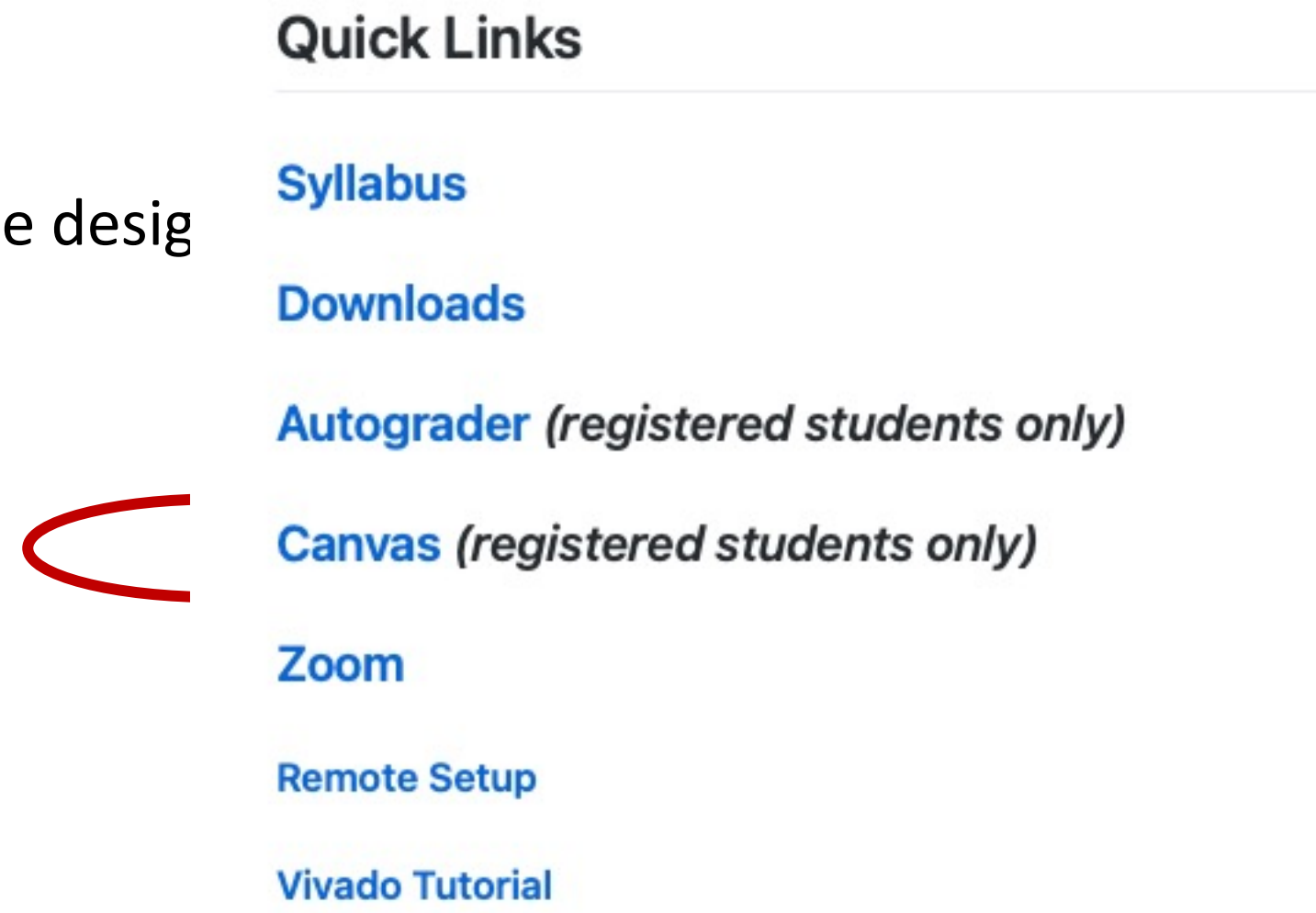
Xilinx Vivado Design Suite

- Used to map Verilog code to an FPGA
- Professional tool with multiple steps

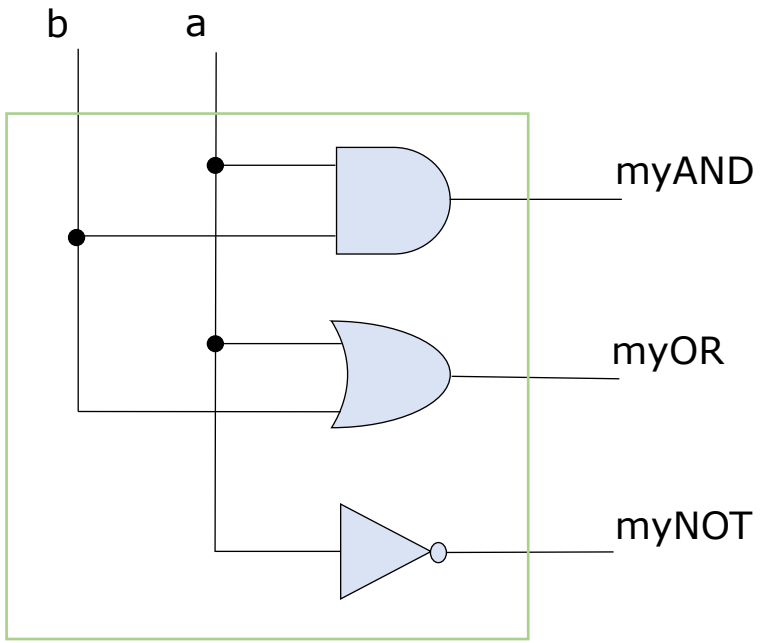


Vivado Tutorial

- This is a ‘tutorial’ lab.
- It is to provide you a reference desig
- It is NOT DUE!



Tutorial *Schematic*



Verilog Logic Operators

Verilog logic operators:

AND:	`&`
OR:	`|`
NOT:	`~`

Example:

```
assign myAND = a & b;
```

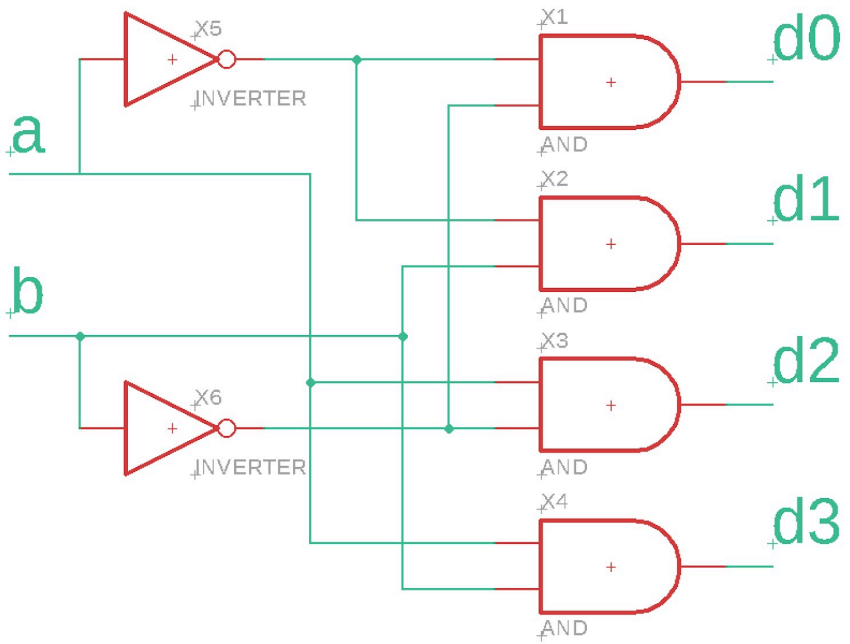
The constraint file should create the following mapping of input and outputs to the switches and LEDs on the Basys3 board:

Signal	Basys3 input	Signal	Basys3 output
a	sw0	myAND	led0
b	sw1	myNOT	led1
		myOR	led2

Verilog Project 1: Demultiplexer

- Create a 3-to-8 demultiplexer in Vivado.

- Here's a 2-to-4 demultiplexer example.



Deliverable #1: Autograder

- Log on to the autograder:

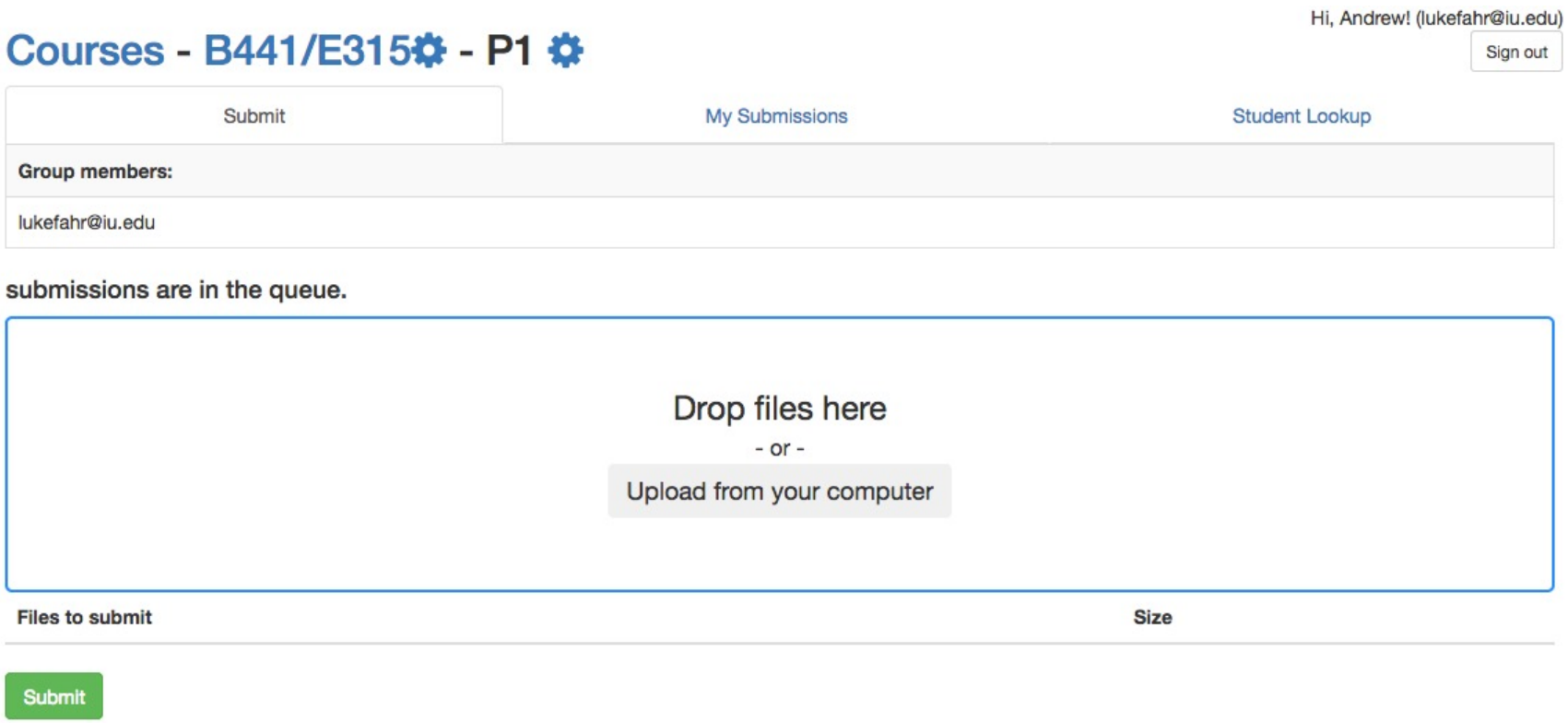
  http://autograder.sice.indiana.edu

- Log on with your ‘username@iu.edu’ account

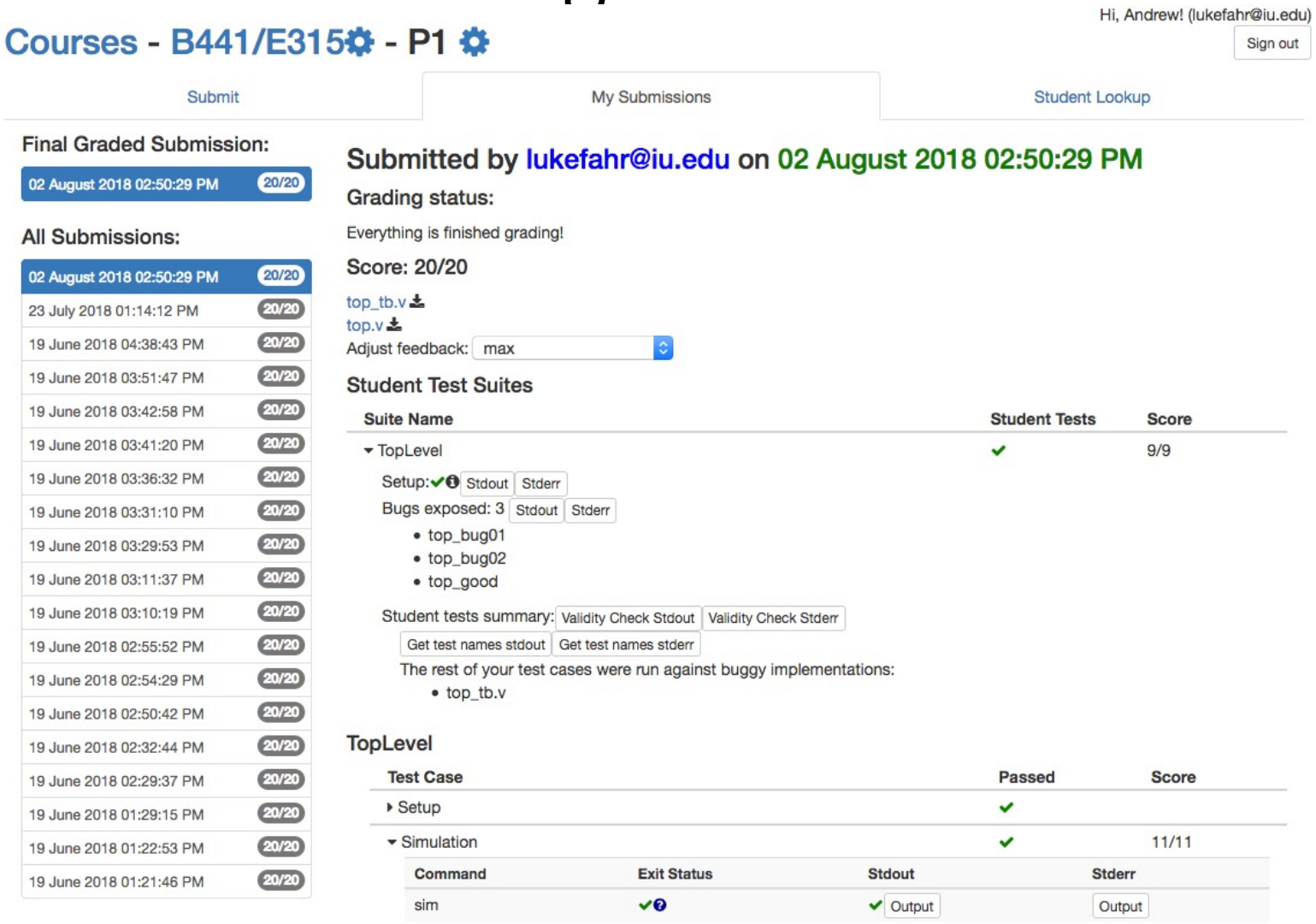
- Select ‘Engr210.s22’ -> ‘P0’

- Upload:
  - `top.v` : This is your top-level Verilog source file
  - `top_tb.v`: This is your top-level Verilog **T**est**B**ench

Deliverable #1: Autograder



Deliverable #1: Autograder



Deliverable #2: FPGA Demo

- Use Vivado to:
  - Synthesize your design
  - Program the FPGA

- Verify to yourself it works. (no points ☹)
- Do a demo for a TA. (the points ☺)

Why 2 Deliverables?

- Encourage testing
  - We give you points for good testbenches
- Check correctness
  - Automatically checks for bugs
- Reduce your debug time
  - Synthesis is slow. Don't until you are 100% sure your code works.

Next Time

- Truth Tables