CSE 390 B Spring 2021

Boolean Logic & Project 1 Overview

Fundamentals of Circuits, Hardware Simulation, Skills Inventory, Project 1 Demo

Agenda

- Let's Get Organized!
  - What's included in your at-home study area?
- Reading Overview and Q&A
- Hardware Design Language
- Project 1
  - Overview & Example: Xor
  - Demo
  - Group work

Agenda

- **Let's Get Organized!**
  - **What's included in your at-home study area?**
- Reading Overview and Q&A
- Hardware Design Language
- Project 1
  - Overview & Example: Xor
  - Demo
  - Group work

Breakout groups

In your small groups…

Talk about what your study area has looked like in previous quarters.

- What does your environment look like?
- What might be/has been a challenge in setting up your ideal at home study environment?
- What could be some tools, resources, and/or vibes (that are within your control) that could help mitigate these challenges?

BREAKOUTS!

Agenda

- Let's Get Organized!
  - What's included in your at-home study area?
- **Reading Overview and Q&A**
- Hardware Design Language
- Project 1
  - Overview & Example: Xor
  - Demo
  - Group work

Boolean Values

- A binary choice: true or false
- Maps to “high” signal (true) or “low” signal (false

“Off”

False

0

“On”

True

1

Boolean Operations

- Can use simple logical operations to combine booleans
  - **Truth table**: Writing out *every possible* set of inputs and the corresponding output of the operation
  - Operations correspond to physical hardware gates
- Examples:
  - x And y
  - x Or y
  - Not x

x	*y*	And
0	0	0
0	1	0
1	0	0
1	1	1

x	*y*	Or
0	0	0
0	1	1
1	0	1
1	1	1

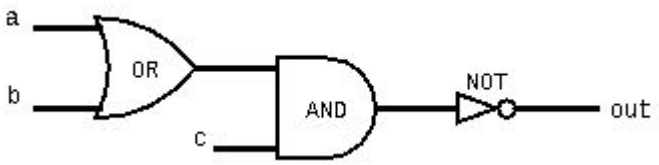
x	Not
0	1
1	0

Boolean Functions

- Combinations of boolean values/inputs
- Reading discussed 3 ways of specifying boolean functions:
  - Boolean expression: Not((a Or b) And c)
  - Truth table:

a	*b*	*c*	*out*
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

  - Circuit diagram of hardware gates:



Reading Q&A

Agenda

- Let's Get Organized!
  - What's included in your at-home study area?
- Reading Overview and Q&A
- **Hardware Design Language**
- Project 1
  - Overview & Example: Xor
  - Demo
  - Group work

Hardware Design Language (.hdl)

- A programming language to specify hardware **components** and how they're **connected**
  - Just another way of writing a boolean function!!

- There are many Hardware Design Languages in use today (e.g. VHDL, Verilog, SystemVerilog)
  - In this course, we'll use a simple one, just called "HDL".

- Unlike Java, HDL is a **declarative** language
  - The order of statements doesn't matter
  - Describes a physical system

Hardware Design Language (.hdl)

- Makeup of an HDL file:
  - File comment describes expected behavior
  - "IN" names chip inputs, "OUT" names chip outputs
  - "PARTS" is where you specify the components that implement the chip
    - For most of project 1, this will be specifying boolean functions!

```
/**
* And gate:
* out = 1 only if both a and b are 1
*/
CHIP And {
  IN a, b;
  OUT out;

  PARTS:
  // Put your code here:
}
```

Reusing Components

- Once you've completed a chip implementation, you can use that chip to implement future chips!

- We only give you one gate, "Nand", to start out with
  - This means your entire computer will essentially be built on top of Nand gates!!!
  - More on this in a few slides

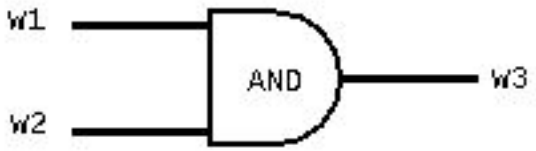
- We also give you some chips you can use without implementing
  - See project specs for more details

Reusing Components Example: And

- The chip specification tells us the name of the input and output wires!

```
CHIP And {
  IN a, b;
  OUT out;
   ...
}
```

- Goal: implement w1 And w2
  - HDL Syntax: `And(a=w1, b=w2, out=w3);`
  - Equivalent circuit diagram:



Multi-Bit Buses in HDL

- Sometimes it can be useful to manipulate groups of wires (called a “bus” of wires)

- HDL provides array like syntax for manipulating buses
  - Example - And4 chip:

```
/**
 * Bit-wise And of two 4-bit inputs
 */
CHIP And4 {
  IN a[4], b[4];
  OUT out[4];

  PARTS:
  And (a=a[0], b=b[0], out=out[0]);
  And (a=a[1], b=b[1], out=out[1]);
  And (a=a[2], b=b[2], out=out[2]);
  And (a=a[3], b=b[3], out=out[3]);
}
```

HDL Resources

- HDL is going to feel uncomfortable at first!
  - It certainly did for me :)

- Some resources for helping you navigate HDL (these are all linked under the "Resources" page on the course website
  - HDL Survival guide
  - Appendix A (HDL Spec)
  - Chip Set Overview (to help you remember the inputs/outputs for various chips)
  - Chapter readings

Agenda

- Let's Get Organized!
  - What's included in your at-home study area?
- Reading Overview and Q&A
- Hardware Design Language
- **Project 1**
  - **Overview & Example: Xor**
  - Demo
  - Group work

Project 1 Overview: It all starts with Nand

- "Nand" stands for "Negated And"
  - The And operation, but every output is negated

x	*y*	And
0	0	0
0	1	0
1	0	0
1	1	1

x	*y*	Nand
0	0	1
0	1	1
1	0	1
1	1	0

Project 1 Overview: It all starts with Nand

- Turns out every logic gate in our computer can be implemented in terms of Nand
- We will give you only Nand to start project 1
- You will build other basic gates (Not, And, etc.) in terms of Nand and then increasingly complex gates in terms of other gates you implement
- Will culminate (after a couple projects) with you building a computer entirely based on Nand gates!

How can it all be built on top of Nand?

- Remember our boolean synthesis strategy from the reading?
  - Could represent any truth table/boolean function in terms of three gates: Not, And, Or

- We can represent Or in terms of Not and And
  - De Morgan's Law! a Or b = Not(Not(a) And Not(b))
- We can represent And in terms of Not and Nand
  - a And b = Not(a Nand b)
- We can represent Not in terms of Nand
  - Not a = a Nand a

- So we can really represent any truth table/boolean function in terms of Nand!!

Project 1 Example: Xor

- Let's walkthrough an example of some of what you will be doing in project 1

- We want to implement the following gate: Xor

```
/**
* Xor gate:
* out = not(a == b)
*/
CHIP Xor {
  IN a, b;
  OUT out;

  PARTS:
  // Put your code here:
}
```

Project 1 Example: Xor

- Plan:
  - Generate truth table
  - Use truth table to generate boolean function
  - Convert boolean function to HDL syntax!

```
/**
* Xor gate:
* out = not(a == b)
*/
CHIP Xor {
  IN a, b;
  OUT out;

  PARTS:
  // Put your code here:
}
```

Project 1 Example: Xor

- First step: build a truth table!
  - Have to interpret the spec: out = not(a == b)

a	b	Xor
0	0	0
0	1	1
1	0	1
1	1	0

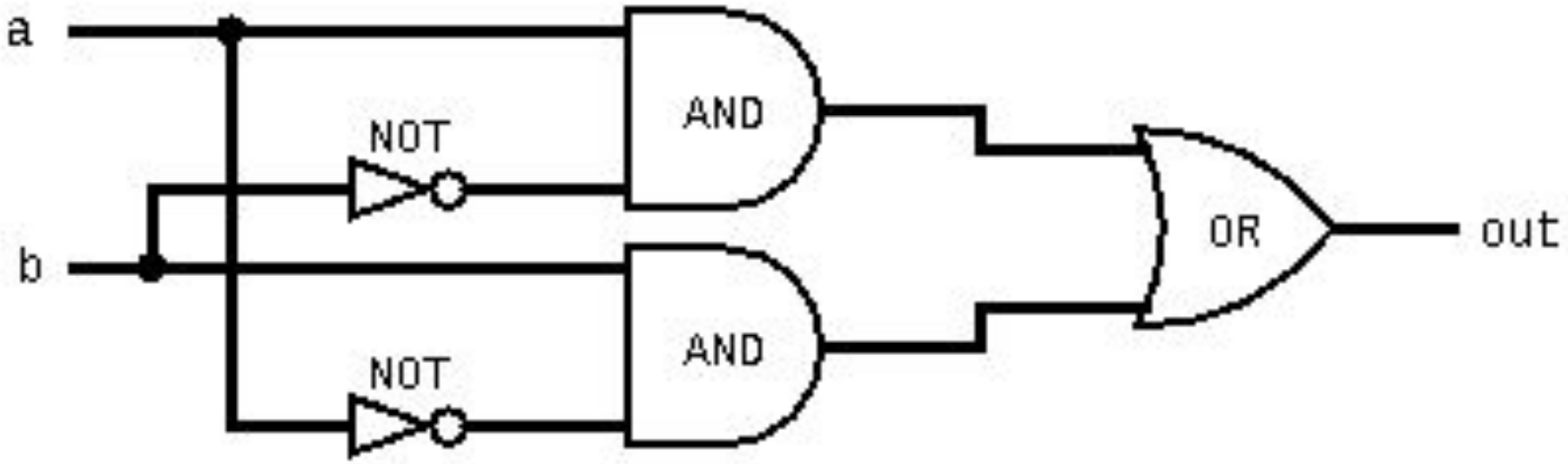
Project 1 Example: Xor

- Second step: Use truth table to generate a boolean function
- Need to use our strategy from the reading!
  - row2 = Not(a) And b
  - row3 = a And Not(b)
  - a Xor b = row2 Or row3 = (Not(a) And b) Or (a And Not(b))

a	b	Xor
0	0	0
0	1	1
1	0	1
1	1	0

Project 1 Example: Xor

- Sometimes helps to convert to a circuit diagram!
  - a Xor b = (Not(a) And b) Or (a And Not(b))

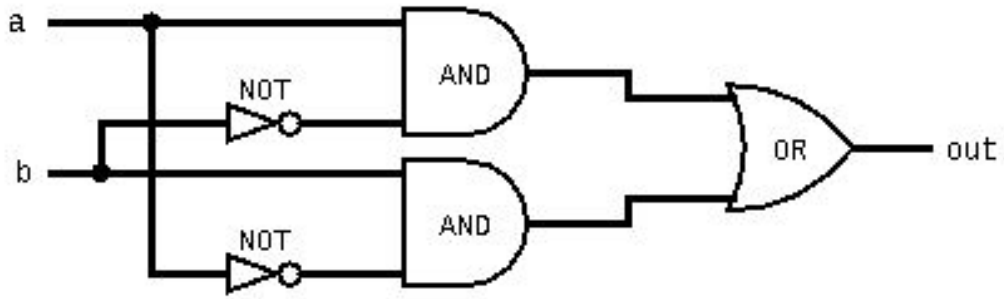


Project 1 Example: Xor

- Third step: Convert boolean function to HDL syntax
  - a Xor b = (Not(a) And b) Or (a And Not(b))
  - Assumes we've implemented Not, And, and Or already
  - Note the use of intermediary wires! (nota, notb, x, y)

```
CHIP Xor {
  IN a, b;
  OUT out;

  PARTS:
  Not(in=a, out=nota);
  Not(in=b, out=notb);
  And(a=a, b=notb, out=x);
  And(a=nota, b=b, out=y);
  Or(a=x, b=y, out=out);
}
```



Agenda

- Let's Get Organized!
  - What's included in your at-home study area?
- Reading Overview and Q&A
- Hardware Design Language
- **Project 1**
  - Overview & Example: Xor
  - **Demo**
  - Group work

Project 1

PART I: Study Skills Inventory

- Self-assessing your skill level in various study practices and habits.

PART II: Boolean Logic

- If you've cloned your repo, you have everything you need to get started on project 1!

PART III: Boolean Logic Reflection

- Reflecting on what your experience was like in working through the Boolean Logic project.

DUE NEXT THURSDAY 11:59PM

Project 1 Demo

- Brief demo of the tools you will use in project 1
- Goal for today is for all of you to at least have opened up the tools
- Ideally you'll be able to implement the first gate in project 1 - Not
  - This will get you familiar with the tools and sort out any confusion you may have!

Agenda

- Let's Get Organized!
  - What's included in your at-home study area?
- Reading Overview and Q&A
- Hardware Design Language
- **Project 1**
  - Overview & Example: Xor
  - Demo
  - **Group work**

Project 1 Group Work

- We now have some time for allocated for you to complete at least the first gate in project 1 in groups

- Complete the following tasks in groups - course staff will bounce around and are happy to answer any questions
  - Read the "Overview" section of the spec and briefly skim the rest of the spec (you'll want to do a more thorough read through of the spec later)
  - Estimate the amount of time you think it will take to finish the project
  - Verify that everyone has cloned their repo
  - Verify that everyone can open the HardwareSimulator (see instructions in the Tools section of the spec)
  - Attempt to implement and test the Not gate

Wrapping Up

What's in store for Week 2?

- Boolean Arithmetic & ALU
- Time Management
- Project 2 Released

Reminders

- Project 1 Due Thursday 4/8 11:59PM PDT
- First Student-TA 1:1 session
- TA Office Hours When2Meet