



09.10.20 23:26

CSCI 150

Introduction to Digital and Computer System Design

Lecture 3: Combinational Logic Design I



Jetic Gū
2020 Fall Semester (S3)

Overview

- Focus: Methodology
- Architecture: Combinatory Logical Circuits
- Textbook v4: Ch3 3.1; v5: Ch3 3.1
- Core Ideas:
 1. Light Reading followed by My Lunch Break

Lecture 1 & 2

- Lecture 1
 - What is digital (logic) circuit
 - How information is represented in digital (logic) circuit
- Lecture 2
 - Atomic components of digital (logic) circuit: Gates, I/O
 - Boolean Algebra

Overview

- Focus: **Methodology**
- Architecture: Combinatory Logical Circuits
- Textbook v4: Ch3 3.1; v5: Ch3 3.1
- Core Ideas:
 1. How to **systematically use** what you've learned in Lecture 2
 2. **practice practice practice**

Design Procedure

You, design this!

Know the Problem

1. Specification of the Problem
2. Basic Design, a functional Beta Implementation
3. Optimisation, Optimisation
Harder, Better, Faster, Stronger

Systematic Design Procedures (Universal)

1. **Specification:** Write a specification for the circuit
2. **Formulation:** Derive relationship between inputs and outputs of the system
e.g. using truth table or Boolean expressions
3. **Optimisation:** Apply optimisation, minimise the number of logic gates and literals required
4. **Technology Mapping:** Transform design to new diagram using available implementation technology
5. **Verification:** Verify the correctness of the final design in meeting the specifications

Systematic Design Procedures

1. **Specification:** Write a specification for the circuit
2. **Formulation:** Derive relationship between inputs and outputs of the system e.g. using truth table or Boolean expressions
3. **Optimisation:** Apply optimisation, minimise the number of logic gates and literals required
4. **Technology Mapping:** Transform design to new diagram using available implementation technology
5. **Verification:** Verify the correctness of the final design in meeting the specifications

An Old Friend

Curtain Motors Revisited

An Old Friend

1. Specification

- Curtain Motor Control

- Input • **Sensor1: 1 when curtain is fully closed**
- Input • **Sensor2: 1 when curtain is fully open**
- Input • **Button1: 1 when user wants to open the curtain**
- Input • **Button2: 1 when user wants to close the curtain**
- Output • **Output1: 1 to make the motor open the curtain**
- Output • **Output2: 1 to make the motor close the curtain**
- Output • **Light: motor is active**



- Stop the motor when the curtain is already fully opened/closed

An Old Friend

1. Specification

- Curtain Motor Control

- Switch • **Sensor1: 1 when curtain is fully closed**
- Switch • **Sensor2: 1 when curtain is fully open**
- Switch • **Button1: 1 when user wants to open the curtain**
- Switch • **Button2: 1 when user wants to close the curtain**
- Prob • **Output1: 1 to make the motor open the curtain**
- Prob • **Output2: 1 to make the motor close the curtain**
- Prob • **Light: motor is active**



- Cond Stop the motor when the curtain is already fully opened/closed

An Old Friend

2. Formulation

- Curtain Motor Control

Switch

- Sensor1: 1 when curtain is fully closed

Switch

- Sensor2: 1 when curtain is fully open

Switch

- Button1: 1 when user wants to **open** the curtain

Switch

- Button2: 1 when user wants to **close** the curtain

Prob

- Output1: 1 to make the motor **open** the curtain

Prob

- Output2: 1 to make the motor **close** the curtain

Prob

- Light: motor is active

FullyClosed = Sensor1

FullyOpened = Sensor2

Cond

Stop the motor when the curtain is already fully opened/
closed

Demo

An Old Friend

2. Formulation

- Curtain Motor Control

- **Sensor1: 1 when curtain is fully closed**
- **Sensor2: 1 when curtain is fully open**
- Button1: 1 when user wants to **open** the curtain
- Button2: 1 when user wants to **close** the curtain
- Output1: 1 to make the motor **open** the curtain
- Output2: 1 to make the motor **close** the curtain
- Light: motor is active

FullyClosed = Sensor1
FullyOpened = Sensor2
Light = Out1 + Out2

Cond Stop the motor when the curtain is already fully opened/closed

An Old Friend

2. Formulation

- Curtain Motor Control

- Switch • **Sensor1: 1 when curtain is fully closed**
- Switch • **Sensor2: 1 when curtain is fully open**
- Switch • Button1: 1 when user wants to **open** the curtain
- Switch • Button2: 1 when user wants to **close** the curtain
- Prob • Output1: 1 to make the motor **open** the curtain
- Prob • Output2: 1 to make the motor **close** the curtain
- Prob • Light: motor is active

$$\begin{aligned}\text{FullyClosed} &= \text{Sensor1} \\ \text{FullyOpened} &= \text{Sensor2} \\ \text{Light} &= \text{Out1} + \text{Out2}\end{aligned}$$

$$\text{Out1} = \text{Bu1} \cdot \overline{\text{FullyOpened}}$$

$$\text{Out2} = \text{Bu2} \cdot \overline{\text{FullyClosed}}$$

Cond Stop the motor when the curtain is already fully opened/closed

An Old Friend

2. Formulation

- Curtain Motor Control

- Switch** • **Sensor1: 1 when curtain is fully closed**
- Switch** • **Sensor2: 1 when curtain is fully open**
- Switch** • **Button1: 1 when user wants to **open** the curtain**
- Switch** • **Button2: 1 when user wants to **close** the curtain**
- Prob** • **Output1: 1 to make the motor **open** the curtain**
- Prob** • **Output2: 1 to make the motor **close** the curtain**
- Prob** • **Light: motor is active**

$$\text{FullyClosed} = \text{Sensor1}$$
$$\text{FullyOpened} = \text{Sensor2}$$
$$\text{Light} = \text{Out1} + \text{Out2}$$

$$\text{Out1} = \text{Bu1} \cdot \overline{\text{FullyOpened}} \cdot \overline{\text{BothPressed}}$$

$$\text{Out2} = \text{Bu2} \cdot \overline{\text{FullyClosed}} \cdot \overline{\text{BothPressed}}$$

$$\text{BothPressed} = \text{Bu1} \cdot \text{Bu2}$$

Cond Stop the motor when the curtain is already fully opened/
closed

An Old Friend

3. Optimisation

$$\begin{aligned}\text{FullyClosed} &= \text{Sensor1} \\ \text{FullyOpened} &= \text{Sensor2} \\ \text{Light} &= \text{Out1} + \text{Out2} \\ \text{Out1} &= \text{Bu1} \cdot \overline{\text{FullyOpened}} \cdot \overline{\text{BothPressed}} \\ \text{Out2} &= \text{Bu2} \cdot \overline{\text{FullyClosed}} \cdot \overline{\text{BothPressed}} \\ \text{BothPressed} &= \text{Bu1} \cdot \text{Bu2}\end{aligned}$$

$$\begin{aligned}\text{Light} &= \text{Out1} + \text{Out2} \\ \text{Out1} &= \text{Bu1} \cdot \overline{\text{Sensor2}} \cdot \overline{\text{Bu1}} \cdot \overline{\text{Bu2}} \\ \text{Out2} &= \text{Bu2} \cdot \overline{\text{Sensor1}} \cdot \overline{\text{Bu1}} \cdot \overline{\text{Bu2}}\end{aligned}$$

An Old Friend

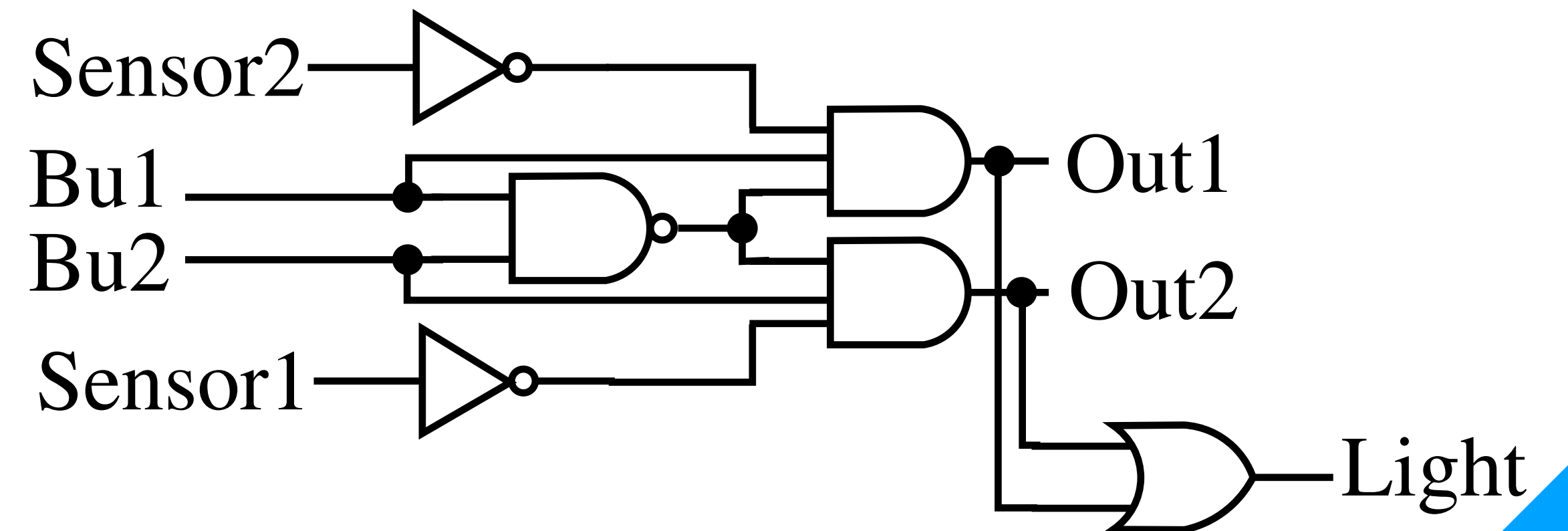
4. Technology Mapping

$$\text{Light} = \text{Out1} + \text{Out2}$$

$$\text{Out1} = \text{Bu1} \cdot \overline{\text{Sensor2}} \cdot \overline{\text{Bu1}} \cdot \text{Bu2}$$

$$\text{Out2} = \text{Bu2} \cdot \overline{\text{Sensor1}} \cdot \overline{\text{Bu1}} \cdot \text{Bu2}$$

- Available Components (Technology)
 - AND, NAND, OR, NOT gate; Switch, Prob



An Old Friend

4. Technology Mapping

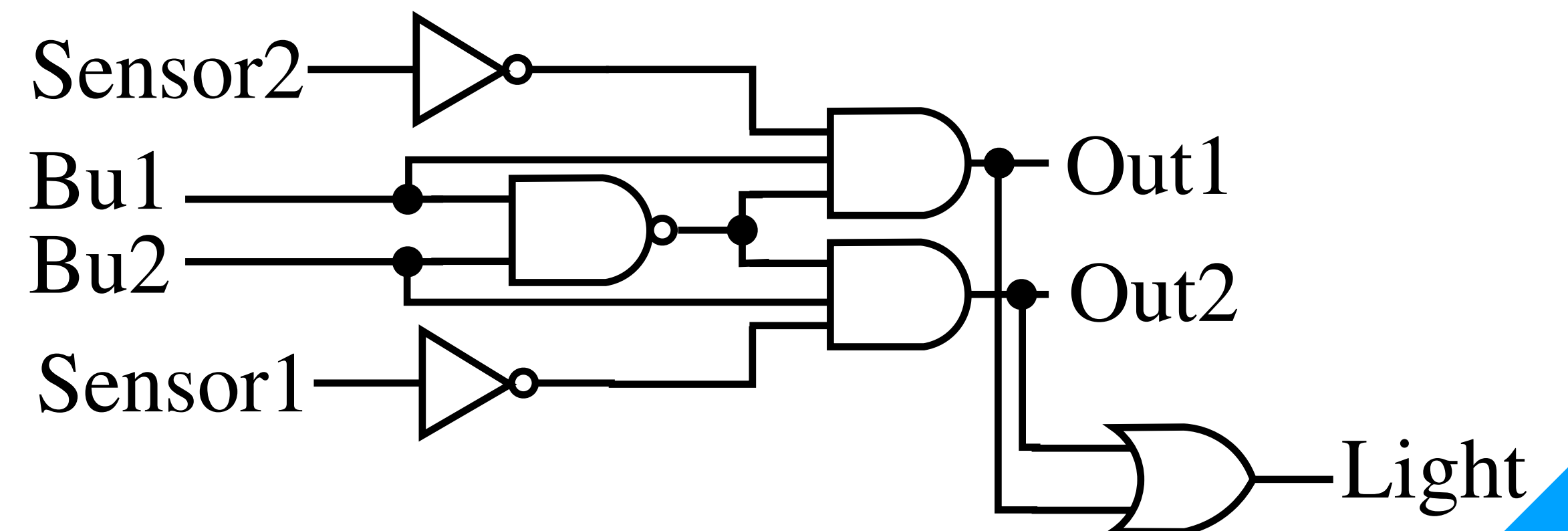
$$\text{Light} = \text{Out1} + \text{Out2}$$

$$\text{Out1} = \text{Bu1} \cdot \overline{\text{Sensor2}} \cdot \overline{\text{Bu1}} \cdot \text{Bu2}$$

$$\text{Out2} = \text{Bu2} \cdot \overline{\text{Sensor1}} \cdot \overline{\text{Bu1}} \cdot \text{Bu2}$$

- Available Components (Technology)
- AND, NAND, OR gate; Switch, Prob

NO NOT GATE!



An Old Friend

4. Technology Mapping

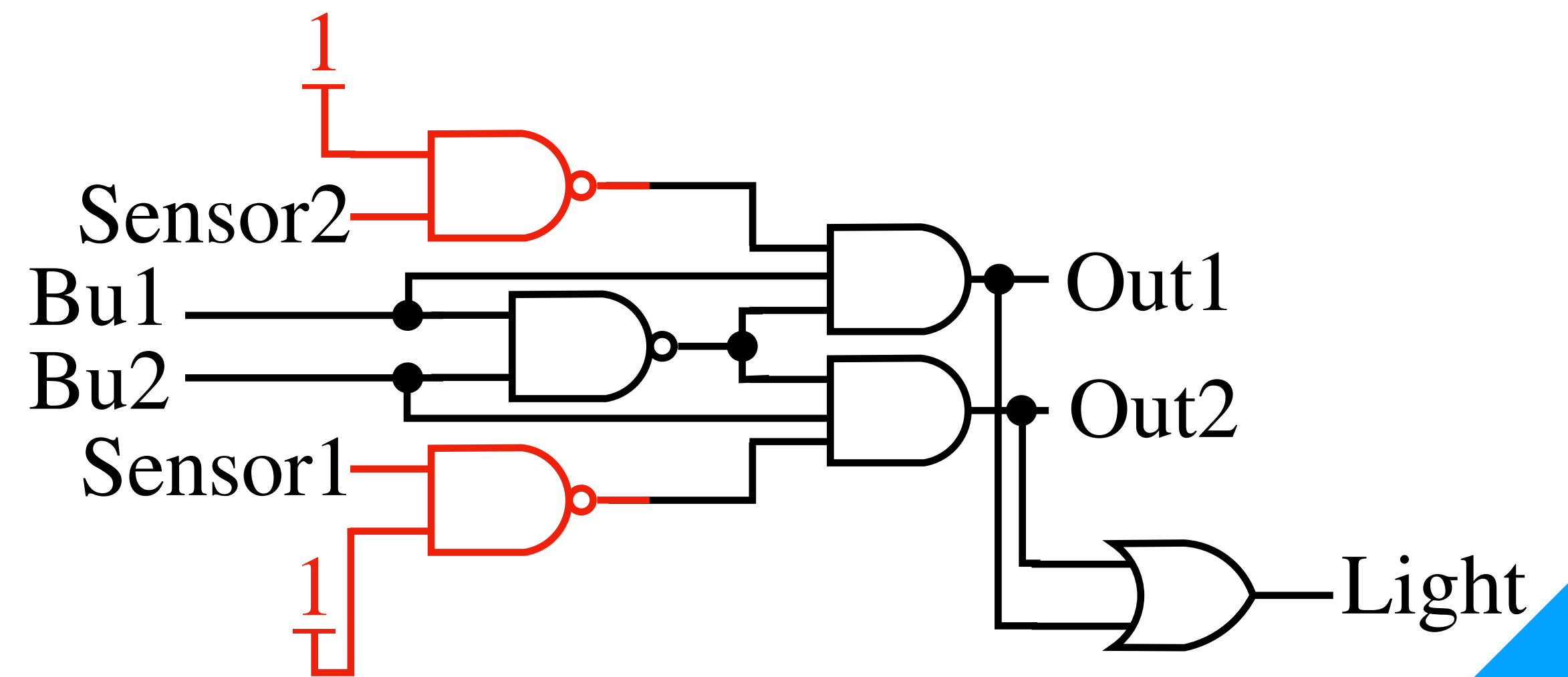
$$\text{Light} = \text{Out1} + \text{Out2}$$

$$\text{Out1} = \text{Bu1} \cdot \overline{\text{Sensor2}} \cdot \overline{\text{Bu1}} \cdot \text{Bu2}$$

$$\text{Out2} = \text{Bu2} \cdot \overline{\text{Sensor1}} \cdot \overline{\text{Bu1}} \cdot \text{Bu2}$$

- Available Components (Technology)
- AND, NAND, OR gate; Switch, Prob

NO NOT GATE!



An Old Friend

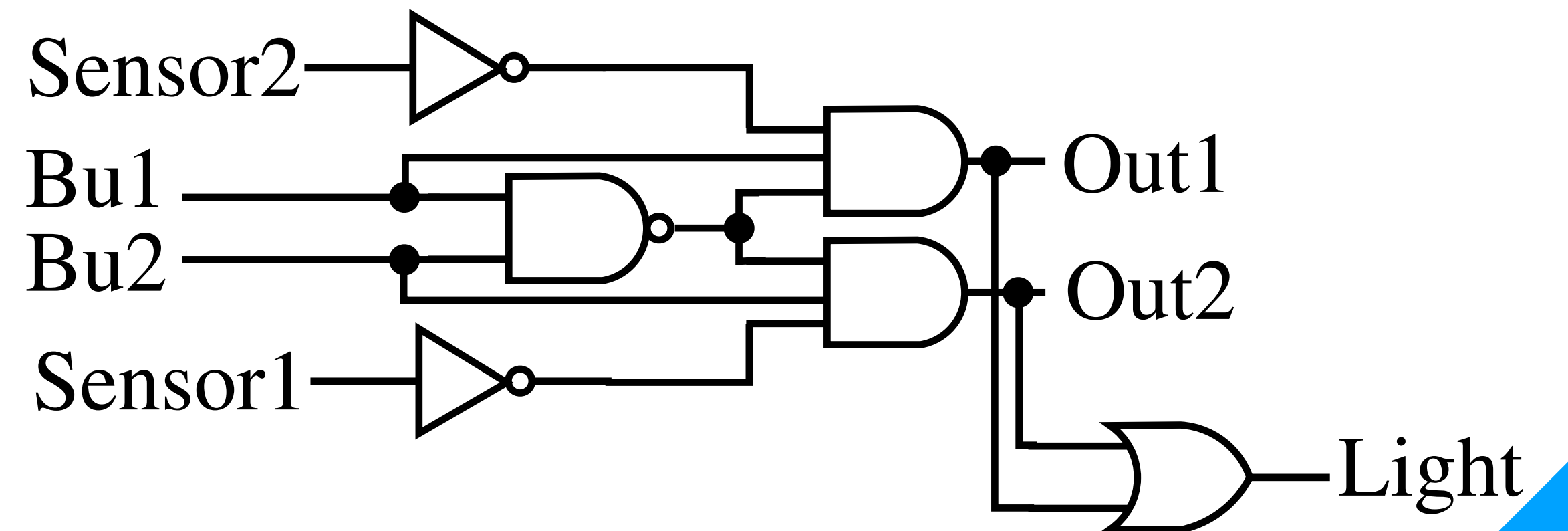
4. Technology Mapping

$$\text{Light} = \text{Out1} + \text{Out2}$$

$$\text{Out1} = \text{Bu1} \cdot \overline{\text{Sensor2}} \cdot \overline{\text{Bu1}} \cdot \text{Bu2}$$

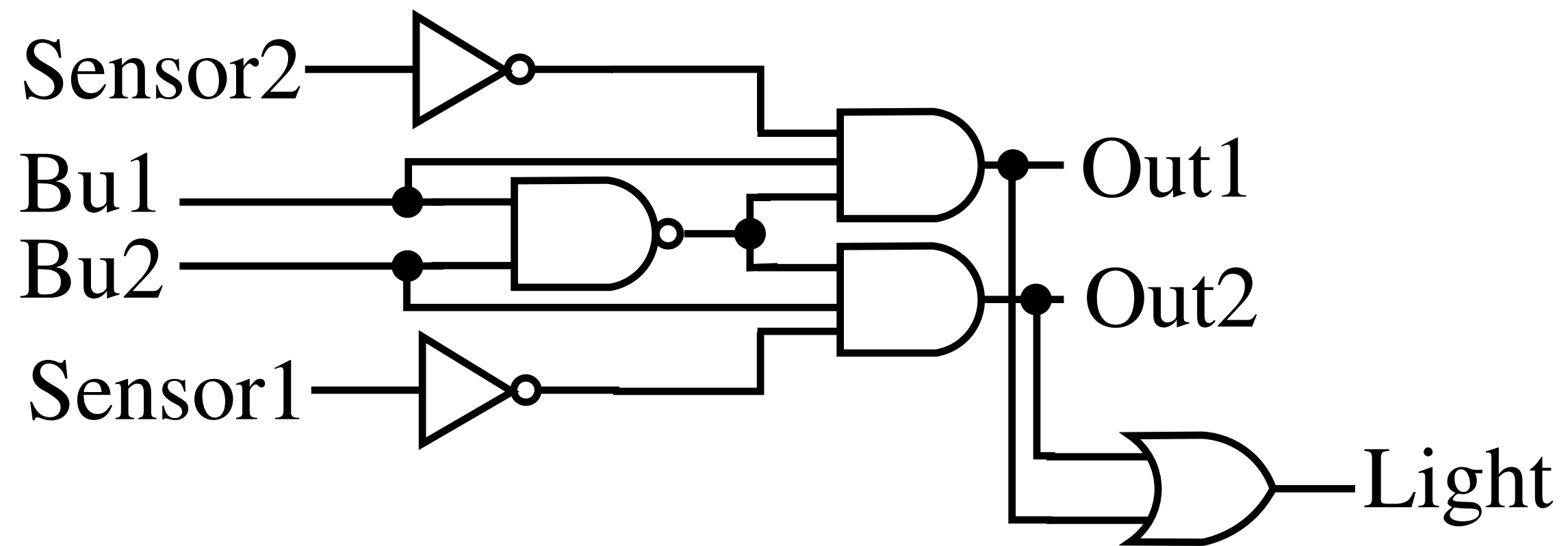
$$\text{Out2} = \text{Bu2} \cdot \overline{\text{Sensor1}} \cdot \overline{\text{Bu1}} \cdot \text{Bu2}$$

- Available Components (Technology)
 - AND, NAND, OR, NOT gate; Switch, Prob



An Old Friend

5. Verification



- Design test cases
 - Sensor2, Bu1 on
 - Sensor1, Bu2 on
 - Bu1, Bu2 on
 - Bu1 on
 - Bu2 on
 - etc...