E40M
Useless Box, Boolean Logic
Useless Box Lab Project #2

- Motor
- Battery pack
- Two switches
  - The one you switch
  - A limit switch

The first version of the box you will build uses mechanical switches to determine the “state” of the box. [https://www.youtube.com/watch?v=aqAUmgE3WyM](https://www.youtube.com/watch?v=aqAUmgE3WyM)

Adding a computer (Arduino) makes the box much more interesting.

e.g. [https://www.youtube.com/watch?v=-PqcCjFaf3I](https://www.youtube.com/watch?v=-PqcCjFaf3I)
Useless Box Lab Project #2

In order to add the Arduino to the box, we now need to understand some additional concepts that are introduced in this and the next few lectures.

- Concepts
  - Finite State Machines
  - Digital Logic
  - Binary numbers
  - CMOS Gates
Useless Box Lab Project #2

The concepts we’ll discuss will help you to understand how modern digital systems work.
Readings For This Material

- Chapter 4 in the reader up to MOS transistors
- For more details
  - A&L 5.1 Digital Signals
    (goes in much more detail than we need)
Useless Box Operation

• The simple version of the Useless Box uses switches, batteries and a motor.

• In order to figure out how to wire these components together, we can use an “action diagram” to illustrate what we want the box to do.

• Friday’s Prelab lecture will discuss how to actually wire the components in a circuit. We’ll discuss the concepts today.
What Do We Want It To Do?

• The motor in the box can be in three different states
  • Forward
  • Reverse
  • Stop

• How does it know when to change state?
Action Diagram - Finite State Machine

Forward  Stop  Reverse
Useless Box Operation – Boolean Logic

• The motor could be in one of three states:
  – Forward, reverse, off
  – State determined by the voltage on the motor terminals

<table>
<thead>
<tr>
<th>State</th>
<th>M+</th>
<th>M-</th>
</tr>
</thead>
<tbody>
<tr>
<td>Forward</td>
<td>4.5V</td>
<td>0V</td>
</tr>
<tr>
<td>Reverse</td>
<td>0V</td>
<td>4.5V</td>
</tr>
<tr>
<td>Off</td>
<td>0V (4.5V)</td>
<td>0V (4.5V)</td>
</tr>
</tbody>
</table>

• This voltage is set by the position of two switches:
  – Switch1
    • On or not on
  – Switch2
    • Limit or not limit
Boolean Variables

• The voltages on the wires in this circuit have two values
  – At least two stable values
    • 4.5V and Gnd

• The switches also seem to have two values (positions)
  – On, off; at limit and not at limit

• What does this remind you of?
  – A Boolean variable?

• Boolean Logic is a form of algebra in which all variables are reduced to True and False (1 and 0 in a binary numbering system).
Electrical Boolean Signal

- Still is just a voltage on a node
  - And to find the voltage you use nodal analysis
    - Or some short cut

- But the voltages of the node settles to only two values
  - True (1) is a high value near the supply (4.5V)
  - False (0) is a low value near the reference (Gnd)

- Each node carries one bit of information

Boole’s thinking has become the practical foundation of digital circuit design and the theoretical grounding of the digital age.
Useless Box Operation

- Think about the situation in logical values

<table>
<thead>
<tr>
<th>State</th>
<th>M+</th>
<th>M-</th>
</tr>
</thead>
<tbody>
<tr>
<td>Forward</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>Reverse</td>
<td>false</td>
<td>true</td>
</tr>
<tr>
<td>Off</td>
<td>false</td>
<td>true</td>
</tr>
</tbody>
</table>

- These outputs are a function of two switches:
  - OnSwitch
    - True, false
  - LimitSwitch
    - True, false
Useless Box Program

If (SwitchOn){
    Motor = Forward;
}
else {
    if (Limit){
        Motor := Stop;
    }
    else {
        Motor = Reverse;
    }
}

- Computer programs use Boolean logic
Logical Operands in Programs (C)

• Switches are either on or off
  – Generally represented by True or False

• Type: Boolean
  – Values are True and False

• Operators:
  – (A && B) AND – Both have to be true
  – (A || B) OR – True if either is true
  – !(A) NOT – True if A is false
Useless Box Boolean Expression

- **SwitchOn** is either true or false; **Limit** is either true or false;
- Can represent **Motor** using two Boolean variables
  - **Forward** is either true or false; **Reverse** is either true or false
    - It is an error if both are true

- What is the Boolean expression for this FSM (Finite State Machine)
  - **Forward**
  - **Reverse**
In most programming languages
   - True = 1; False = 0

So to build a circuit that can represent a bit \{0,1\}
   - Need something that can drive its output to either:
     * The power supply voltage (which we call Vdd)
     * Or the reference level (which we call ground, or gnd)

In the useless box we built the logic from switches
   - And the first computers used mechanical switches too
     * Relays. (Mentioned in the last lecture)
     * But that is so yesterday …
Modern Digital Logic - CMOS

- In the next set of lecture notes you’ll learn about CMOS logic gates that perform digital logic operations.
- Your Arduino has tens of thousands of these gates.

- CMOS “NAND” Gate
Truth Tables & Logic Gates

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>AND</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

\[(A \&\& B) \text{ AND}\]

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>OR</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

\[(A \| B) \text{ OR}\]

<table>
<thead>
<tr>
<th>A</th>
<th>NOT</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

\[!(A) \text{ NOT}\]

Logic Gate Symbols

- **NOT** gate
- **AND** gate
- **NAND** gate
- **OR** gate
- **NOR** gate
- **XOR** gate
**V\textsubscript{dd} and Gnd**

- For many circuits
  - The bottom supply is chosen as the reference
    - So it is called Gnd
  - And many devices connect to the same power supply
    - This is often called V\textsubscript{dd} (or V\textsubscript{cc})

- We’ll see specific examples in the next set of notes when we discuss CMOS transistors and logic gates.
Symbols For $V_{dd}$ and Ground
Learning Objectives

• Understand how to describe a simple system as a finite state machine

• How to represent a Boolean signal in an electrical circuit
  – $V_{dd} = \text{True}; \ G\text{n}\text{d} = \text{False}$
  – Understand the function of AND, OR, NOT operations