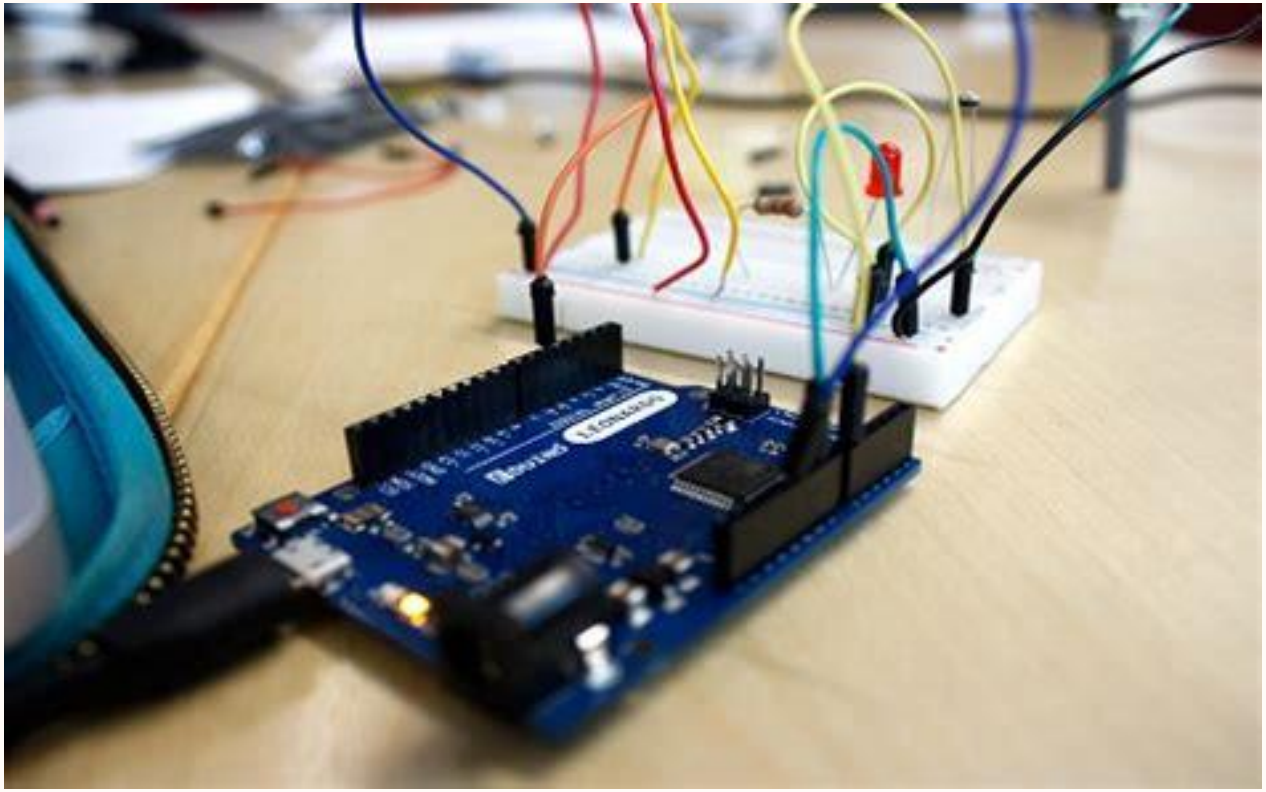
2022/23 SLD Project

The Arduino Project



Jan 2023

Grant Knowles

Agenda

- Morning Clinic
  - Kit Buster Project Objective
  - Project functionality
- Afternoon Workshop
  - Physical Implementation
  - Step 1 – Servo Operation
  - Step 2 – LED Implementation
  - Step 3 – Buzzer Implementation
  - Further Evolution

Objective

- To expand our Arduino knowledge through implementation of a more complex solution
- Integrate functionality from multiple Elegoo Lessons
- Explore program flow
- Explore programming approach
- Learn new code functions

Objective

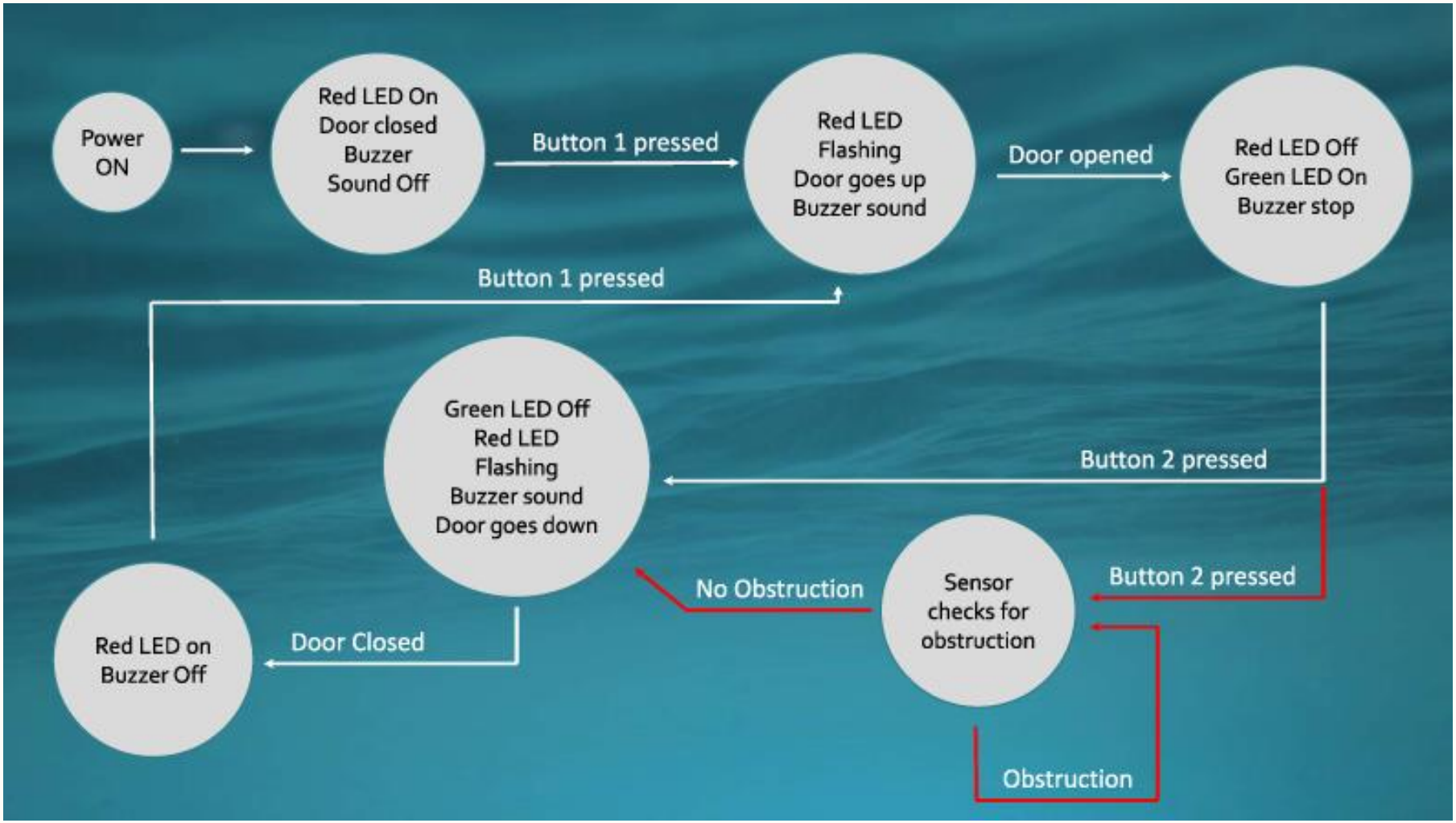
- To implement Lloyd's Brewery Door solution for a single door.
- Basic functionality:
  - Motorized door
  - Status Indicator lights
  - Warning sounds
  - Obstruction Sensor

Operation

- Real life scenario.

- Train approaches the door, is greeted with a closed door and solid red light.
- The conductor presses Button 1 requesting the door to be opened.
- The red light flashes, the buzzer sounds and the door opens
- Once the door opens, the red light is turned off, the green light turns on and the buzzer stops
- The train advances, drops off the car and backs out
- The conductor presses Button 2 requesting the door to be closed.
- The system confirms there are no obstructions at the door
- The red light flashes, buzzer sounds and the door closes.
- Once the door is closed, the red light turns solid and the buzzer stops.

Flow Chart


Power ON
Red LED On
Door closed
Buzzer
Sound Off
Button 1 pressed
Red LED
Flashing
Door goes up
Buzzer sound
Door opened
Red LED Off
Green LED On
Buzzer stop
Button 1 pressed
Green LED Off
Red LED
Flashing
Buzzer sound
Door goes down
Button 2 pressed
Red LED on
Buzzer Off
Door Closed
No Obstruction
Sensor
checks for
obstruction
Button 2 pressed
Obstruction

Summary

- To expand our Arduino knowledge through implementation of a more complex solution
- Integrate functionality from multiple Elegoo Lessons
- Explore program flow
- Explore programming approach
- Learn new code functions

- The Afternoon Workshop
  - Physical Implementation
  - Step 1 – Servo Operation
  - Step 2 – LED Implementation
  - Step 3 – Buzzer Implementation
  - Further Evolution

The End

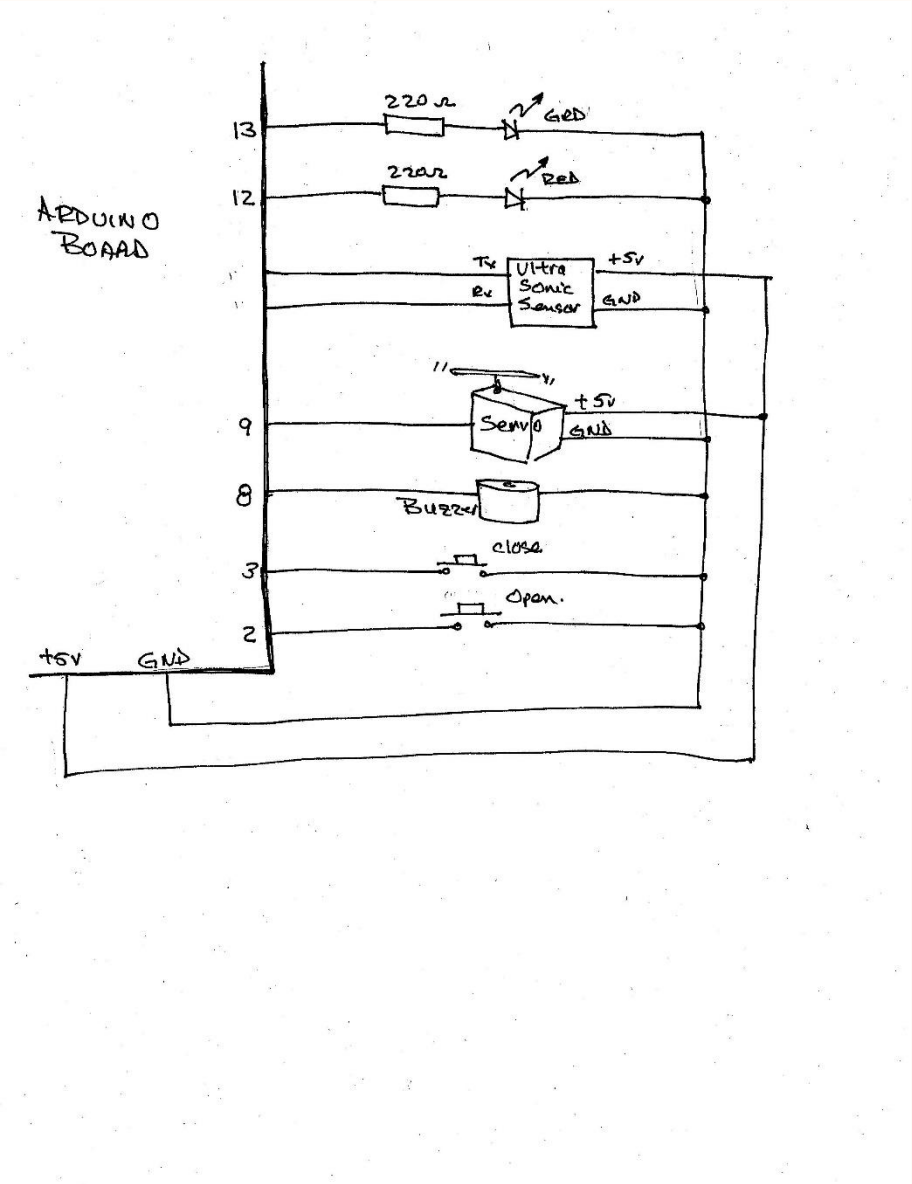
Afternoon Workshop

- The Afternoon Workshop
  - Physical Implementation
  - Step 1 – Servo Operation
  - Step 2 – LED Implementation
  - Step 3 – Buzzer Implementation
  - Further Evolution

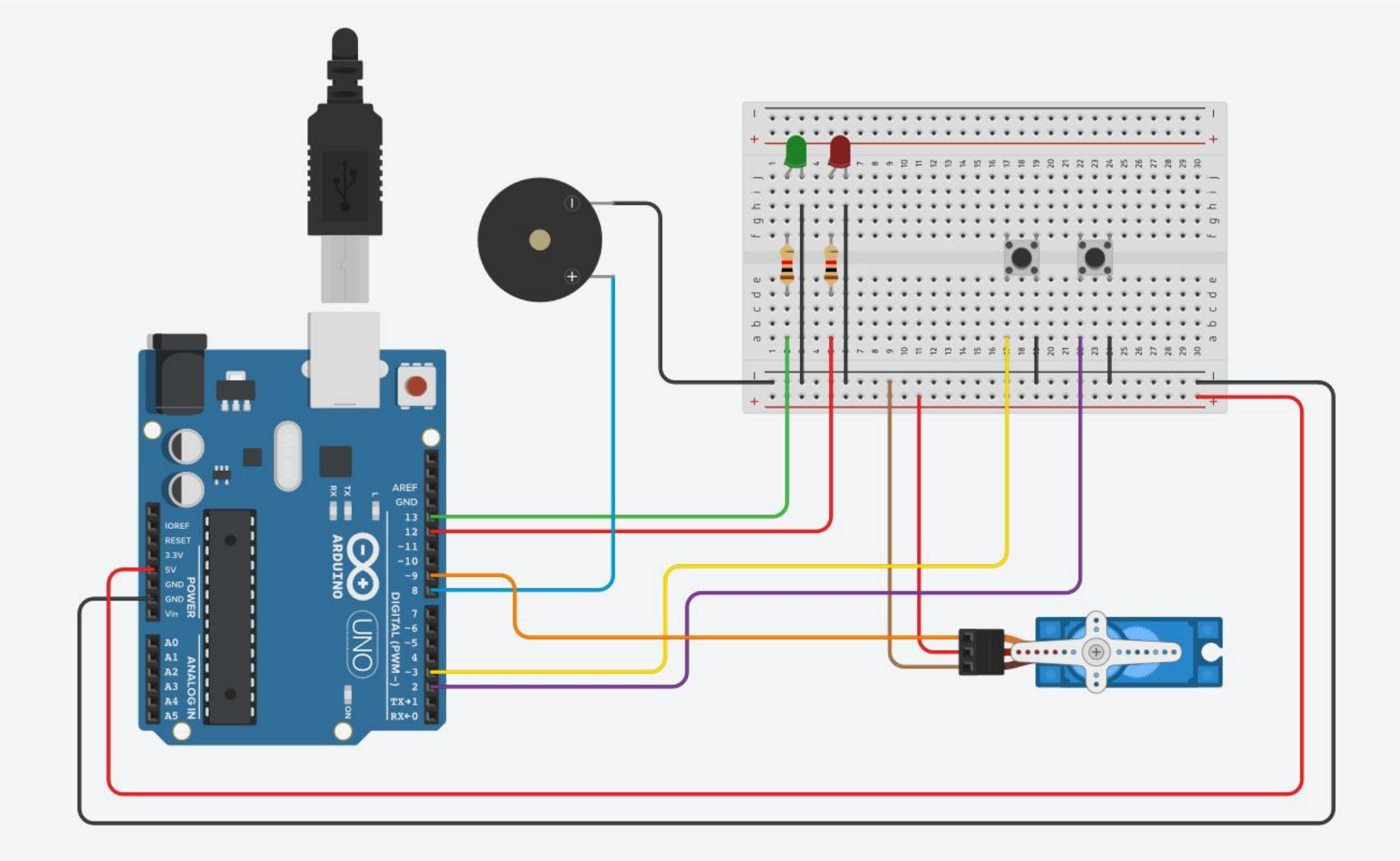
Step 1 - Parts Required

- UNO board
- Servo
- LEDs: green red
- Resistor 220 Ohm, qty:2
- Buzzer
- Buttons, qty:2
- Jumper Wires

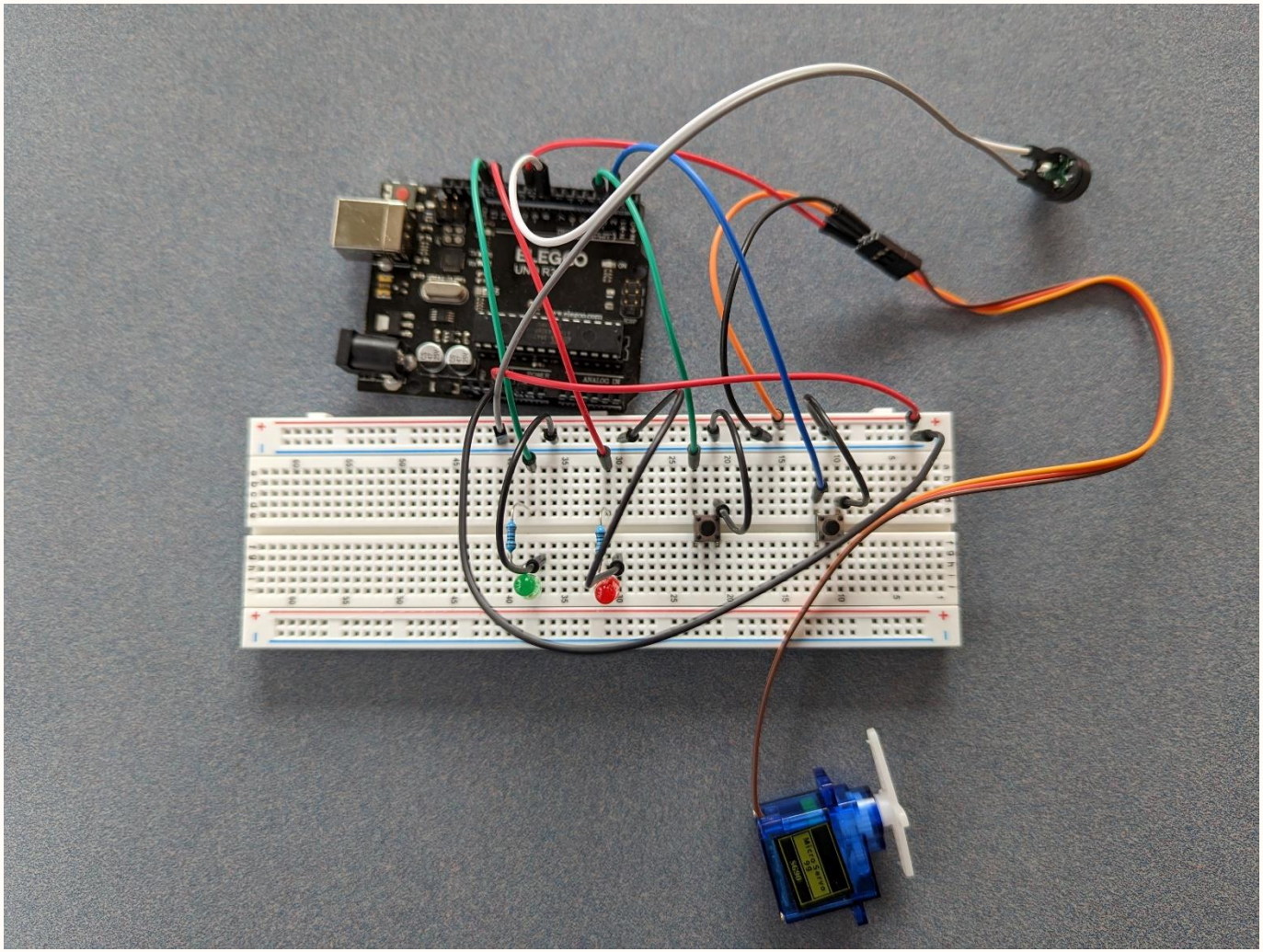
Step 1 - Schematic



Step 1 - Physical Implementation



Step 1 - Physical Implementation



Step 1 – Build Solution

- Now it’s time for getting your fingers dirty!
- Assembly your hardware as per the schematic

Project Approach

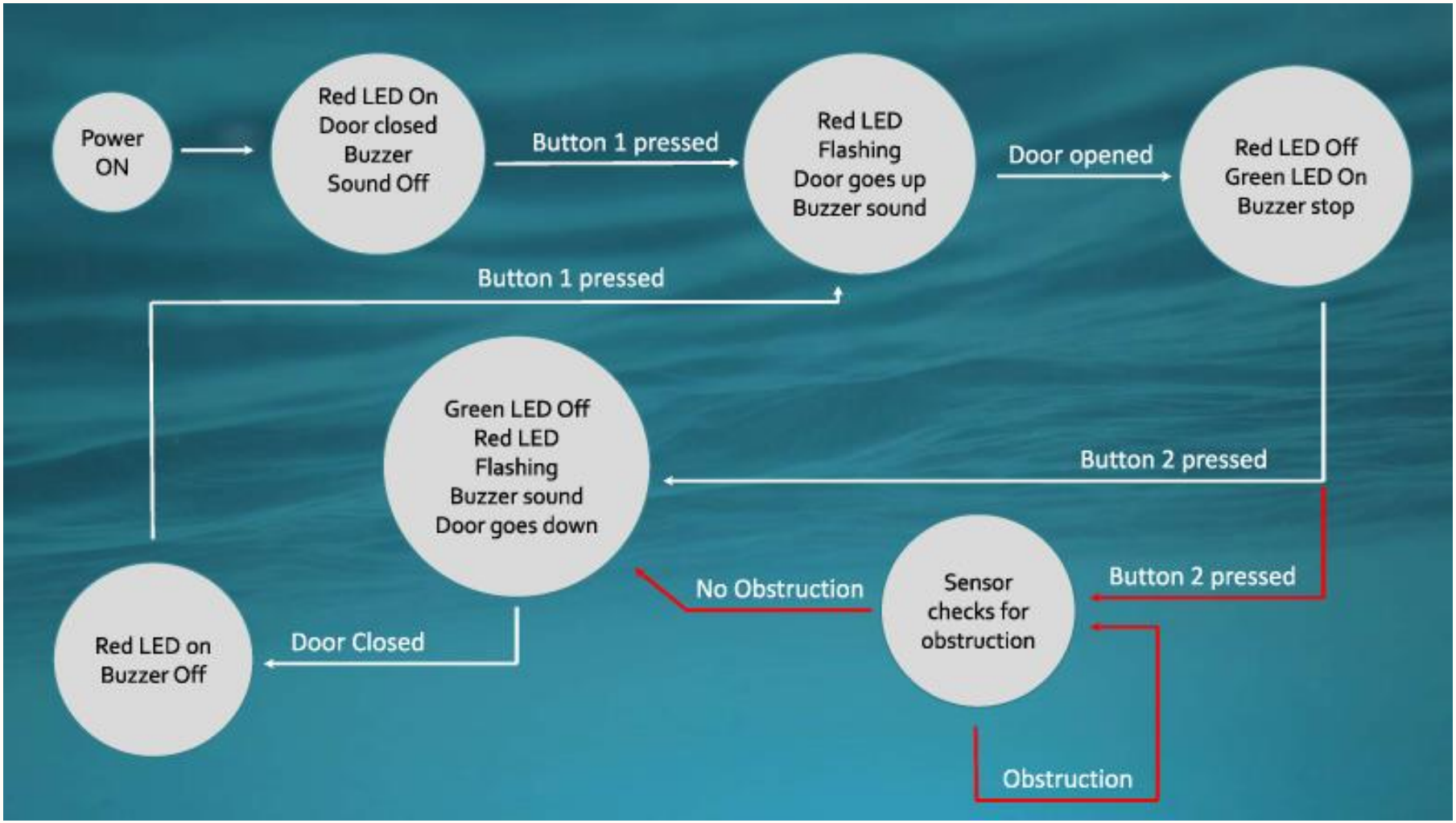
- There are more than one way to solve a problem!
- This happens to be the route I chose!

Operation

- Real life scenario.

- Train approaches the door, is greeted with a closed door and solid red light.
- The conductor presses Button 1 requesting the door to be opened.
- The red light flashes, the buzzer sounds and the door opens
- Once the door opens, the red light is turned off, the green light turns on and the buzzer stops
- The train advances, drops off the car and backs out
- The conductor presses Button 2 requesting the door to be closed.
- The system confirms there are no obstructions at the door
- The red light flashes, buzzer sounds and the door closes.
- Once the door is closed, the red light turns solid and the buzzer stops.

Flow Chart


Power ON
Red LED On
Door closed
Buzzer
Sound Off
Button 1 pressed
Red LED
Flashing
Door goes up
Buzzer sound
Door opened
Red LED Off
Green LED On
Buzzer stop
Button 1 pressed
Green LED Off
Red LED
Flashing
Buzzer sound
Door goes down
Button 2 pressed
Red LED on
Buzzer Off
Door Closed
No Obstruction
Sensor
checks for
obstruction
Button 2 pressed
Obstruction

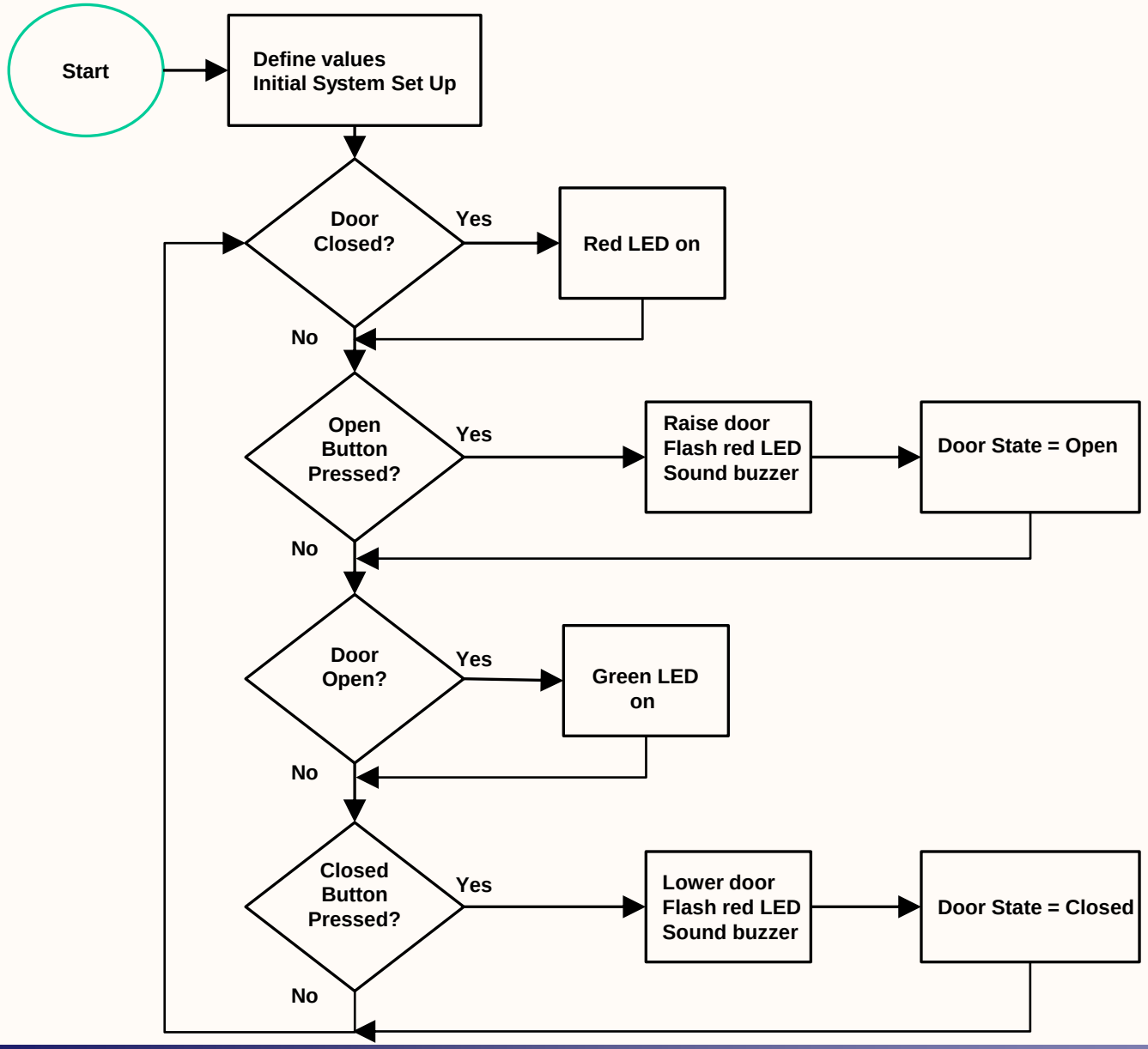
Functionality Approach

- Based on Door Status
  - Closed
  - Opening
  - Open
  - Closing
- State Change
  - Requested via Push Buttons
- Outputs
  - Servo
  - LEDs
  - Buzzer

Project Approach

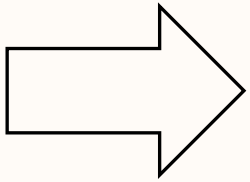
- To implement our solution in 3 increments;
  - Step 1 – Core functionality
    - Read buttons
    - Activate servo
  - Step 2 – Add lights
  - Step 3 – Add sound
  - Step Future – Add obstruction detection

Flow Chart


Start
Define values
Initial System Set Up
Door
Closed?
Yes
Red LED on
No
Open
Button
Pressed?
Yes
Raise door
Flash red LED
Sound buzzer
Door State = Open
No
Door
Open?
Yes
Green LED
on
No
Closed
Button
Pressed?
Yes
Lower door
Flash red LED
Sound buzzer
Door State = Closed
No

Flow Chart – Servo / LED / Buzzer

Raise door
Flash red LED
Sound buzzer



Servo
For (start angle, max angle, increment)
Increment servo by one degree
Write to servo

LED Blink
Based on system clock
Switch LED state
Write to LED – on/off toggle

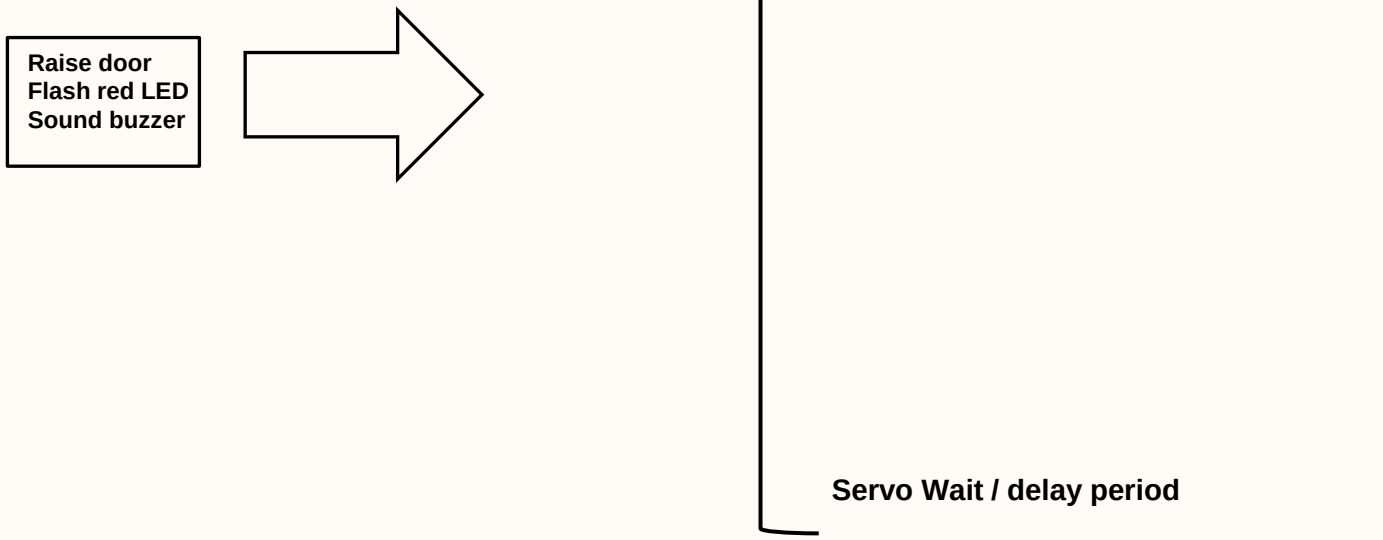
Buzzer Tone
Based on system clock
Switch tones
Write to buzzer – audio tone

Servo Wait / delay period

Step 1 – Buttons & Servo

- First Configuration
- Functionality supported
  - Buttons – Open / Close requests
  - Servo – Open / Close door
  - LED indication – Open / Close

Flow Chart - Servo



Software Components

- Leverage Elegoo Lessons;
  - Push Buttons
  - LEDs
  - Servo
  - Buzzer
- Borrow from the web
- Access our own experts

Step 1 -

- **Initial sketch**
  - **Read Buttons,**
  - **Status LEDs,**
  - **Servo movement**

Step 1 - Push Buttons

- **Lesson 5 Digital Inputs**
  - **Monitor the Open / Close request buttons**

```
int ledPin = 5;
int buttonApin = 9;
int buttonBpin = 8;

byte leds = 0;

void setup() {
   pinMode(ledPin, OUTPUT);
   pinMode(buttonApin, INPUT_PULLUP);
   pinMode(buttonBpin, INPUT_PULLUP);
 }
void loop() {
   if (digitalRead(buttonApin) == LOW)  {
      digitalWrite(ledPin, HIGH);
   }
   if (digitalRead(buttonBpin) == LOW)  {
      digitalWrite(ledPin, LOW);
   }
}
```

Step 1 - LEDs

- **Lesson 3 LEDs**
  - **Red / Green Status LEDs**
  - **Flash Red LED while door is in motion**
    - **Adjustable flash rate**

```
void setup() {
  pinMode(5, OUTPUT);  //  initialize digital pin 5 as an output.
}

// the loop function runs over and over again forever

void loop() {
  digitalWrite(5, HIGH);   //  turn the LED on (HIGH is the voltage level)
  delay(1000);             //  wait for a second
  digitalWrite(5, LOW);    //  turn the LED off by making the voltage LOW
  delay(1000);             //  wait for a second
}
```

Step 1 - Servo

- **Lesson 9 Servo**
  - **Control movement speed**
  - **Adjustable upper / lower stop range**

```
#include “Servo.h”

Servo myservo;  // create servo object to control a servo

int pos = 0;    // variable to store the servo position

void setup() {
  Serial.begin(9600);
  myservo.attach(9);  // attaches the servo on pin 9 to the servo object
}

void loop() {
  for (pos = 0; pos <= 180; pos += 1) { // goes from 0 degrees to 180 degrees in steps of 1 degree
    myservo.write(pos);              // tell servo to go to position in variable 'pos'
    delay(15);                       // waits 15ms for the servo to reach the position
  }
  for (pos = 180; pos >= 0; pos -= 1) { // goes from 180 degrees to 0 degrees
    myservo.write(pos);              // tell servo to go to position in variable 'pos'
    delay(15);                       // waits 15ms for the servo to reach the position
  }
}
```

Step 1 – Code Walk Thru

- **Review the Step 1 sketch**

Step 1 – Run Sketch

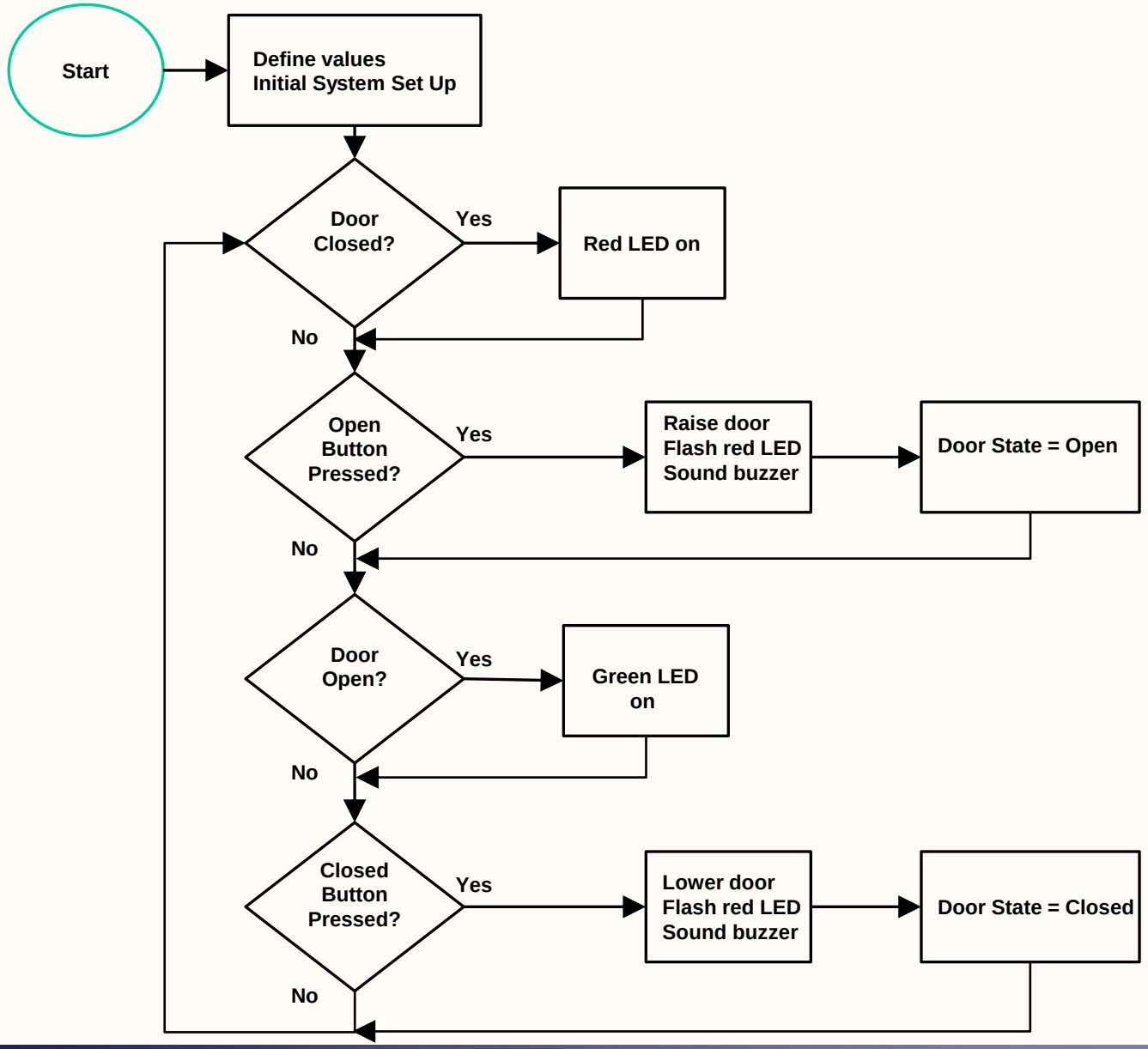
- Compile & Load the first sketch
  - Step 1 – Buttons_Servo

- Exercise the servo/LEDs by pressing the buttons

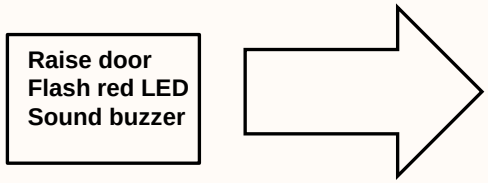
Step 2 – Buttons, Servo & LEDs

- Second Configuration
- Extends the Step 1 configuration
- Incremental functionality supported
  - Flashing LEDs while the door is in motion
- New challenge
  - How to perform two functions in parallel with a single thread processor?
    - Operate servo
    - Flash red LED

Flow Chart


Start
Define values
Initial System Set Up
Door
Closed?
Yes
Red LED on
No
Open
Button
Pressed?
Yes
Raise door
Flash red LED
Sound buzzer
Door State = Open
No
Door
Open?
Yes
Green LED
on
No
Closed
Button
Pressed?
Yes
Lower door
Flash red LED
Sound buzzer
Door State = Closed
No

Flow Chart – Servo / LED



Servo
For (start angle, max angle, increment)
Increment servo by one degree
Write to servo

LED Blink
Based on system clock
Switch LED state
Write to LED – on/off toggle

Servo Wait / delay period

Step 2 – Flash LED

- **Sketch code walk thru**

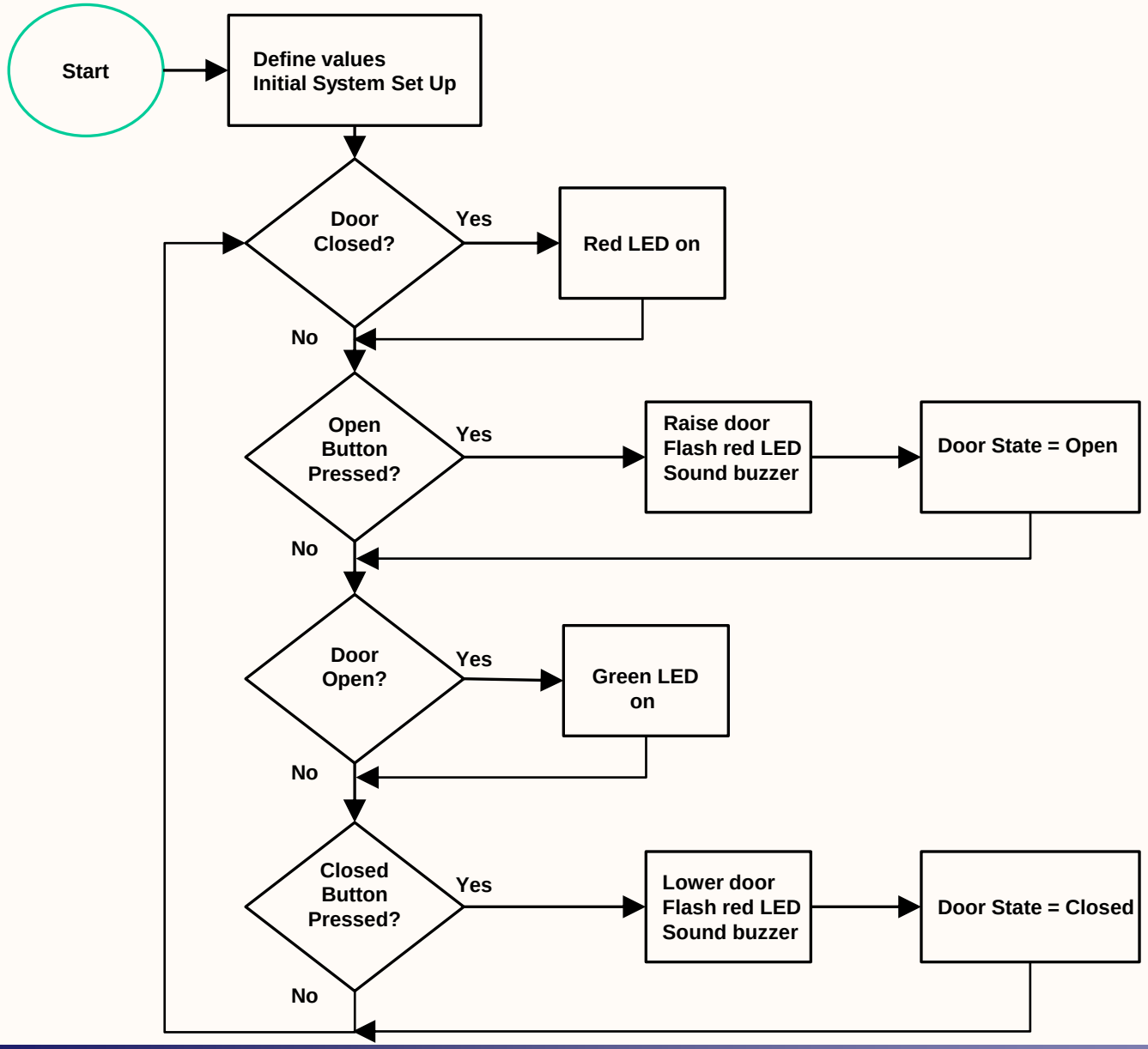
Step 2 – Build Solution

- Play time again!
- No hardware changes are required.
- Compile & Load the second sketch
  - Step 2 – Buttons_Servo_LED
- Exercise the servo/LEDs by pressing the buttons

Step 3 – Buttons, Servo, LEDs & Buzzer

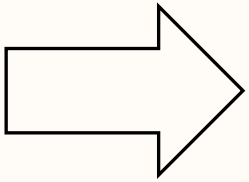
- Third Configuration
- Extends the Step 2 configuration
- Incremental functionality supported
  - Sound Buzzer while the door is in motion
- Further challenge
  - How to perform three discrete functions in parallel with a single thread processor?
    - Operate servo
    - Flash red LED
    - Sound buzzer (alternate frequencies)

Flow Chart


Start
Define values
Initial System Set Up
Door
Closed?
Yes
Red LED on
No
Open
Button
Pressed?
Yes
Raise door
Flash red LED
Sound buzzer
Door State = Open
No
Door
Open?
Yes
Green LED
on
No
Closed
Button
Pressed?
Yes
Lower door
Flash red LED
Sound buzzer
Door State = Closed
No

Flow Chart

Raise door
Flash red LED
Sound buzzer



Servo
For (start angle, max angle, increment)
Increment servo by one degree
Write to servo

LED Blink
Based on system clock
Switch LED state
Write to LED – on/off toggle

Buzzer Tone
Based on system clock
Switch tones
Write to buzzer – audio tone

Servo Wait / delay period

Buzzer

- **Lesson 7 Passive Buzzer**
  - **Two tone sound**
  - **Adjustable frequencies**
  - **Adjustable duration**

```
#include "pitches.h"

// notes in the melody:
int melody[] = {
  NOTE_C5, NOTE_D5, NOTE_E5, NOTE_F5, NOTE_G5, NOTE_A5, NOTE_B5, NOTE_C6};
int duration = 500;  // 500 miliseconds

void setup() {
}

void loop() {
   for (int thisNote = 0; thisNote < 8; thisNote++) {
      tone(8, melody[thisNote], duration);  //  pin8 output the voice, every scale is 0.5 second
      delay(1000);  //  Output the voice after several minutes
  }

delay(2000); // restart after two seconds
}
```

Step 3 – Buzzer

- **Sketch code walk thru**

Step 3 – Build Solution

- Play time again!
- No hardware changes are required.
- Compile & Load the second sketch
  - Step 3 – Buttons_Servo_LED_Buzzer
- Exercise the servo/LEDs by pressing the buttons

Future Enhancements

Future enhancements could include:

- **Incorporate an obstruction detection**
- **Incorporating button debouncer s/w,**
- **Merge the buttons into a single button,**
- **Incorporate error checking in the ultra sonic sensor,**
- **Incorporate a single multi colour LED,**
- **A restart sequence if the power is interrupted,**
- **etc**

Next Steps

- Recent Homework:
  - 10 Ultrasonic Sensor
  - 11 Temperature & Humidity Sensor
  - 12 Joystick
  - 13 IR Receiver
- Ron’s 3 Servo Control Exercises
- Execute Lessons
  - 14 LCD Display
  - 15 Thermometer
  - 16 Eight LED with 74HC595
  - 17 Serial Monitor
  - 18 Photocell

Next Steps

- **Your Project**
  - **Start thinking about what you want to do with your Arduino**

- **Future workshops;**
  - **Monday Feb 6th - ZOOM**
  - **Saturday Feb 25th – Kit Busters Workshop**
  - **Monday Mar 13th - ZOOM**

The End