|  |  |
| --- | --- |
| **Course:** | **Computer Design** |
| **Professor:** | **Evan McCarty (36196)** |
| **Team:** | **Ahmad Awaidah (Aawai), Michael Nguyen (mnguy37), Mike Vinanzaca (mvina2)** |
| **Final Project Title:** | **Remote Control Car** |

**A red circle with white letters

AI-generated content may be incorrect.**

# **Project Documentation**

* **(Overall project Idea Paragraph\_1):** We’re building a remote-controlled car powered by 3-4 Arduinos. It will be two-wheel drive with differential steering, head/brake lights, and a remote control. The goal is to configure the brains of the car with an IR receiver and configure a remote IR controller to control and steer the cars direction (front, back, steer-left, steer-right). Finally, after the electrical circuit of the car is finalized and is working with the remote control. We’ll design a removeable plastic cover designed as a car of our choice to cover the electrical circuit, to simulate a true car’s design.
* **(Description of multiple Arduinos Paragraph\_2) (Input/Output device types 2 devices per person):** Each Arduino will have its own job:
  + **Arduino\_1 (A1):** *(2 Motors)* A1 will use 2 motors for two-wheel drive for the front two wheels with differential steering, implemented through the following dynamic:
    - Turn right: left wheel faster than right wheel
    - Turn left: right wheel faster than left wheel
  + **Arduino\_2 (A2):** *(LED Headlights, LED Brake lights, Light sensor)* A2 will use yellow/white LED lights for the headlights, and red LED lights for the brake lights. Additionally, we’ll investigate configuring the LED lights to turn on when certain metrics are hit.
  + **Arduino\_3 (A3):** *(Joysticks, IR Controller, and Buttons)* A3 will be the remote controller for the remote controller car.
  + **Arduino\_4 (A4):** *(IR Receiver, Speaker, LCD Screen)* A4 will contain the IR Receiver and Speaker or additional components serving as play features for the host.
* **(Communication):** This project will be utilizing a remote control managed by Arduino\_3, sending commands to the car using an infrared transmitter (IR). For communication with the car itself, Arduino\_4 will function as the main controller sending instructions to Arduino\_1(motors) and Arduino\_2(lights) using the Inter-Integrated Circuit Protocol (I2C).
* **(Original Work Paragraph\_3):** Original work within this project includes the differential steering, and how we will implement that with the method described above. As well as the plastic cover, how we choose to design it will be completely original. We’ve considered building materials for this car designed cover, which include coke cans, recycled plastic, etc.

# **Programming Language**

* C/C++

# **Integration**

* Arduino IDE

# **In-Scope Hardware**

* **Main Controller:**
* Arduino UNO R4 Wi-Fi (x1)
  + The brain of the project runs the code, controls all other components, and has built in Wi-fi for internet connectivity.
* **Display Components**
* I2C LCD 1602 Display (x1)
  + Shows 16 characters x 2 lines of text. Great for displaying sensor readings, menus, or messages
* OLED Screen (x1)
  + Small high-contrast display. Shows text, graphics, or simple animations.
* 7-segment Display (x1)
  + Show numbers 0-9
* **Sensors**
* Temperature and Humidity Sensor (x1)
  + Measures room temperature and moisture in the air.
* Ultrasonic Module (x1)
  + Measures distance using sounds waves.
* PIR Motion Sensor Module (x1)
  + Detects movements of people.
* MFRC522 RFID Module (x1)
  + Reads RFID cards.
* MPR121 Module (x1)
  + Detects touch up to 12 different surfaces.
* Soil Moisture Module (x1)
  + Measures water content in soil.
* GY-87 10 DOF Module (x1)
  + Advanced sensor combining accelerometer, gyroscope, magnetometer, and barometer.
* Photoresistor (x1)
  + Light sensor that changes resistance based on brightness.
* Thermistor (x1)
  + Temperature sensor that changes resistance with heat.
* **Input/Control Devices:**
* IR Controller (x1)
* IR Receiver (x1)
  + Remote control system. Controller sends signals, receiver gets them.
* Keypad (x1)
  + Grid of buttons for number/letter input. Used for passwords, calculators, or menu navigation.
* Joystick Module (x1)
  + Analog stick controller (like game controller). Controls robots, cursors, or games.
* Potentiometer (x1)
  + Rotating knobs that changes resistance. Used for volume control, brightness adjustment, or analog input.
* Button (x4)
* Button (x10)
  + Simple push buttons for user input. Turn things on/off or trigger actions.
* Tilt Switch (x1)
  + Detects when tilted or shaken. Used in motion alarms or orientation sensing.
* **Output Device – Motors:**
* Stepper Motor (x1)
* Stepper Motor Driver (x1)
  + Precise motor that moves in exact steps.
* 9G Servo (x1)
  + Motor that rotates to specific angles (0-180°). Perfect for robot arms, steering, or precise movement.
* Motor (x1)
  + Basic DC motor for continuous rotation. Used in wheels, fans, or simple movement.
* Pump (x1)
  + Moves liquids from one place to another. Great for watering systems or liquid dispensing.
* Fan (x1)
  + Creates airflow for cooling or ventilation projects.
* **Output Devices – Audio/Viusal**
* Speaker (x1)
* Audio Power Amplifier Module (x1)
  + Plays sounds, music, alarms, or voice. Amplifier makes it louder.
* Passive Buzzer (x1)
  + Makes tones when given specific frequencies. You control the pitch.
* Active Buzzer (x1)
  + Makes a fixed beep sound when powered. Simple on/off alarm.
* Green LED (x5)
* Red LED (x5)
* Yellow LED (x5)
* Blue LED (x5)
* White LED (x1)
* RGB LED (x1)
  + Light indicators. RGB can make any color. Used for status lights, decorations, or indicators.
* WS2812 RGB Strip (x1)
  + Chain of programmable color LEDs. Creates light shows, decorations, or visual feedback.
* **Power & Control:**
* Breadboard Power Module with Battery (x1)
  + Provides portable power to your breadboard projects without needing USB connection.
* Relay (x1)
  + Electronic switch that controls high-power devices. Lets Arduino safely control lights, motors, or appliances.
* **Electronic Components:**
* TA6586 (x1)
* 74HC595 (x1)
* S8550 Transistor (x1)
* S8050 Transistor (x2)
* 1N4007 Diode (x5)
* Capacitor 104 pF (x5)
* Capacitor 10uF (x5)
* Resistor 10Ω (x10)
* Resistor 100Ω (x10)
* Resistor 220Ω (x30)
* Resistor 330Ω (x10)
* Resistor 1KΩ (x10)
* Resistor 2KΩ (x10)
* Resistor 5.1KΩ (x10)
* Resistor 10KΩ (x10)
* Resistor 100KΩ (x10)
* Resistor 1MΩ (x10)
* **Building Materials:**
* Tube (x1)
  + Protective covering or structural component for wiring or mechanical parts.
* Type-C USB Cable (x1)
  + Powers Arduino and uploads your code from computer.
* Jump Wire F/M (x20)
* Jump Wire M/M (x65)
  + Connect components together. M/M = male-to-male, F/M = female-to-male connectors.
* Breadboard (x1)
  + Temporary circuit building platform. Connect components without soldering.

# **In-Scope Functions**

* AnalogWrite
* AnalogRead
* Map()
* Loop()
* ETC…
* X
* Y
* Z

# **Out-of-Scope Variables**

* Svc.Servers
* svc.DriveSpaceInfo
* svc.DatabaseFileInfo

# **Additional Functionality**

* Svc.Servers
* svc.DriveSpaceInfo
* svc.DatabaseFileInfo
* What would you change if we wanted the green LED to be on pin 4 or on pin 11?
* What would you change if we wanted each LED to be on for 20 seconds?
* What would you change if we wanted to use a yellow LED instead of the onboard LED?
* What would you change if we wanted all LED’s to be off for a second before turning the next LED on?
* Why are we not allowed to use delay() to make this lab function?
* What functionality is lost when delay() is used?
* What might happen if we didn’t use a resistor when wiring each LED?
* What might happen if we used a 10 Ohm resistor or a 10K Ohm resistor when wiring each LED?
* If the TA were to remove a wire from your design, how would you debug the situation to determine what actions are needed to fix it?
* How would you determine if a wire is physically broken inside of its plastic insulation?

# **Sources**

* <http://www.multiwingspan.co.uk/arduino.php?page=led9>
* <https://forum.arduino.cc/t/logic-buffer-tristate/1125771>
* <https://forum.arduino.cc/t/how-to-use-tri-state-buffers/667856>
* [**https://reviseomatic.org/help/2-assembler/Assembler%20Tristate%20Logic.php**](https://reviseomatic.org/help/2-assembler/Assembler%20Tristate%20Logic.php)
* [**https://learn.circuit.rocks/the-basic-arduino-schematic-diagram**](https://learn.circuit.rocks/the-basic-arduino-schematic-diagram)
* [**https://docs.arduino.cc/hardware/uno-rev3/**](https://docs.arduino.cc/hardware/uno-rev3/)
* [**https://www.amazon.com/dp/B01D8KOZF4?ref=ppx\_yo2ov\_dt\_b\_fed\_asin\_title**](https://www.amazon.com/dp/B01D8KOZF4?ref=ppx_yo2ov_dt_b_fed_asin_title)