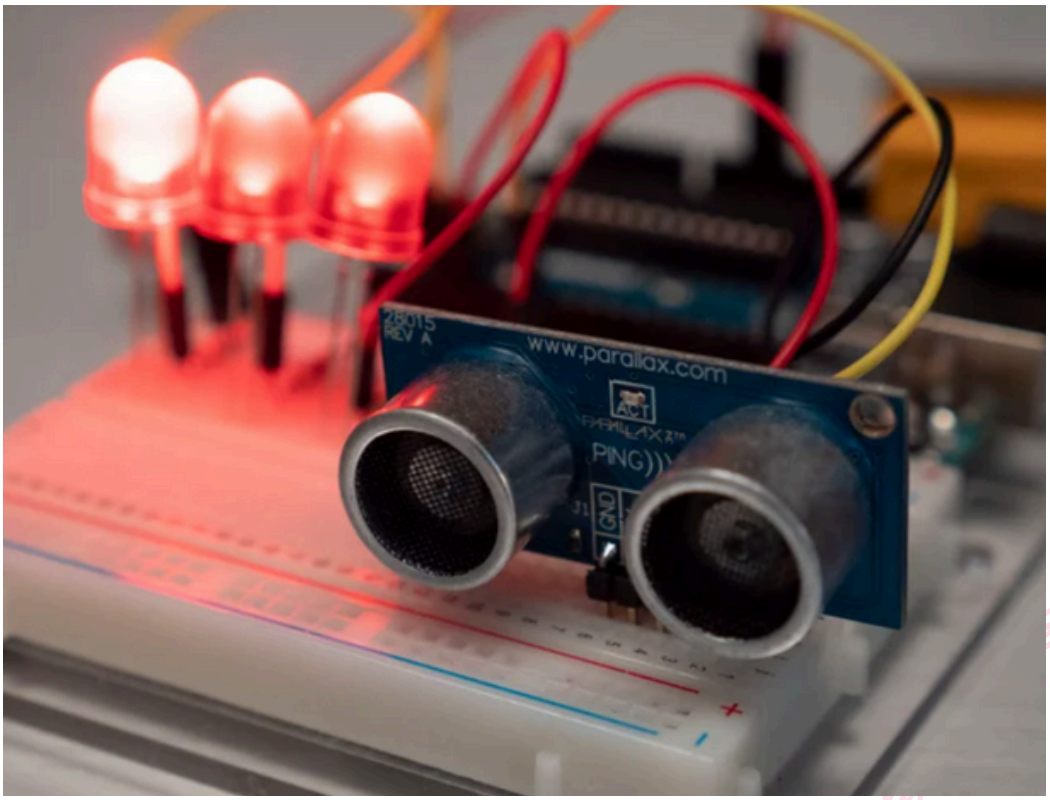
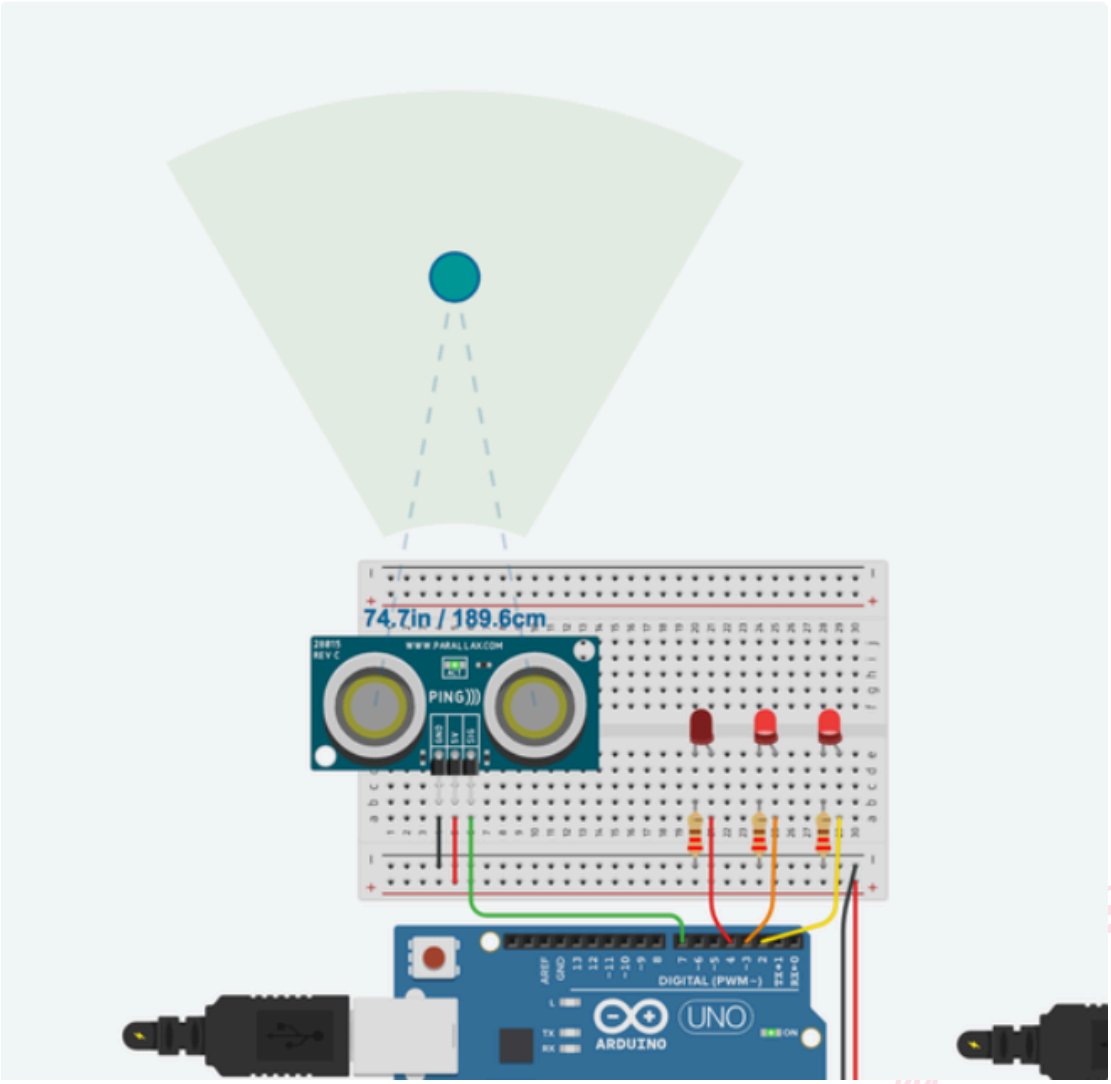
Ultrasonic Distance Sensor in Arduino With TinKercad

Let's explore measuring distances using an ultrasonic sensor with Arduino. We'll connect components on a breadboard and write code to control an LED. Build on your skills with pushbuttons and motion sensors to enhance your Arduino projects.



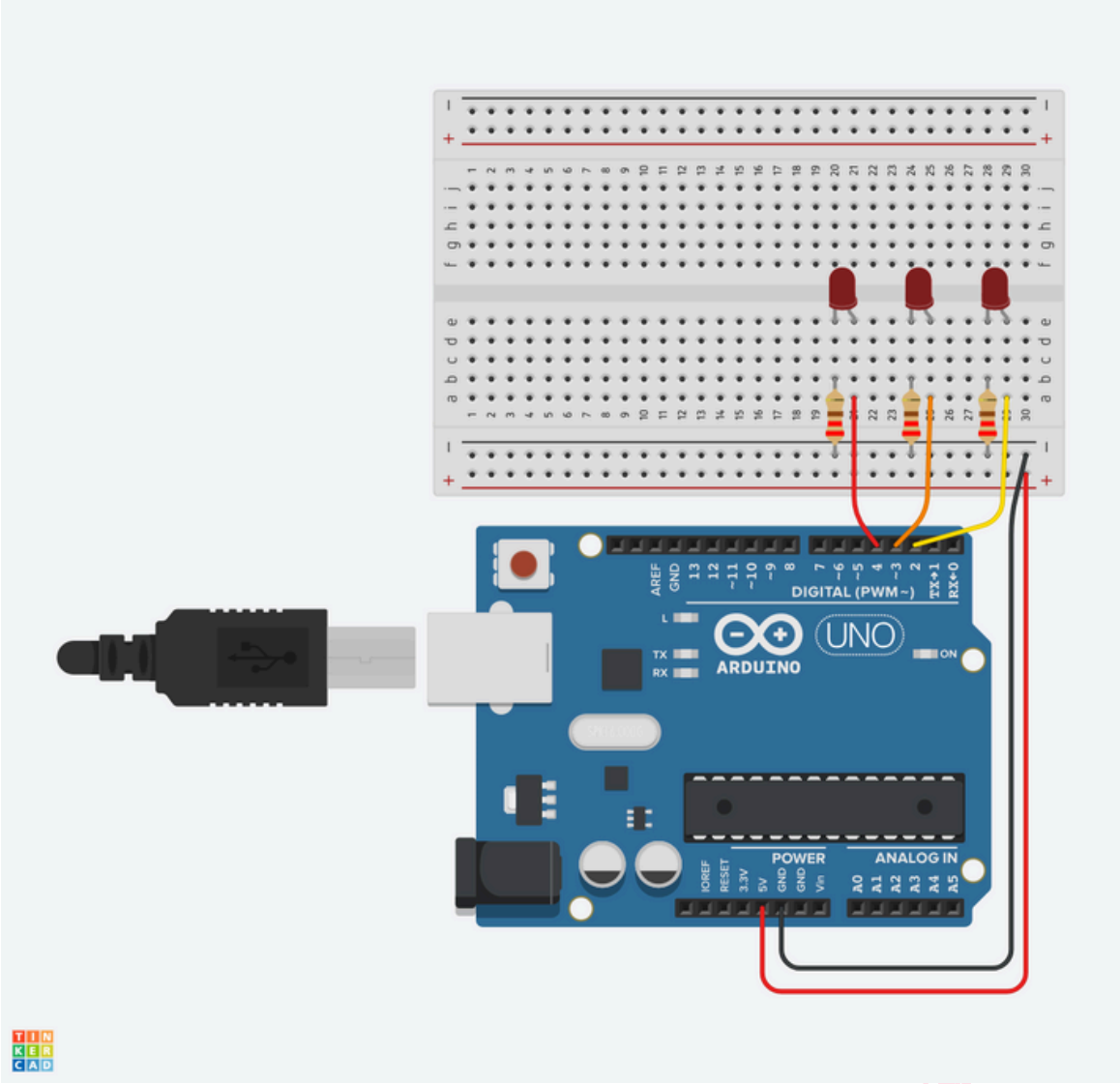
Ultrasonic rangefinders use sound waves to bounce off objects in front of them, much like bats using echolocation to sense their environment. The proximity sensor sends out a signal and measures how long it takes to return. The Arduino program receives this information and calculates the distance between the sensor and object.



- In this lesson, you'll build this simulated circuit yourself along side the sample.

- To optionally build the physical circuit, gather up your Arduino Uno board, USB cable, solderless breadboard, three LEDs, resistors (any value from 100-1K), ultrasonic rangefinder, and breadboard wires.

Step 1 : Build the LED Circuit



- Start by wiring up your Arduino and breadboard with power and ground next to the example circuit.

- Then add the the three red LEDs to the breadboard, as shown. These will be the "bar graph" lights for visually indicating the sensor's distance measurement.

- Drag an Arduino Uno and breadboard from the components panel to the workplane, next to the existing circuit.

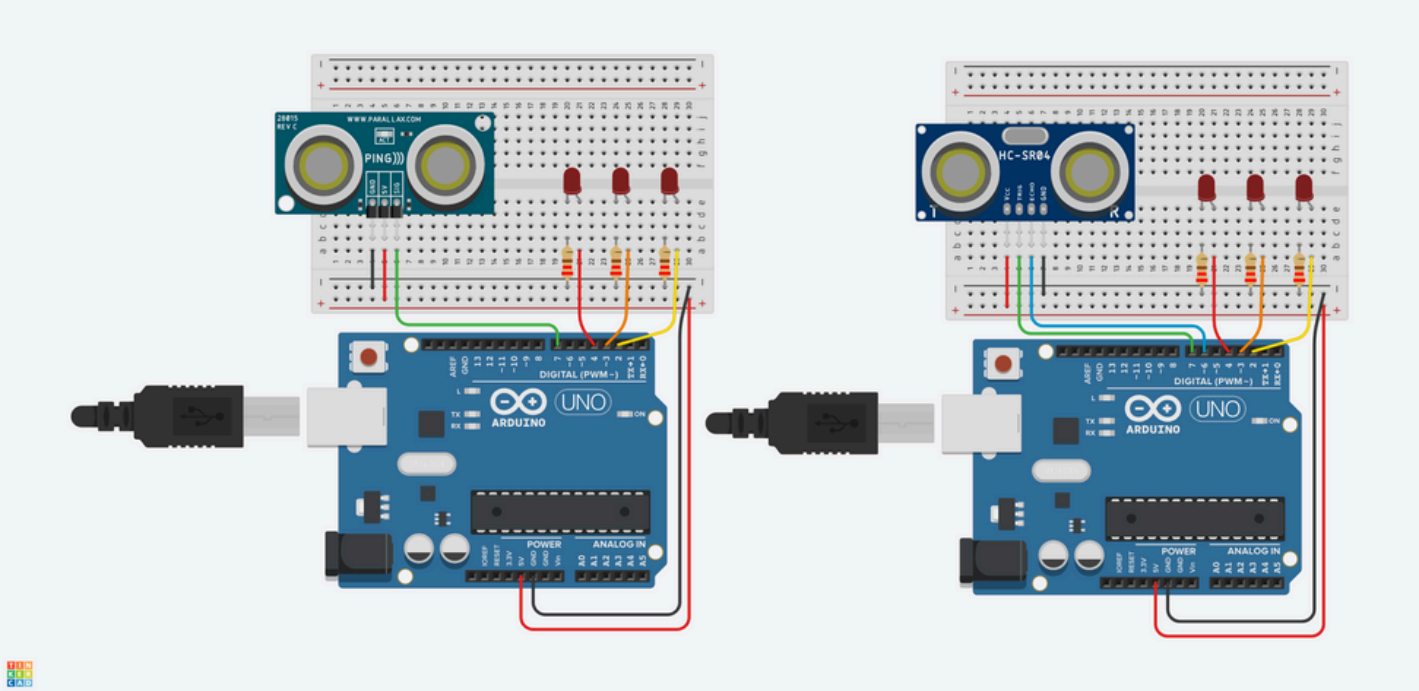
- Connect the 5 volt and ground pins on the Arduino to the power (+) and ground (-) rails on the breadboard with wires. You can change the wire colors if you want to! Either use the inspector dropdown or the number keys on your keyboard.

- Drag three LEDs on the breadboard in row E, spaced 2 breadboard sockets apart. You can change the LED color using the inspector that pops up when you click on each one.

- Use a 220 Ohm resistor to connect each LED's cathode (left leg) to the ground rail (black) of the breadboard. You can change a resistor's value by highlighting it and using the dropdown menu.

- Connect the LED anodes (right legs) to digital pins 4, 3, and 2 on the Arduino. The LED anode (+) is the terminal that current flows into. This will connect to the digital output pins on the Arduino. The cathode (-) is the terminal that current flows from. This will connect to the ground rail.

Step 2: Add Proximity Sensor



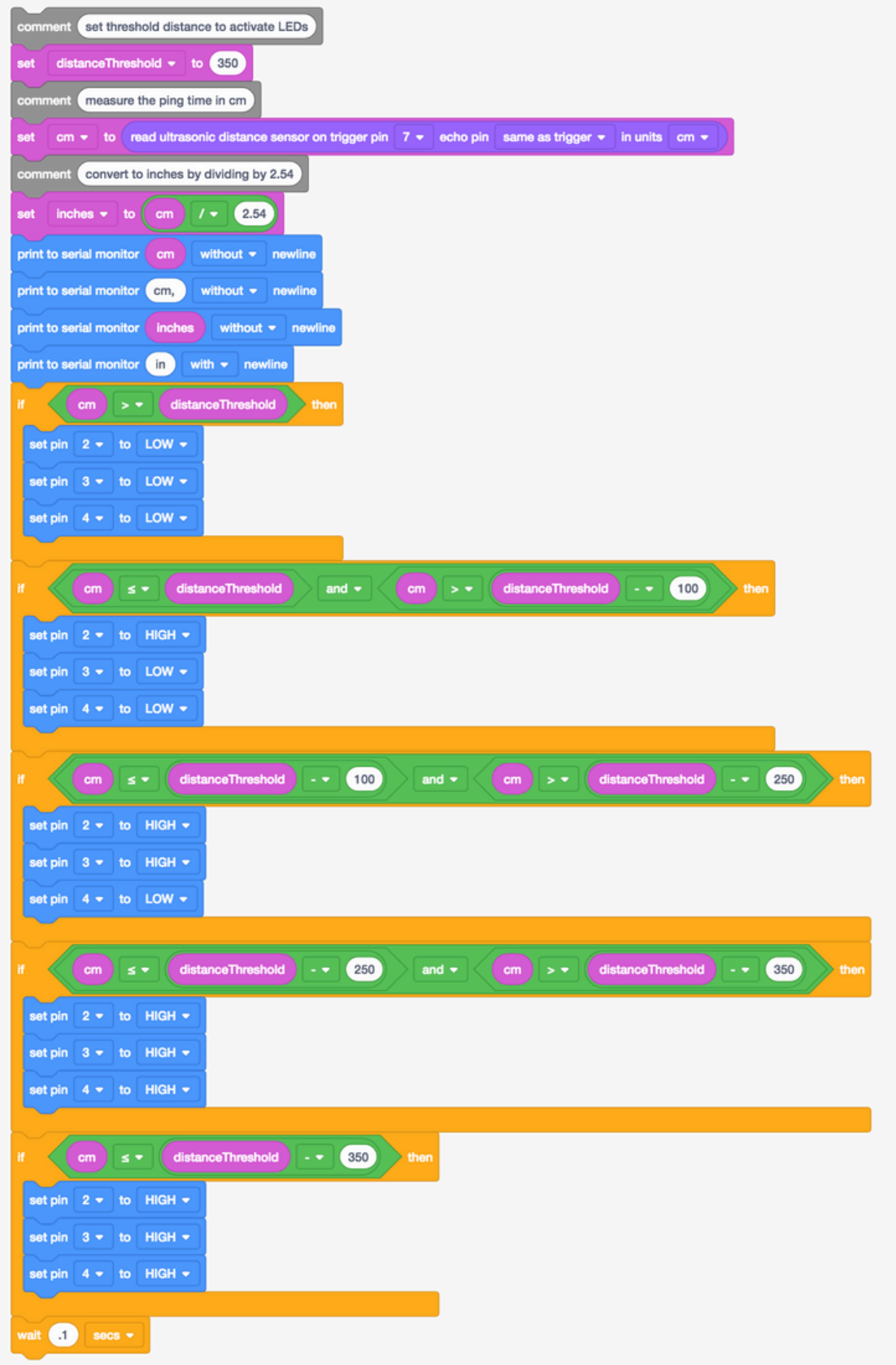
- Proximity sensors come in multiple flavors. Here in TinKercad Circuits, you can choose between a three-pin sensor or a four-pin sensor. In general, ultrasonic rangefinders have one pin that connects to ground, another that connects to 5 volts, a third for sending a signal, and a fourth for receiving a signal. The 'send' and 'receive' pins are combined into one pin on the three-pin flavor.

- In the circuits editor, find the ultrasonic rangefinder in the components drawer. To find the four-pin sensor, view "All" in the components panel (using the dropdown menu).

- Place the sensor on the breadboard to the left of the LEDs in row E, as shown in the figure.

- Wire up the sensor so the 5V pin connects to the 5V voltage rail, the GND pin connects to the ground rail, the SIG or TRIG pin to Arduino pin 7, and, if using the four-pin flavor, the ECHO pin connects to Arduino pin 6.

Step 3: Code With Blocks



- Let's use the code blocks editor to listen to the state of the sensor, then make decisions about which LEDs to light up based on the sensor's value.

- Click the "Code" button to open the code editor. The grey Notation blocks are comments for making note of what you intend for your code to do, but this text isn't required or executed as part of the program.

- Click on the Variables category in the code editor. Create a new variable called distanceThreshold and use a "set" block to set it to 350 (centimeters).

- To store the sensor value, create a variable named "cm".
- Drag out a "set" block and adjust the dropdown to our new variable cm.

- In the Input category, drag out a "read ultrasonic distance sensor on" block, and place it inside the set block.

- Adjust the dropdown menus inside the input block to set the trigger pin to 7, the echo pin to "same as trigger" and units to cm.

- Optionally create a new variable for converting centimeters to inches with a set block and an arithmetic block to read "set inches to (cm / 2.54)".

- Add some serial monitoring blocks to print out the sensor distance in centimeters and inches.

- Click the Control category and drag out an if then block, then navigate to Math and drag a comparator block onto the if block.

- In the Variables category, grab the cm variable and the distanceThreshold variable and drag them into the comparator block, adjusting the dropdown so it reads "if cm > distanceThreshold then".

- Add three digital output blocks inside the if statement to set pins 2, 3, and 4 LOW.

- Duplicate this if statement four times and add arithmetic blocks and and/or blocks to create five total state detection if statements. The first state is "the distance is farther than our threshold" so no LEDs light up. When the distance is closer than or equal to the distanceThreshold and greater than distanceThreshold-100, light up only pin 2's LED. When the temperature is between distanceThreshold-100 and distanceThreshold-250, light up two LEDs. And so on to account for all the desired states.

Congratulations!

You have learned to detect distance using an ultrasonic sensor. You could expand this project by making it a proximity alarm by adding a piezo buzzer that turns on when all three LEDs are lit up (closest distance). Consider swapping the distance sensor for a temperature motor Or add motors to create a robot with obstacle detection!