Wiring and Arduino Sketch for
Jackson Railcar VMR1001
Environmental Sound Module Pro

Parts:

- Environmental Sound Module Pro–Jackson Railcar VMR1001
- Arduino Nano (or other brand)
- HC-SR04 Ultrosonic Sensor
- G3MB-202P Solid State Relay (Level HIGH*)
- Maximum 5 Watt 8 Ohm speaker
- 9-24VDC Power Supply (Sound Module)
- 5VDC Power Supply (Nano)
- Wires and Connectors As Needed

Wiring:

HC-SR04 -> Nano

- VCC -> 5V**
- GND -> GND***
- TRIG -> D9
- ECHO -> D10

G3MB-202P -> Nano

- DC+/VCC -> 5V**
- DC-/GND -> GND***
- CH1/IN -> D7

Power Supply -> Nano

- 5VDC+ -> 5V**
- DC-/GND -> GND***

*The supplied sketch works for a “Level HIGH” relay, meaning it is closed when the voltage is set to HIGH. Some relays are “Level LOW,” meaning they are closed when the voltage is set to LOW. If using a “Level LOW” relay, make the following change to the sketch below:

- Line 12 change from: digitalWrite(RELAY_PIN, on ? HIGH : LOW); change to: digitalWrite(RELAY_PIN, on ? LOW : HIGH);

**All of the above devices should be connected to the 5V pin on the Nano. Do not use the VIN for the power supply.

***All of the above devices should be connected to the same GND pin. Do not wire power across the Nano and it can cause long-term damage to the circuitry.

Sketch:

The following Arduino sketch is for a sensor trigger using a “Level HIGH” Relay that triggers when an object is within 60 cm and plays for a minimum of 10 seconds, longer if the object is still detected. The distance and time can be adjusted in lines 7 and 8 of the sketch. Please NOTE that the line numbers are for your convenience and should NOT be copied into the sketch. The Arduino IDE will number the lines automatically.

```
1.  // ----- Pin Definitions -----
2.  const int TRIG_PIN  = 9;
3.  const int ECHO_PIN  = 10;
4.  const int RELAY_PIN = 7;   // Active-HIGH relay input
5.
6.  // ----- Settings -----
7.  const unsigned int DETECT_CM = 60;
8.  const unsigned long HOLD_MS  = 10000;
9.
10.     // Active-HIGH relay: HIGH = ON, LOW = OFF
11. void setRelay(bool on) {
12.       digitalWrite(RELAY_PIN, on ? HIGH : LOW);
13.     }
14.
15.     unsigned long lastDetectedMs = 0;
16.     bool relayOn = false;
17.
18.     // Take a single ultrasonic reading
19.     long readDistanceCm() {
20.       digitalWrite(TRIG_PIN, LOW);
21.       delayMicroseconds(2);
22.       digitalWrite(TRIG_PIN, HIGH);
23.       delayMicroseconds(10);
24.       digitalWrite(TRIG_PIN, LOW);
25.
26.       unsigned long duration = pulseIn(ECHO_PIN, HIGH, 30000UL);
27.       if (duration == 0) return -1;
28.
29.       return duration / 58;  // convert to cm
30.     }
31.
32.     // Simple noise reduction (3 samples, choose closest valid)
33.     long readFilteredDistance() {
34.       long best = -1;
35.
```

```
  for (int i = 0; i < 3; i++) {
    long cm = readDistanceCm();
    if (cm > 0) {
      if (best < 0 || cm < best) best = cm;
    }
    delay(10);
  }

  return best;
}

void setup() {
  pinMode(TRIG_PIN, OUTPUT);
  pinMode(ECHO_PIN, INPUT);

  pinMode(RELAY_PIN, OUTPUT);
  setRelay(false);  // start OFF

  Serial.begin(9600);
}

void loop() {
  unsigned long now = millis();

  long distance = readFilteredDistance();
  bool detected = (distance > 0 && distance <= DETECT_CM);

  if (detected) {
    lastDetectedMs = now;

    if (!relayOn) {
      relayOn = true;
      setRelay(true);   // turn ON (HIGH)
    }
  }

  // Turn off only if no detection for HOLD_MS
  if (relayOn && (now - lastDetectedMs >= HOLD_MS)) {
    relayOn = false;
    setRelay(false);    // turn OFF (LOW)
  }

```

```
    // Debug output
    Serial.print("Distance: ");
    Serial.print(distance);
    Serial.print(" cm | Relay: ");
    Serial.println(relayOn ? "ON" : "OFF");

    delay(50);
  }
```

Best of Luck!