International Institute of Information Technology
Introduction to IoT

Lab 4 Spring 2024

Overview:

In numerous applications, Infrared (IR) sensors find extensive use for tasks like obstacle detection, distance measurement, and automation. These sensors operate by emitting infrared radiation and subsequently gauging the reflected radiation, enabling the determination of an object's distance.

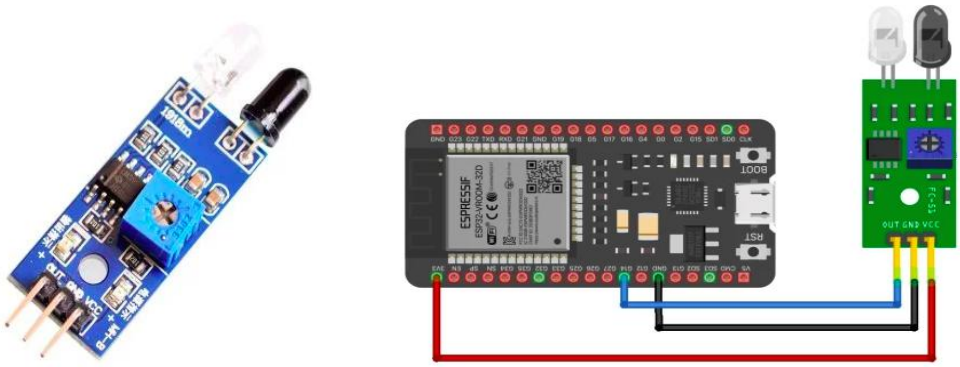
- **Problem statement:**In this laboratory experiment, the primary objective is to employ an IR sensor for detecting obstacles in the initial phase. Upon the successful identification of an obstacle, the system triggers the activation of an LED, serving as a visual indicator for the detection event. Progressing from the initial obstacle detection, the subsequent step aims to introduce a timer function, determining whether the identified object persists beyond a predefined time limit. This extended duration of detection is conveyed through a distinctive blinking pattern of the LED, contributing to a comprehensive exploration of obstacle detection and duration monitoring in the experimental setup.

IR SENSOR:

IR sensors typically consist of an infrared LED and a photodiode. The LED emits infrared radiation, and the photodiode measures the reflected radiation. By measuring the reflected radiation, the sensor can determine the distance of an object. IR sensors are widely used for obstacle detection and distance measurement applications because of their low cost, ease of use and reliability. They are typically used in applications where the object to be detected is within a certain range, typically a few centimeters to a few meters.

Object Detection: Infrared obstacle avoidance sensor can detect objects at a range of up to 30cm, although the range can vary depending on the specific sensor model.
Operating Voltage: The typical operating voltage range for infrared obstacle avoidance sensor is 3-5V DC.
Current Consumption: The current consumption of infrared obstacle avoidance sensor is generally low, typically ranging from 20-50mA.
Output Signal: Infrared obstacle avoidance sensors typically provide a digital output signal, which can be used to indicate the presence or absence of an obstacle. Some sensors may also provide an analog output signal, which can be used to determine the distance to the obstacle.
Operating Temperature: Infrared obstacle avoidance sensors can typically operate over a wide temperature range, from -20 to 60 degrees Celsius.

Detection Angle: Infrared obstacle avoidance sensors have a narrow detection angle, typically around 35-45 degrees.
Sensitivity: The sensitivity of infrared obstacle avoidance sensors can be adjusted using a potentiometer, allowing them to be tuned for optimal performance in different environments.
Interference: Infrared obstacle avoidance sensors can be affected by interference from other sources of infrared radiation, such as sunlight or fluorescent lighting. Some sensors may include features to help reduce interference, such as a shield to block out unwanted signals.



Part A: Connecting the IR sensor to the ESP32 for obstacle detection:

Connections: Connect the Vcc pin of the IR sensor to a 3.3V or 5V power source on the ESP32. Connect the GND pin of the IR sensor to the ground (GND) on the ESP32. Connect the OUT pin of the IR sensor to a digital input pin on the ESP32 (e.g., GPIO pin).

Pseudo code:
1. Define the pin for the IR sensor.
2. Initialize serial communication.
3. Set the IR sensor pin as INPUT.
4. Enter a loop that repeats indefinitely.
5. Read the digital signal from the IR sensor using digitalRead.
6. Print the output to the serial monitor (i.e. detected-HIGH, not detected-LOW).
7. Introduce a delay of 5 second.
8. Repeat from step 5.

Explanation:- The pseudocode outlines a program for interfacing an IR sensor with an ESP32 microcontroller. It begins by defining the GPIO pin

connected to the sensor's OUT pin. The program initializes serial communication for monitoring and configures the specified pin as INPUT to read the digital signal from the IR sensor. Within a continuous loop, it reads the sensor's digital output, prints the result to the serial monitor. A delay of at least 1 second is included to stabilize readings. The loop repeats indefinitely, enabling continuous monitoring of the IR sensor's output for applications such as proximity detection.

Part B: To trigger an LED to turn on when an obstacle is detected.

Connection:

1. Connect the IR sensor:
   Connect the VCC pin of the IR sensor to the 3.3V output on the ESP32.
   Connect the GND pin of the IR sensor to the GND (ground) pin on the ESP32.
   Connect the OUT (signal) pin of the IR sensor to GPIO pin on the ESP32.
2. Connect the LED:
   Connect the anode (longer lead) of the LED to a current-limiting resistor (220 ohms).
   Connect the other end of the resistor to pin 13 on the ESP32.
   Connect the cathode (shorter lead) of the LED to the GND (ground) pin on the ESP32.

Here's a textual representation of the connections: IR Sensor: VCC -> 3.3V on ESP32 GND -> GND on ESP32 OUT -> GPIO pin on ESP32 LED: Anode (longer lead) -> Resistor -> Pin 13 on ESP32 Cathode (shorter lead) -> GND on ESP32.

Pseudo code:

1. Define the GPIO pin for the IR sensor.
2. Define the GPIO pin for the LED.
3. Set up serial communication and configure the IR sensor pin as INPUT.
4. Configure the LED pin as OUTPUT.
5. In a loop: Read the digital signal from the IR sensor.
6. Print an alert message indicating obstacle detection.
7. Blink the LED if object is detected: Turn on the LED. Wait for half a second. Turn off the LED. Wait for half a second. Introduce a delay of 1 second between iterations.

Part C: Write a timer function to check if the object is continuously detected for 10 seconds, and if so, activate the LED:

Pseudo Code:

1. Define the pin for the IR sensor.
2. Define the pin for the LED.
3. Initialize a variable (lastDetectionTime) to store the time of the last detection.
4. Initialize a constant (detectionInterval) for the desired detection interval (e.g., 10 seconds).
5. In the setup function: Initialize serial communication. Set the IR sensor pin as INPUT. Set the LED pin as OUTPUT.
6. In the loop function: Read the digital signal from the IR sensor. Print the sensor value to the serial monitor.
7. Check if an object is detected: If detected, check if continuously detected for 10 seconds: If true, turn on the LED and print an alert.
8. If not continuously detected, reset the timer. If no object is detected, turn off the LED and reset the timer. Add a delay for stability (adjust as needed).

Explaination: The pseudocode outlines the logic for an object detection system using an IR sensor and an LED. It initializes variables to store the time of the last detection and sets a constant for the desired detection interval, such as 10 seconds. In a continuous loop, the code reads the digital signal from the IR sensor, prints the sensor value to the serial monitor, and checks if an object is detected. If an object is continuously detected for 10 seconds, it turns on the LED and prints an alert. If no object is detected, it turns off the LED and resets the timer. The system then adds a delay to ensure stability before repeating the process. This design allows for the monitoring of continuous object detection and provides visual feedback through the LED. Adjustments can be made based on specific hardware and project needs.

ThankYou !