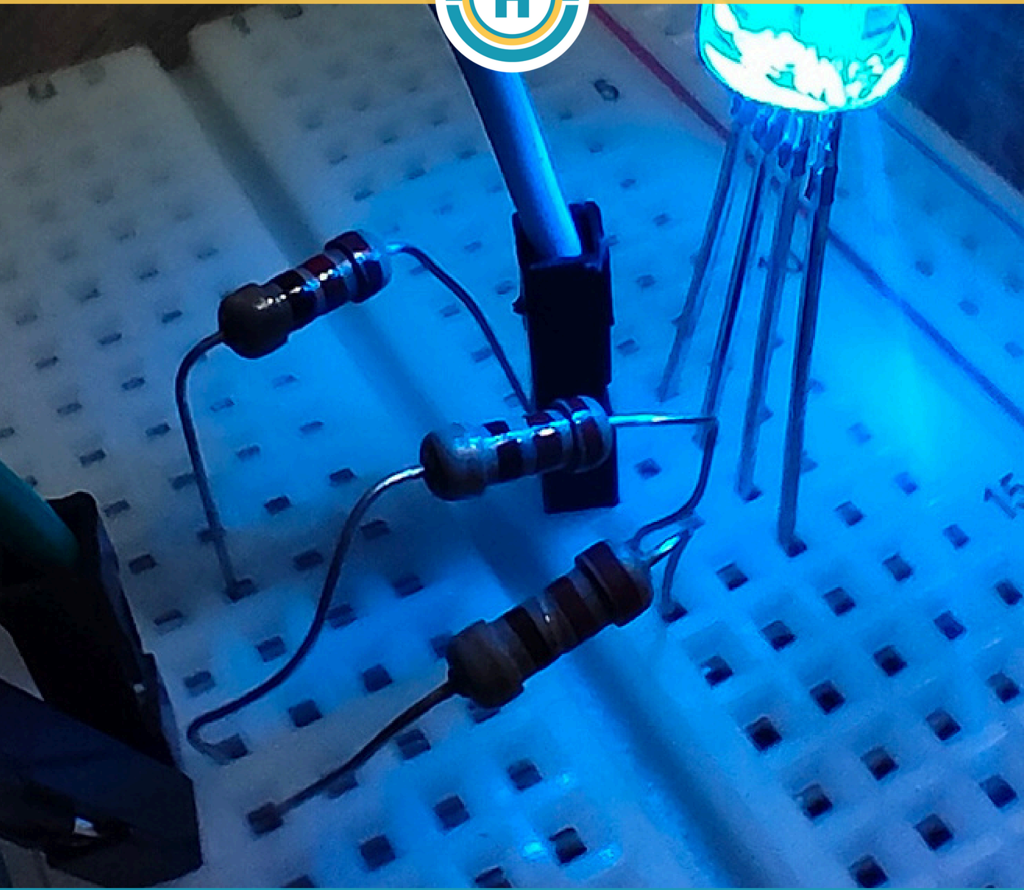
COURSE GUIDE I

ARDUINO INTRO

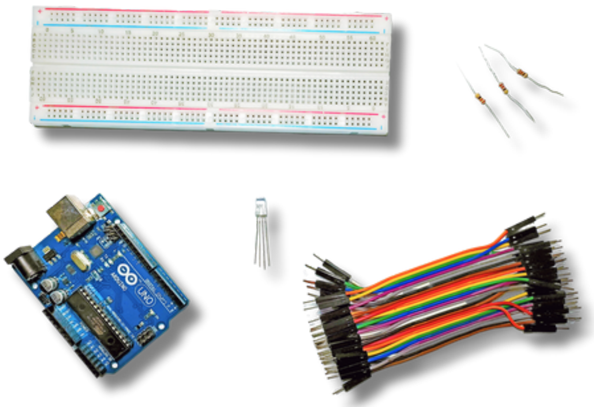
YOUR ARDUINO ADVENTURE STARTS HERE

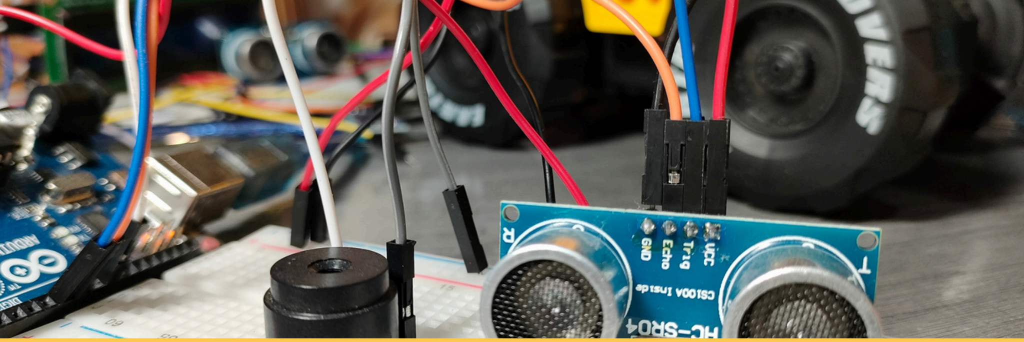


Sherwin Ramos

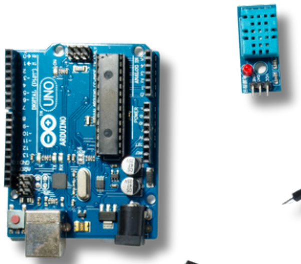
2024

Table of Contents





Hi there!



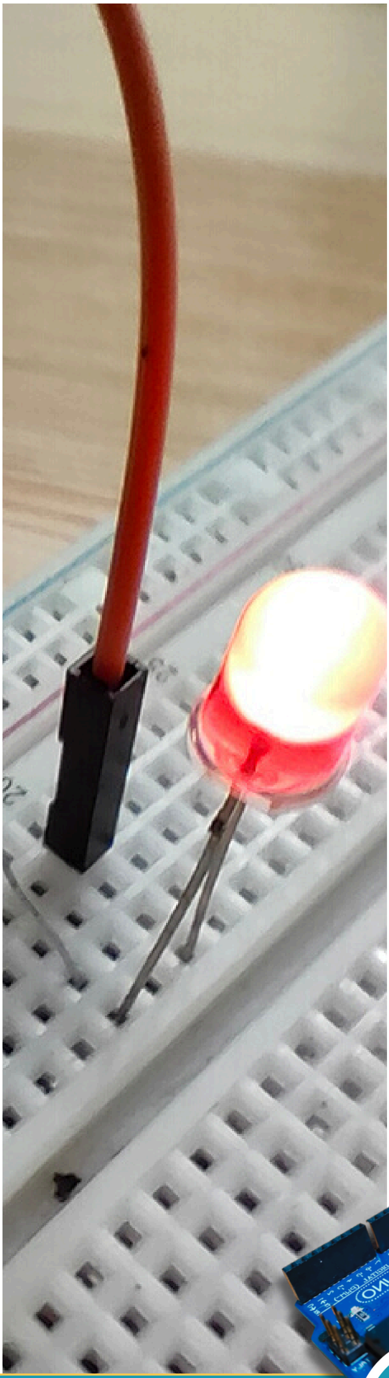
Welcome to the Arduino Intro Course Guide I!

In this comprehensive guide, you will embark on an exciting journey into the world of Arduino – a powerful platform for building and prototyping electronic projects.

Arduino enables you to bring your ideas to life, from simple LED blink projects to complex robotic systems. Throughout this course, you will learn essential concepts such as programming fundamentals, sensor interfacing, control systems, and more.

Let's learn, create, and discover what's possible with this versatile platform. Get ready to unleash your creativity and make your ideas a reality with Arduino!

Sherwin Ramos

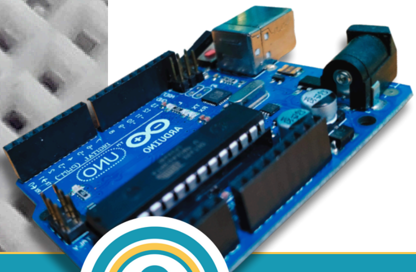


UNIT

1

INTRODUCTION TO THE ARDUINO PHYSICAL COMPUTING PLATFORM

In Unit 1, you will explore the fundamentals of the Arduino Physical Computing Platform, including identifying components, setting up the Arduino IDE, and writing basic programs. By the end of this unit, you'll have the foundational knowledge and skills to start creating your own Arduino projects.



Content Standards

The learners demonstrate an understanding of the interaction of the hardware and software components in Arduino.

By the end of this unit, students will be able to:

1. Identify Arduino Components
- Describe the main components of an Arduino board, including the microcontroller, digital and analog pins, power supply, and USB interface.
- Explain the function of each component and how they interact within the Arduino ecosystem.

2. Set Up the Arduino Environment
- Install the Arduino Integrated Development Environment (IDE) on a computer.
- Connect an Arduino board to the computer using a USB cable.
- Verify and upload basic sketches to the Arduino board.

3. Understand Basic Arduino Programming Concepts
- Define key programming concepts such as variables, data types, control structures, and functions within the context of Arduino.
- Write simple programs (sketches) to perform basic operations, such as blinking an LED.

4. Troubleshoot Basic Arduino Issues
- Identify common problems that may occur when working with Arduino and outline basic troubleshooting steps.

Performance Standards

The learners shall be able to independently make and program LED electronic circuits using Arduino.

Students will demonstrate their understanding and proficiency by:

1. Component Identification and Explanation
- Correctly identify and label all major components on an Arduino board during a practical assessment.
- Explain the role of each component in both written and verbal formats.

2. Environment Setup
- Successfully install the Arduino IDE on a computer without assistance.
- Demonstrate the ability to connect an Arduino board to the computer and upload a basic sketch, such as the "Blink" example.

3. Programming Basics
- Write and upload a sketch that uses variables, control structures, and functions to perform a specific task, such as turning an LED on and off with specific delay times.
- Submit code that follows proper syntax and formatting conventions.

4. Troubleshooting
- Identify and resolve at least two common issues that can occur during Arduino projects, such as incorrect wiring or syntax errors in the code.

UNIT TOPIC CONTENT 1

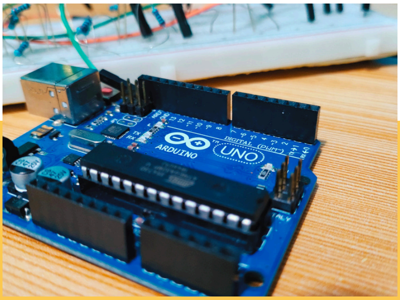
Introduction to Arduino Microcontroller

Overview

Welcome to the first topic of our Arduino journey: Introduction to Arduino Microcontroller. This topic serves as the foundational building block for understanding the Arduino platform, its components, and its capabilities. Whether you're new to electronics or looking to expand your knowledge, this topic will provide you with the essential skills and knowledge to get started with Arduino.

Learning Competencies

1. Define the Arduino Physical Computing Platform
2. Determine the parts of an Arduino microcontroller
3. Identify the features of the Arduino IDE



What is a microcontroller?

A microcontroller is a compact integrated circuit designed to govern a specific operation in an embedded system. Unlike a general-purpose computer, a microcontroller is dedicated to performing one task and running one specific application. It consists of a processor, memory, and input/output peripherals, all on a single chip.

Key Takeaways

By the end of this topic, you will have a solid understanding of what an Arduino microcontroller is, how it works, and how to set up and program your Arduino board. This foundation will prepare you for more advanced projects and explorations in subsequent units.

Resources:

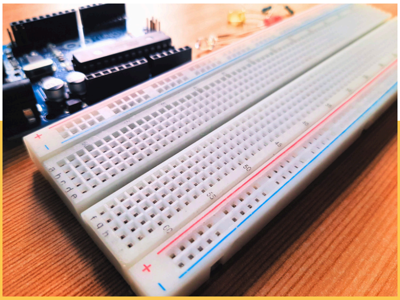
- https://tinyurl.com/bdznsvhj
- https://youtu.be/0OGZSiJfyVI

UNIT TOPIC CONTENT 2

Using the Breadboard

Overview

In this topic, we will explore an essential tool for anyone working with electronics: the breadboard. The breadboard is a versatile and reusable platform for prototyping and testing electronic circuits without the need for soldering. Understanding how to use a breadboard effectively is a critical skill for anyone embarking on Arduino projects, as it allows for quick and easy circuit assembly and modification.



What is a breadboard?

A breadboard is a rectangular plastic board with a grid of interconnected holes, which allows for the insertion of electronic components and wires to build circuits. The internal connections of a breadboard are designed to facilitate the easy and flexible creation of temporary prototypes, making it an ideal tool for experimenting with and testing different circuit designs.

Learning Competencies

1. Define a breadboard
2. Determine the proper use a breadboard
3. Identify the purpose of jumper wires

Key Takeaways

By the end of this topic, you will be comfortable using a breadboard to prototype and test various electronic circuits. You will also have a solid understanding of how to integrate breadboards into your Arduino projects, allowing you to create more sophisticated and versatile designs. This knowledge will be invaluable as you progress to more advanced topics and projects in the world of Arduino and electronics.

Resources:

https://tinyurl.com/mwmjbb4x

https://youtu.be/gGxrD3sitg4

UNIT TOPIC CONTENT 3

LED Circuits and Arduino IDE

Overview

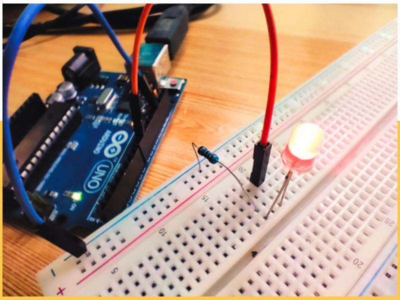
In this topic, we will delve into one of the most fundamental and exciting aspects of working with Arduino: creating LED circuits and programming them using the Arduino Integrated Development Environment (IDE). LEDs (Light Emitting Diodes) are simple yet powerful components that can be used to provide visual feedback, indicators, and even create complex light displays. This unit will guide you through the basics of LED circuits and introduce you to the Arduino IDE.

Learning Competencies

1. Define an LED
2. Explain the purpose of a resistor
3. Create an LED circuit using Arduino
4. Write an Arduino sketch to turn on and turn off an LED

Key Takeaways

By the end of this topic, you will have a solid understanding of how to create and control LED circuits using Arduino and the Arduino IDE. You will be able to write and upload your own sketches, providing a strong foundation for more complex projects and further exploration in the world of Arduino and physical computing.



What is an LED Circuit?

An LED circuit is a basic electrical circuit that uses an LED to emit light when current flows through it. LEDs are polar devices, meaning they have a positive (anode) and a negative (cathode) side, and they must be connected in the correct orientation to function. Understanding how to properly connect and control LEDs is a crucial skill for any Arduino enthusiast.

Resources:

https://tinyurl.com/34z5jzbb

https://youtu.be/VYI1lg7tt6o

UNIT TOPIC CONTENT 4

Working with More Than 1 LED

Overview

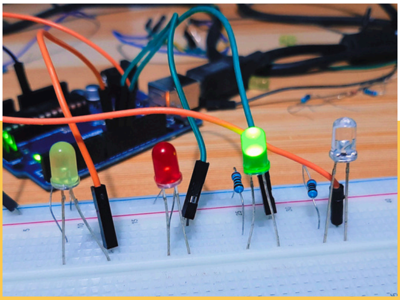
In this topic, we will expand our understanding of LED circuits by exploring how to work with multiple LEDs simultaneously. Controlling multiple LEDs opens up a wide range of possibilities for creating more complex and dynamic visual effects, from simple patterns to sophisticated light displays. This unit will guide you through the principles of managing multiple LED circuits and teach you how to program them using Arduino.

Learning Competencies

1. Create a breadboard circuit with 2 LEDs attached to different pins
2. Write an Arduino sketch/code to turn on and turn off more than 1 LED.

Key Takeaways

By the end of this topic, you will have the skills and confidence to design and program circuits with multiple LEDs, unlocking the potential for more complex and engaging Arduino projects. You will be able to apply these techniques to a variety of applications, enhancing both your hardware and software expertise in the realm of physical computing.



Why Work with Multiple LEDs?

- **Enhanced Visual Effects**: Using multiple LEDs allows for the creation of more intricate and captivating visual displays.
- **Practical Applications**: Many real-world applications, such as traffic lights, LED matrices, and status indicators, require the control of multiple LEDs.
- **Advanced Programming**: Managing multiple LEDs introduces more complex programming concepts, such as loops, arrays, and functions, enhancing your coding skills.

Resources:

https://tinyurl.com/v84eexur

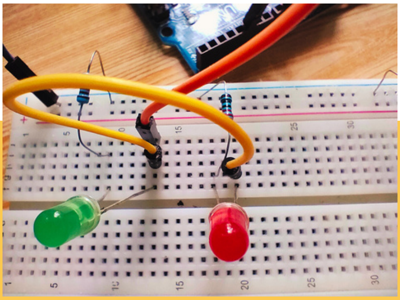
https://youtu.be/F2I4ONSIk3E

UNIT TOPIC CONTENT 5

Parallel LED Circuits

Overview

In this topic, we will delve into the specifics of creating parallel LED circuits. Parallel circuits are essential when you want to ensure that each LED operates independently of the others, receiving the same voltage but potentially different currents. Understanding how to design and implement parallel LED circuits will enhance your ability to create robust and reliable lighting solutions for various applications.



What is a Parallel LED Circuit?

A parallel LED circuit is a type of electrical circuit where multiple LEDs are connected across the same voltage source, but each LED has its own independent path. This means that each LED receives the same voltage, but the total current in the circuit is the sum of the currents through each LED. Parallel circuits are beneficial because if one LED fails, the others continue to operate normally.

Learning Competencies

1. Make a parallel connection with 2 LEDs in 1 Pin
2. Make a parallel connection with 4 LEDs and 2 Pins

Key Takeaways

By the end of this topic, you will have a comprehensive understanding of how to design, wire, and program parallel LED circuits. You will be able to create more complex and reliable lighting solutions for your Arduino projects, enhancing both your theoretical knowledge and practical skills in electronics.

Resources:

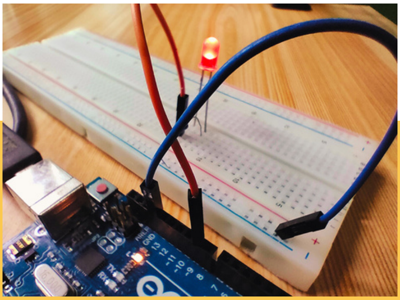
- https://tinyurl.com/5ezk9mb5
- https://youtu.be/GQtYKuk1GLs

UNIT TOPIC CONTENT 6

Fade Effect

Overview

In this topic, we will explore how to create a fade effect using Arduino. A fade effect gradually changes the brightness of an LED, creating a smooth transition that can be used for various applications, such as mood lighting, visual indicators, and artistic installations. Understanding how to program a fade effect will enhance your ability to create dynamic and visually appealing Arduino projects.



What is a Fade Effect?

A fade effect is a gradual increase or decrease in the brightness of an LED. This effect is achieved by varying the amount of current supplied to the LED over time, typically using Pulse Width Modulation (PWM). PWM allows for precise control of the LED's brightness by rapidly switching the LED on and off at different duty cycles.

Learning Competencies

1. Define a PWM pin
2. List the PWM pins in the Arduino Uno board
3. Discuss how an LED fade effect works.

Resources:

- https://tinyurl.com/bdzjz8vk
- https://tinyurl.com/42r7y54j
- https://youtu.be/unIR9nLVYWA

Key Takeaways

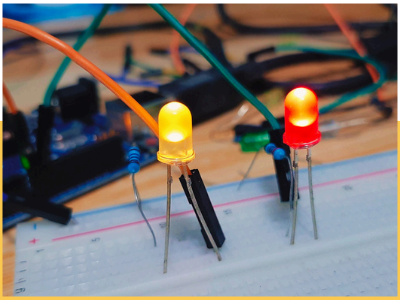
By the end of this topic, you will have a thorough understanding of how to create fade effects with Arduino, using PWM to control the brightness of LEDs. You will be able to implement smooth and dynamic lighting transitions in your projects, enhancing both the functionality and aesthetic appeal of your Arduino creations.

UNIT TOPIC CONTENT 7

Random LED Brightness

Overview

In this topic, we will explore how to create random LED brightness effects using Arduino. Random brightness effects can add an element of unpredictability and interest to your projects, making them more dynamic and engaging. This unit will guide you through the principles of generating random values in Arduino and applying them to control LED brightness.



What are Random LED Brightness Effects?

Random LED brightness effects involve changing the brightness of LEDs in a non-predictable manner. By using random values to set the brightness, you can create effects that vary in intensity and timing, leading to visually intriguing patterns and behaviors.

Learning Competencies

1. Identify the syntax of the random() and randomSeed() functions
2. Apply random brightness to more than 1 LED
3. Use random numbers to determine the brightness of an LED.

Resources:

- https://tinyurl.com/bd5az7xy
- https://youtu.be/GUy7sElmZFA
- https://youtu.be/rOgP9e-7YcA

Key Takeaways

By the end of this topic, you will have a comprehensive understanding of how to create random LED brightness effects using Arduino. You will be able to generate and control random values to produce dynamic and engaging light displays, enhancing the creativity and sophistication of your Arduino projects.

UNIT TOPIC CONTENT 8

RGB LED

Overview

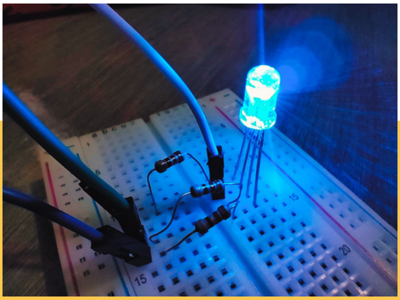
In this topic, we will dive into the world of RGB LEDs and learn how to control them using Arduino. RGB LEDs are versatile components that can emit a wide range of colors by combining red, green, and blue light at different intensities. Understanding how to work with RGB LEDs will enable you to create colorful and dynamic lighting effects for a variety of applications, from decorative displays to informative indicators.

Learning Competencies

1. Identify the 2 types of RGB LED
2. Name the pins of the RGB LED
3. Identify the correct circuit for the RGB LED.
4. Write an Arduino sketch to mix the colors of the RGB LED

Key Takeaways

By the end of this topic, you will have a thorough understanding of how to work with RGB LEDs, including wiring, programming, and creating colorful effects. You will be able to incorporate RGB LEDs into your Arduino projects, adding a new dimension of creativity and functionality.



What is an RGB LED?

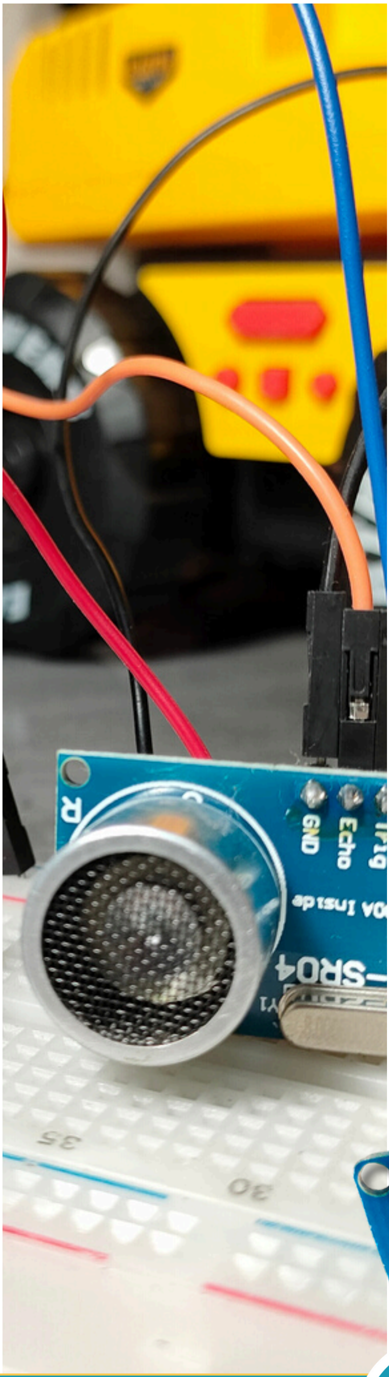
An RGB LED is a single LED package that contains three separate LEDs: one red, one green, and one blue. By varying the brightness of each individual LED, you can mix these primary colors to produce almost any color in the visible spectrum. RGB LEDs come in two main types: common anode and common cathode.

Resources:

https://tinyurl.com/59z4ztxh

https://tinyurl.com/3km7ncna

https://youtu.be/qd32N6u-mN0

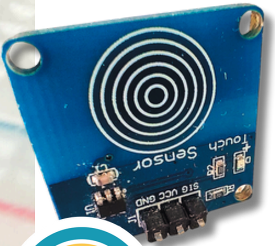


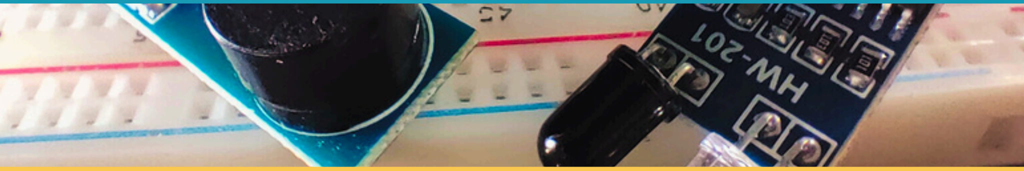
UNIT

2

INTERFACING SENSORS WITH ARDUINO

Building on the foundations from Unit 1, this Unit will teach you how to interface various sensors with Arduino, including wiring, coding, and reading data from sensors like obstacle, light, and motion sensors. By the end of this unit, you'll be able to incorporate sensor data into your projects, making them interactive and responsive.





Content Standards

The learner demonstrates an understanding of the concepts and underlying principles of various sensors.

By the end of this unit, students will be able to:

1. Identify and Describe Sensors
   - Identify various types of sensors used with Arduino, such as touch sensors, light sensors, motion sensors, and distance sensors.
   - Explain the working principles of these sensors and their applications in real-world scenarios.

2. Connect Sensors to Arduino
   - Demonstrate the correct wiring techniques for connecting analog and digital sensors to an Arduino board.
   - Use pull-up/pull-down resistors and other necessary components to ensure proper sensor function.

3. Read Sensor Data
   - Write Arduino code to read data from sensors using functions such as analogRead() and digitalRead().

4. Process and Analyze Sensor Data
   - Use the processed data to make decisions or control outputs in an Arduino project.

5. Control Outputs with Sensor Data
   - Write code to use sensor data to control outputs such as LEDs and piezo buzzer.
   - Create projects that respond dynamically to changes in the environment detected by sensors.

6. Troubleshoot Sensor Interfacing Issues
   - Identify and resolve common issues that occur when interfacing sensors with Arduino, such as incorrect wiring and signal noise.

Performance Standards

The learners shall be able to independently create a robotics application applying the functions and capabilities of sensors.

Students will demonstrate their understanding and proficiency by:

1. Sensor Identification and Explanation
   - Correctly identify and label different sensors during a practical assessment.
   - Explain the function and application of each sensor in both written and verbal formats.

2. Sensor Connection
   - Successfully connect multiple sensors to an Arduino board, ensuring correct wiring and use of additional components like resistors.
   - Demonstrate the ability to set up both analog and digital sensors without assistance.

3. Code Development for Sensor Reading
   - Write and upload Arduino sketches that accurately read data from various sensors.
   - Ensure the code uses correct functions and follows proper syntax and formatting conventions.

4. Data Processing and Output Control
   - Complete a project that involves processing sensor data and using it to control an output, such as an LED, buzzer, or motor.
   - Document the project with a detailed explanation of the code, data processing techniques, and hardware setup, including a wiring diagram.

5. Troubleshooting Skills
   - Demonstrate the ability to use the Serial Monitor to diagnose and fix issues in their sensor interfacing code.

UNIT TOPIC CONTENT 1

Pushbutton

Overview

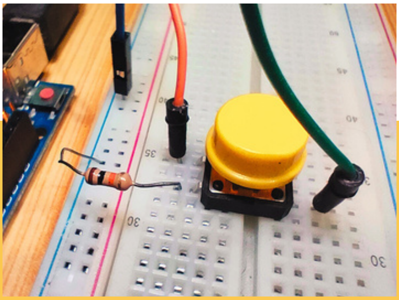
In this topic, we will explore how to use pushbuttons with Arduino to create interactive projects. Pushbuttons are simple yet powerful input devices that allow users to trigger events, control outputs, and navigate through menus in Arduino-based systems. Understanding how to interface pushbuttons with Arduino is fundamental for developing responsive and user-friendly applications.

Learning Competencies

1. Define a Pushbutton
2. Identify the correct breadboard circuit for a pushbutton or switch.
3. State the syntax of the if-else statement
4. Analyze how an if-else statement works.
5. Determine the current state of a pushbutton.
6. Create a breadboard circuit with a pushbutton and an LED
7. Write an Arduino sketch/code to turn on and turn off an LED using a pushbutton.
8. Make an LED remember its current state every time a pushbutton is pressed.

Key Takeaways

By the end of this topic, you will have a comprehensive understanding of how to use pushbuttons with Arduino, from basic wiring and code to advanced applications and troubleshooting. You will be able to create interactive and user-friendly projects that respond to pushbutton inputs, enhancing the functionality and engagement of your Arduino projects.



What is a Pushbutton?

A pushbutton is a type of switch that completes or interrupts an electrical circuit when pressed. It is commonly used as an input device in electronics to initiate actions, control devices, or provide user input.

Resources:

https://tinyurl.com/zraa9dpu

https://youtu.be/FJNqYx5EZH4

UNIT TOPIC CONTENT 2

Touch Sensor

Overview

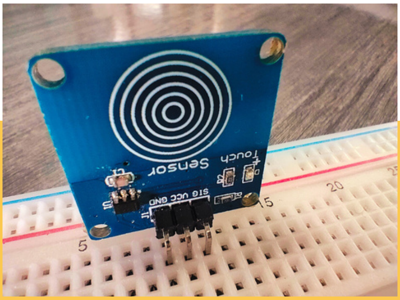
In this topic, we will explore the integration of touch sensors with Arduino to create interactive and responsive projects. Touch sensors detect touch with minimal pressure required for activation, making them ideal for applications where buttons or switches may not be practical or desired. Understanding how to interface touch sensors with Arduino opens up possibilities for creating innovative and user-friendly electronic devices.

Learning Competencies

1. Define the Touch sensor
2. Determine the pins of the Touch sensor
3. Describe the working principle of a Touch sensor
4. Identify the correct connection of the Touch sensor to the Arduino board
5. Write an Arduino sketch/code to read the value of a touch sensor
6. Control an LED using the Touch sensor
7. Create a mode functionality using the touch sensor

Key Takeaways

By the end of this topic, you will have a solid understanding of how to integrate touch sensors with Arduino, from basic wiring and programming to advanced applications and troubleshooting. You will be able to create interactive and responsive projects that leverage touch sensing technology, enhancing both functionality and user engagement in your Arduino creations.



What is a Touch Sensor?

A touch sensor is an electronic device that detects touch or proximity to a surface. Unlike mechanical switches or buttons, touch sensors do not require physical pressure to activate; they respond to capacitance changes or electrical fields induced by a human touch.

Resources:

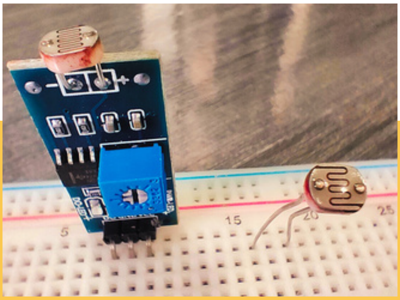
- https://tinyurl.com/mrxvuwfw
- https://tinyurl.com/4wxhfr9p
- https://youtu.be/Zo_a4FBeiwc
- https://youtu.be/4L7WwMxwZg8

UNIT TOPIC CONTENT 3

Light Sensor

Overview

In this topic, we will delve into the integration of light sensors with Arduino to create projects that respond dynamically to changes in light intensity. Light sensors, also known as photoresistors or photodiodes, detect ambient light levels and convert them into electrical signals. Understanding how to interface light sensors with Arduino allows for applications ranging from automatic lighting systems to environmental monitoring devices.



What is a Light Sensor?

A light sensor is a device that detects light and converts it into an electrical signal proportional to the intensity of light falling on it. Light sensors can vary in sensitivity, response time, and spectral range, making them suitable for different types of applications.

Learning Competencies

1. Define an LDR (Light Dependent Resistor)
2. Explain how a light sensor works
3. Make a light sensor using an LDR
4. Determine the value of a light sensor
5. Write an Arduino sketch/code to turn on and turn off an LED using a light sensor.

Key Takeaways

By the end of this topic, you will have a comprehensive understanding of how to integrate light sensors with Arduino, from basic wiring and programming to advanced applications and troubleshooting. You will be able to create projects that respond intelligently to changes in light levels, enhancing functionality and interactivity in your Arduino creations.

Resources:

https://tinyurl.com/yc3tywwv

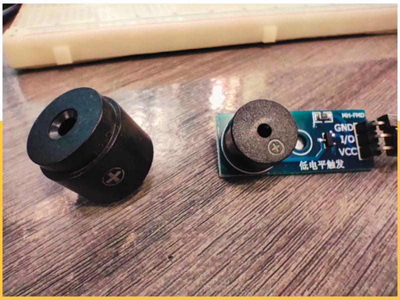
https://youtu.be/MrHDPBO9OXO

UNIT TOPIC CONTENT 4

Piezo Buzzer

Overview

In this topic, we will explore how to use piezo buzzers with Arduino to create audible alerts, tones, and melodies. Piezo buzzers are compact and versatile components that produce sound when an electrical signal is applied to them. Understanding how to interface piezo buzzers with Arduino enables you to add audio feedback and signals to your projects, enhancing their functionality and user interaction.



What is a Piezo Buzzer?

A piezo buzzer is an electronic component that generates sound through the vibration of a piezoelectric crystal when an alternating voltage is applied. It is commonly used in electronic devices for producing beeps, tones, alarms, and musical notes.

Learning Competencies

1. Define the tone() function
2. Produce sounds using a piezo buzzer

Resources:

- https://tinyurl.com/c5nnawpz
- https://tinyurl.com/mumd9s5v
- https://youtu.be/bBYEJ37lxys

Key Takeaways

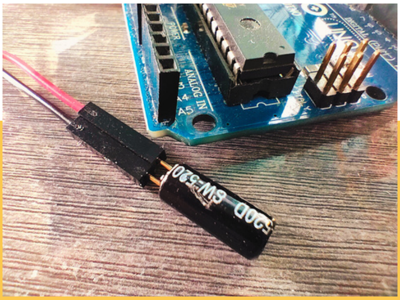
By the end of this topic, you will have a solid understanding of how to integrate piezo buzzers with Arduino, from basic wiring and tone generation to advanced applications and troubleshooting. You will be able to enhance your projects with audible feedback and interactive sound effects, making them more engaging and functional.

UNIT TOPIC CONTENT 5

Tilt Sensor

Overview

In this topic, we will explore how to integrate tilt sensors with Arduino to detect orientation changes and create responsive electronic projects. Tilt sensors, also known as tilt switches or tilt sensors, detect changes in orientation or tilt angle and can be used to trigger actions or control outputs in Arduino-based systems. Understanding how to interface tilt sensors with Arduino opens up opportunities for applications such as tilt-sensitive alarms, orientation-based controllers, and interactive devices.



What is a Tilt Sensor?

A tilt sensor is a device that detects changes in orientation or tilt angle relative to a horizontal or vertical axis. It typically consists of a mechanism that changes state (open or closed) based on its orientation, thereby indicating a tilt event.

Learning Competencies

1. Define a tilt sensor
2. Determine how a tilt sensor works
3. Write an Arduino sketch/code to read the value of a tilt sensor.
4. Write an Arduino sketch/code to turn on and turn off an LED using a tilt sensor.

Key Takeaways

By the end of this topic, you will have a comprehensive understanding of how to integrate tilt sensors with Arduino, from basic wiring and tilt detection to advanced applications and troubleshooting. You will be able to create projects that respond intelligently to changes in orientation or tilt angle, enhancing functionality and interaction in your Arduino creations.

Resources:

https://tinyurl.com/4zvcweah

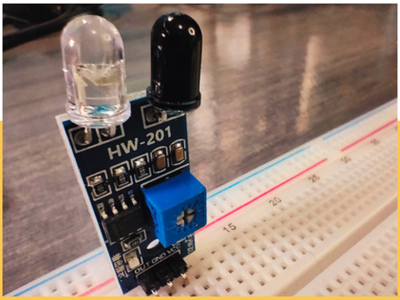
https://youtu.be/dS26GODIdSo

UNIT TOPIC CONTENT 6

Infrared Obstacle Sensor

Overview

In this topic, we will explore how to interface the HW-201 IR obstacle sensor with Arduino to detect the presence of obstacles and create responsive electronic projects. The HW-201 sensor utilizes infrared (IR) technology to detect objects within its detection range, making it ideal for applications such as obstacle avoidance in robotics, automated door systems, and proximity detection.



What is the HW-201 IR Obstacle Sensor?

The HW-201 IR obstacle sensor is a module that emits infrared light and detects reflections from nearby objects. It consists of an IR transmitter (LED) and an IR receiver (photodiode) placed side by side. When an object is within the sensor's detection range, the reflected IR light is detected by the receiver, indicating the presence of an obstacle.

Learning Competencies

1. Define Proximity Sensor
2. State the working principle of a proximity sensor.
3. Identify the proper connection of an infrared sensor to an Arduino board
4. Write an Arduino sketch/code to read the value of an infrared sensor.
5. Write an Arduino sketch/code to turn on and turn off an LED using an infrared sensor.
6. Differentiate the conditional operators in Arduino.
7. Differentiate the relational operators in Arduino
8. Combine an infrared sensor and touch sensor in 1 project.

Key Takeaways

By the end of this topic, you will have a comprehensive understanding of how to integrate the HW-201 IR obstacle sensor with Arduino, from basic wiring and sensor operation to advanced applications and troubleshooting. You will be able to create projects that effectively detect and respond to obstacles, enhancing the functionality and intelligence of your Arduino-based systems.

Resources:

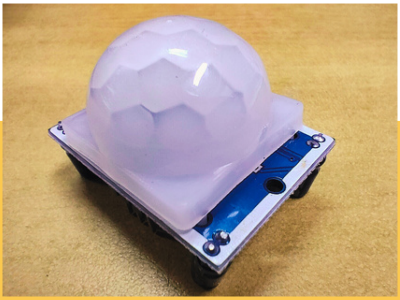
- https://tinyurl.com/yed9tep6
- https://tinyurl.com/sbs7fp7m
- https://youtu.be/Ny9lFhp9Db8
- https://youtu.be/LWeqClKtRXI

UNIT TOPIC CONTENT 7

Passive Infrared (PIR) Sensor

Overview

In this topic, we will delve into the use of Passive Infrared (PIR) sensors with Arduino to detect motion. PIR sensors are commonly used in security systems, automated lighting, and various other applications where motion detection is crucial. Integrating PIR sensors with Arduino allows for the creation of responsive and interactive systems that react to the presence of people or animals.



What is a Passive Infrared (PIR) Sensor?

A PIR sensor detects infrared radiation (IR) emitted by warm objects, such as humans and animals. It uses a pair of pyroelectric sensors to measure the differential IR radiation. When an object with heat moves in its field of view, the sensor detects this change and triggers a digital output, indicating motion.

Learning Competencies

1. Identify the Components of a PIR Sensor
2. Demonstrate the correct wiring of a PIR sensor to an Arduino board, ensuring proper power, ground, and signal connections.
3. Write Arduino code to read the digital output from the PIR sensor.
4. Develop code to use motion detection data to control devices, such as turning on an LED or activating a buzzer.

Key Takeaways

By the end of this topic, you will have a thorough understanding of how to integrate and program PIR sensors with Arduino, enabling you to create responsive and interactive systems that detect and respond to motion. You'll be equipped to develop practical applications that enhance security, convenience, and interactivity in your projects. Let's explore the capabilities of PIR sensors and bring your ideas to life with Arduino!

Resources:

https://tinyurl.com/356rx3r4

https://youtu.be/gilMy18D2lY

UNIT TOPIC CONTENT 8

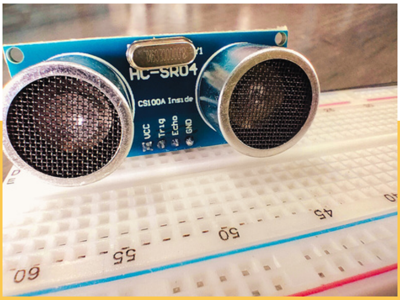
Ultrasonic Sensor

Overview

In this topic, we will explore how to interface ultrasonic sensors with Arduino to measure distance and detect objects in various applications. Ultrasonic sensors use sound waves to determine distances to nearby objects without physical contact, making them ideal for projects such as robotics, proximity detection, and distance measurement.

Learning Competencies

1. Define an ultrasonic sensor
2. State how ultrasonic sensors work
3. Identify the pins of the ultrasonic sensor
4. Measure distance in inches using an ultrasonic sensor
5. Write an Arduino sketch/code to turn on and turn off an LED using an ultrasonic sensor.



What is an Ultrasonic Sensor?

An ultrasonic sensor emits ultrasonic waves and measures the time it takes for the waves to bounce back after hitting an object. By calculating the time delay, the sensor can determine the distance to the object based on the speed of sound in the air.

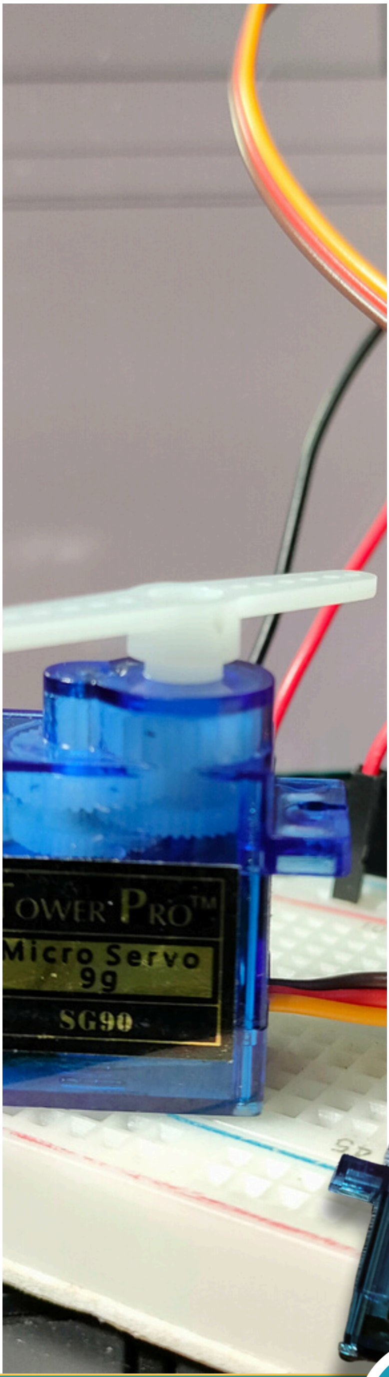
Key Takeaways

By the end of this topic, you will have a comprehensive understanding of how to integrate ultrasonic sensors with Arduino, from basic wiring and sensor operation to advanced applications and troubleshooting. You will be able to create projects that leverage ultrasonic sensing technology to enhance functionality and intelligence in your Arduino-based systems.

Resources:

https://tinyurl.com/2u83nt4e

https://youtu.be/YjuPtd9XfzA



UNIT

3

WORKING WITH ACTUATORS IN ARDUINO

Following your sensor knowledge from Unit 2, this Unit will guide you through working with actuators such as motors, servos, and LEDs. By the end of this unit, you'll be able to control actuators based on sensor inputs, creating dynamic and interactive projects.

Content Standards

The learner demonstrates an understanding of the concept and underlying principles of actuators.

By the end of this unit, students will be able to:

1. Identify Different Types of Actuators
   - Describe various types of actuators, including DC motors, servo motors, stepper motors, and solenoids.
   - Explain the operating principles and typical applications of each type of actuator.

2. Interface Actuators with Arduino
   - Connect different types of actuators to an Arduino board, including wiring, power requirements, and necessary components.
   - Explain the role of pulse-width modulation (PWM) in controlling motor speed and position.

3. Write Code to Control Actuators
   - Develop Arduino sketches to control actuators using digital and analog signals.
   - Utilize built-in Arduino libraries, such as the Servo library, to simplify actuator control.

4. Implement Actuator-Based Projects
   - Design and build projects that incorporate actuators to perform specific tasks, such as rotating a shaft, moving an object, or controlling a mechanism.
   - Integrate sensors with actuators to create responsive and interactive systems.

5. Troubleshoot Actuator Issues
   - Identify common problems when working with actuators, such as incorrect wiring, insufficient power supply, and programming errors.

Performance Standards

The learners shall be able to independently create a robotics application applying the functions and capabilities of actuators.

Students will demonstrate their understanding and proficiency by:

1. Actuator Identification and Explanation
   - Correctly identify and describe different types of actuators during a practical assessment.
   - Explain the operating principles and typical applications of each actuator type in both written and verbal formats.

2. Actuator Interfacing
   - Successfully connect a DC motor, servo motor, and stepper motor to an Arduino board, demonstrating proper wiring and power management.
   - Explain the function of any additional components used in the setup.

3. Programming Actuators
   - Write and upload sketches that control the speed and direction of a DC motor, the position of a servo motor, and the steps of a stepper motor.

4. Project Implementation
   - Complete a project that involves an actuator performing a specific task, such as a robotic arm, automated gate, or rotating display.
   - Document the project with a detailed explanation of the code, hardware setup, and operation, including a wiring diagram and flowchart.

5. Troubleshooting
   - Demonstrate the ability to diagnose and fix issues with actuator control, such as incorrect motor behavior or lack of response.

UNIT TOPIC CONTENT 1

Potentiometer

Overview

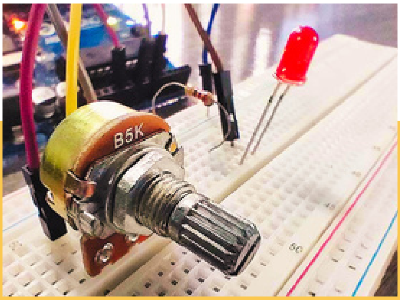
In this topic, we will explore how to use potentiometers with Arduino to read variable resistance values and create interactive projects. A potentiometer, often referred to as a "pot," is a three-terminal resistor with a sliding or rotating contact that forms an adjustable voltage divider. Understanding how to interface potentiometers with Arduino allows you to add analog input capabilities to your projects, enabling precise control and adjustments.

Learning Competencies

1. Define a potentiometer
2. Determine the correct wiring of a potentiometer
3. Describe the working principle of a potentiometer (variable resistor)
4. Write a sketch to read the value of a potentiometer
5. Explain the map() function
6. Control LEDs using a potentiometer

Key Takeaways

By the end of this topic, you will have a comprehensive understanding of how to integrate potentiometers with Arduino, from basic wiring and analog reading to advanced applications and troubleshooting. You will be able to create projects that allow for precise control and user interaction, enhancing the functionality and user experience of your Arduino-based systems.



What is a Potentiometer?

A potentiometer is an adjustable resistor used to measure and control voltage levels within an electronic circuit. It consists of a resistive element, a wiper, and three terminals: one connected to a power source, one to ground, and one to the wiper, which provides an adjustable voltage output.

Resources:

- https://tinyurl.com/5yukh383
- https://tinyurl.com/mtjn9pps
- https://youtu.be/56WObT-M8nA
- https://youtu.be/eyyUNNpzLiw

UNIT TOPIC CONTENT 2

Servo Motor

Overview

In this topic, we will explore how to interface servo motors with Arduino to achieve precise control over angular position in various applications. Servo motors are versatile actuators widely used in robotics, automation, and control systems due to their ability to position output shafts at specific angles.

Learning Competencies

1. Define an actuator
2. Define servo motor
3. Determine the correct wiring of a servo motor
4. Write an Arduino sketch/code to move a servo motor at certain angles
5. Connect 2 servo motors to an Arduino board.
6. Write a sketch to move 2 servo motors.
7. Use a for-loop to make smooth motor movements

Key Takeaways

By the end of this topic, you will have a comprehensive understanding of how to integrate servo motors with Arduino, from basic wiring and control to advanced applications and troubleshooting. You will be able to create projects that leverage the precision and versatility of servo motors, enhancing the functionality and capability of your Arduino-based systems.

What is a Servo Motor?

A servo motor is a rotary actuator that allows for precise control of angular position, velocity, and acceleration. It consists of a motor coupled to a sensor for position feedback, typically controlled by a signal that specifies a desired position. The motor then adjusts to that position, providing accurate and repeatable movement.

Resources:

- https://tinyurl.com/psr6r68h
- https://tinyurl.com/yka3eytu
- https://youtu.be/WvRlRleC6Zk
- https://youtu.be/cnXPad-emw4

UNIT TOPIC CONTENT 3

Controlling Servo Motors

Overview

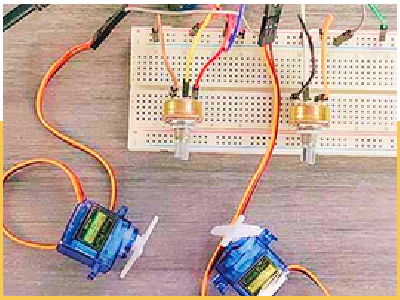
In this topic, we will delve into controlling servo motors using buttons and potentiometers with Arduino. This integration allows for interactive and user-friendly control mechanisms, enabling precise and intuitive manipulation of servo motors for various applications such as robotics, automation, and remote control systems.

Learning Competencies

1. Determine the proper wiring of a potentiometer and a servo motor.
2. Write an Arduino sketch/code to control the angles of a servo motor using a potentiometer
3. Write an Arduino sketch/code to control the angles of a servo motor using 2 potentiometers
4. Define the increment and decrement operators
5. Determine the proper wiring of 2 pushbuttons and a servo motor.
6. Write an Arduino sketch/code to control the angles of a servo motor using pushbutton switches

Key Takeaways

By the end of this topic, you will have a comprehensive understanding of how to integrate and use buttons and potentiometers to control servo motors with Arduino. You will be able to create interactive and precise control mechanisms for your projects, enhancing user experience and functionality.



Why Control Servo Motors with Buttons and Potentiometers?

- ·**Interactive Control**: Buttons and potentiometers provide an easy and intuitive way for users to interact with and control servo motors.
- ·**Precision and Adjustability**: Potentiometers allow for fine-tuning and precise adjustments, while buttons can be used for discrete control actions.
- ·**Versatile Applications**: This setup can be used in a wide range of applications, from simple control panels to complex robotic systems.

Resources:

- https://tinyurl.com/bde7d3k5
- https://youtu.be/zSf_68xA3Zs
- https://youtu.be/ko7Wb75pvnk

UNIT TOPIC CONTENT 4

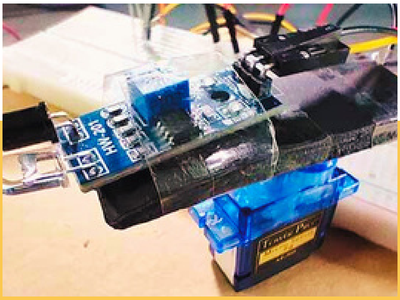
Servo Motors and Proximity Sensors

Overview

In this topic, we will explore how to combine servo motors with infrared (IR) obstacle sensors in Arduino projects to create interactive and autonomous systems. This integration allows you to design projects that respond to environmental changes by adjusting servo motor positions based on obstacle detection. Applications include obstacle-avoidance robots, automated object sorting systems, and smart navigation devices.

Learning Competencies

1. Identify the correct wiring of an infrared sensor and a servo motor
2. Write an Arduino sketch/code to control the angles of a servo motor using an infrared sensor



Why Combine Servo Motors with IR Obstacle Sensors?

- **Autonomous Navigation**: By integrating IR sensors with servo motors, you can create systems that automatically navigate around obstacles, making them suitable for robotic applications.
- **Interactive Systems**: This combination enables the creation of interactive projects where the position of the servo motor is adjusted based on real-time environmental feedback.

Key Takeaways

By the end of this topic, you will have a comprehensive understanding of how to combine servo motors with IR obstacle sensors in Arduino projects. You will be able to design, build, and troubleshoot systems that use real-time obstacle detection to control servo motor movements, enhancing the capabilities of your Arduino-based projects.

Resources:

https://tinyurl.com/5n6hznja

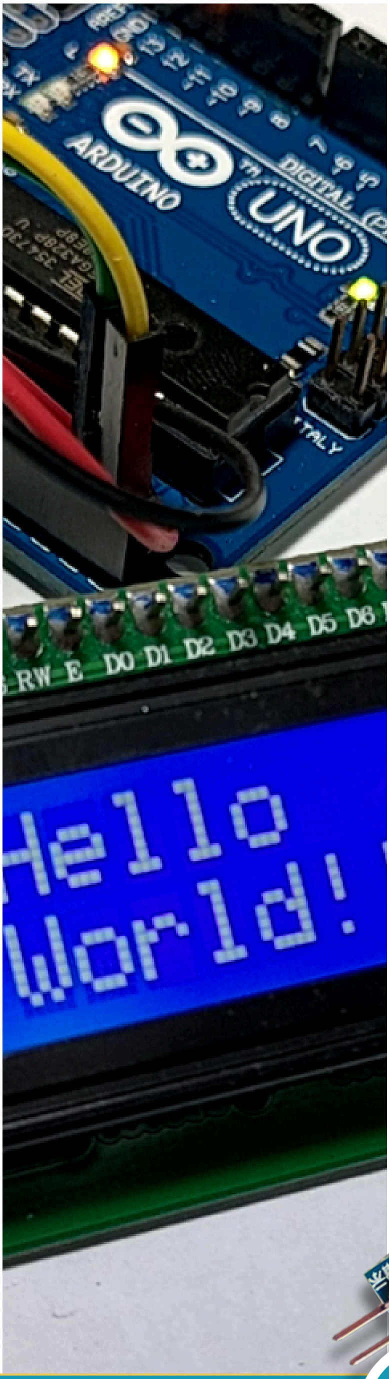
https://youtu.be/9SRZ-vKrS4c

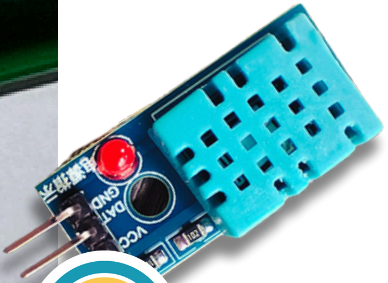
UNIT

4

DISPLAY AND MONITORING SYSTEMS

Expanding on your actuator skills from Unit 3, this Unit will teach you how to use display and monitoring systems like I2C LCDs to visualize data from sensors and actuators. By the end of this unit, you'll be able to create projects that effectively present real-time information for enhanced user interaction.





Content Standards

The learner demonstrates an understanding of the concept and underlying principles of Liquid Crystal Displays (LCD).

By the end of this unit, students will be able to:

1. Understand the Basics of Environmental Sensors
   - Identify different types of environmental sensors, such as temperature and humidity sensors (e.g., DHT11), water level sensors, and soil moisture sensors.
   - Explain the operating principles of each type of sensor.

2. Interface Environmental Sensors with Arduino
   - Connect various environmental sensors to an Arduino board, including wiring, power requirements, and signal connections.
   - Explain the function of each sensor within the context of an environment monitoring system.

3. Read Sensor Data with Arduino
   - Write Arduino code to read data from temperature and humidity sensors, water level sensors, and soil moisture sensors.
   - Utilize libraries (e.g., DHT library for temperature and humidity sensors) to simplify sensor data acquisition.

4. Display Sensor Data on an LCD
   - Connect and configure an LCD (e.g., 16x2 LCD) with an Arduino board.
   - Write Arduino code to display sensor data on an LCD, including formatting and updating the display.

5. Implement Environment Monitoring Systems
   - Design and build environment monitoring systems that read data from multiple sensors and display the information on an LCD.
   - Integrate multiple sensors to provide comprehensive environmental monitoring.

6. Troubleshoot Sensor and Display Issues
   - Identify common problems that may occur when working with sensors and LCDs, such as incorrect wiring, calibration issues, and display errors.

Performance Standards

The learners shall be able to independently create an environment monitoring system applying the functions and capabilities of a Liquid Crystal Display.

Students will demonstrate their understanding and proficiency by:

1. Sensor Identification and Explanation
   - Correctly identify and describe different types of environmental sensors during a practical assessment.
   - Explain the operating principles and typical applications of each sensor type in both written and verbal formats.

2. Sensor Interfacing
   - Successfully connect temperature and humidity sensors, water level sensors, and soil moisture sensors to an Arduino board, demonstrating proper wiring and power management.

3. Reading and Displaying Sensor Data
   - Write and upload sketches that read data from environmental sensors and display the information on an LCD.

4. Project Implementation
   - Complete a project that involves an environment monitoring system, where multiple sensors are used to gather data and display it on an LCD.
   - Document the project with a detailed explanation of the code, hardware setup, and operation, including a wiring diagram and flowchart.

5. Troubleshooting
   - Demonstrate the ability to diagnose and fix issues with sensor readings and LCD displays, such as incorrect values, display artifacts, and communication problems.
   - Use tools like the Serial Monitor to debug and understand sensor-related problems, providing solutions for at least two common issues.

UNIT TOPIC CONTENT 1

Liquid Crystal Display

Overview

In this topic, we will explore how to interface an I2C LCD with Arduino to create efficient and user-friendly display systems. The I2C (Inter-Integrated Circuit) protocol allows for simpler wiring and easier communication between the Arduino and the LCD, making it an ideal choice for projects requiring data display.

Learning Competencies

1. Define an I2C LCD
2. Determine the correct wiring of the I2C LCD
3. Determine the Arduino libraries to use for the I2C LCD.
4. Install the required libraries for I2C LCD
5. Identify the basic commands for the I2C LCD.
6. Write an Arduino sketch/code to display characters on the I2C LCD

Key Takeaways

By the end of this topic, you will have a comprehensive understanding of how to integrate I2C LCDs with Arduino, from basic wiring and configuration to advanced applications and troubleshooting. You will be able to create efficient and user-friendly display systems that enhance the functionality and user experience of your Arduino-based projects.

What is an I2C LCD?

An I2C LCD is an LCD module that communicates with the Arduino via the I2C protocol, which uses only two data lines (SDA and SCL) for communication. This reduces the number of pins needed for interfacing compared to a parallel LCD, making it simpler and more efficient to use in projects with limited I/O pins.

Resources:

https://tinyurl.com/3fd43zv5

https://youtu.be/4o9VM7NI9Os

UNIT TOPIC CONTENT 2

LCD Custom Characters and Scrolling Effect

Overview

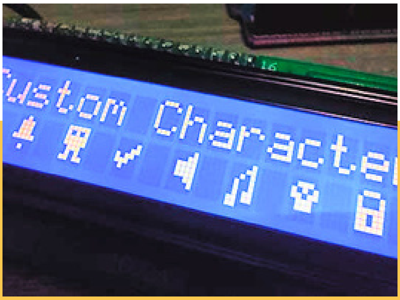
In this topic, we will explore how to create and display custom characters on an I2C LCD using Arduino. Custom characters allow you to extend the functionality of your display by adding unique symbols, icons, or graphical elements tailored to your specific project needs.

Learning Competencies

1. Identify the commands to create a custom character
2. Design a custom character using an online character generator tool.
3. Determine the commands in making a text scrolling effect
4. Explain the process of making a scrolling effect
5. Write an Arduino sketch/code to display scrolling characters on the I2C LCD.

Key Takeaways

By the end of this topic, you will have a comprehensive understanding of how to integrate I2C LCDs with Arduino, from basic wiring and configuration to advanced applications and troubleshooting. You will be able to create efficient and user-friendly display systems that enhance the functionality and user experience of your Arduino-based projects.



Why Create Custom Characters?

- ·**Enhanced Display**: Custom characters allow you to enhance the visual output of your projects by adding unique symbols, icons, or graphical elements.
- ·**Project Specificity**: Tailoring characters to your project's requirements can make the display more intuitive and user-friendly.
- ·**Improved Functionality**: Custom characters can be used to display graphical data, create simple animations, or highlight important information.

Resources:

https://tinyurl.com/ymc8pnku

UNIT TOPIC CONTENT 3

Using Pushbuttons and LEDs with LCD

Overview

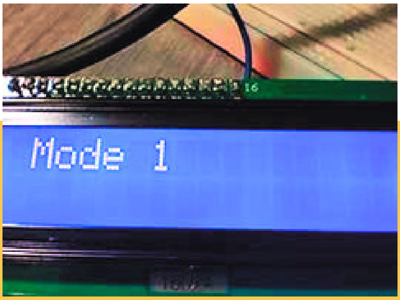
In this topic, we will explore how to integrate pushbuttons and LEDs with an I2C LCD using Arduino to create interactive and user-friendly systems. This combination allows for dynamic input and feedback mechanisms, making it ideal for projects that require user interaction and real-time display updates.

Learning Competencies

1. Make the proper circuit for the pushbutton, LEDs, and LCD
2. Monitor the status of an LED using the I2C LCD
3. Create a mode functionality using pushbuttons and LCD

Key Takeaways

By the end of this topic, you will have a comprehensive understanding of how to integrate pushbuttons, LEDs, and I2C LCDs with Arduino, from basic wiring and input/output control to advanced applications and troubleshooting. You will be able to create interactive and user-friendly systems that enhance the functionality and user experience of your Arduino-based projects.



Why Integrate Pushbuttons and LEDs with I2C LCD?

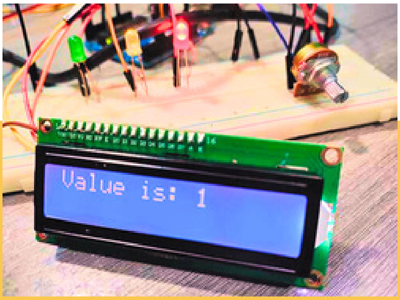
- ·**Interactive Control**: Pushbuttons provide a simple and effective way for users to interact with the system, allowing for inputs like selections, commands, or navigations.
- ·**Visual Feedback**: LEDs provide immediate visual feedback, indicating the status of operations, user inputs, or system states.
- ·**Enhanced User Experience**: Combining these elements with an I2C LCD creates a rich user interface, displaying real-time data, system statuses, and responses to user inputs.

Resources:

- https://tinyurl.com/3cxsy9uv
- https://youtu.be/_Rv8S7v6k_g

UNIT TOPIC CONTENT 4

Using Potentiometers and LEDs with LCD



Overview

In this topic, we will explore how to integrate potentiometers and LEDs with an I2C LCD using Arduino. This combination allows for dynamic control and visual feedback, making it ideal for projects that require adjustable settings, real-time monitoring, and user interaction.

Learning Competencies

1. Display the value of a potentiometer on the I2C LCD
2. Use a potentiometer to control LEDs and display their status on the screen.

Why Integrate Potentiometers and LEDs with I2C LCD?

- ·**Dynamic Control**: Potentiometers allow for adjustable settings, such as varying brightness, volume, or other analog values.
- ·**Visual Feedback**: LEDs provide immediate visual feedback, indicating changes in settings or system statuses.
- ·**Enhanced User Interface**: Combining potentiometers and LEDs with an I2C LCD creates a comprehensive user interface, displaying real-time data and feedback.

Key Takeaways

By the end of this topic, you will have a comprehensive understanding of how to integrate potentiometers, LEDs, and I2C LCDs with Arduino, from basic wiring and input/output control to advanced applications and troubleshooting. You will be able to create dynamic and interactive systems that enhance the functionality and user experience of your Arduino-based projects

Resources:

https://tinyurl.com/422md882

UNIT TOPIC CONTENT 5

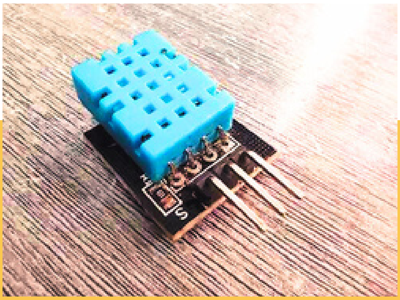
Monitoring Temperature and Humidity (DHT11)

Overview

In this topic, we will explore how to monitor temperature and humidity using the DHT11 sensor module with Arduino. The DHT11 is a widely used sensor for measuring environmental conditions, and integrating it with Arduino allows for real-time monitoring and display of these values.

Learning Competencies

1. Recognize the pin configuration of the DHT11 sensor
2. Set Up the DHT11 Sensor with Arduino
3. Program the Arduino to Read Sensor Data
4. Display Sensor Data on Serial Monitor and I2C LCD



What is the DHT11 Module?

The DHT11 is a basic, low-cost digital temperature and humidity sensor. It uses a capacitive humidity sensor and a thermistor to measure the surrounding air and provides a digital signal on the data pin. The DHT11 is easy to use but requires careful timing to capture data.

Key Takeaways

By the end of this topic, you will have a comprehensive understanding of how to monitor temperature and humidity with the DHT11 sensor and Arduino, from basic wiring and programming to advanced applications and troubleshooting. You will be able to create real-time monitoring systems that enhance the functionality and user experience of your Arduino-based projects.

Resources:

- https://tinyurl.com/4a929939
- https://tinyurl.com/p4njz4b9
- https://youtu.be/cMPjw7NAvYg

UNIT TOPIC CONTENT 6

Monitoring Water Levels with a Water Level Sensor

Overview

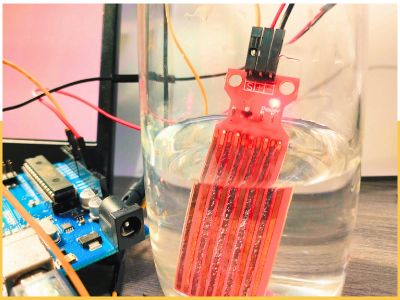
In this topic, we will explore how to monitor water levels using a water level sensor with Arduino. Water level sensors are essential for applications such as water tank monitoring, flood detection, and automated watering systems. Integrating these sensors with Arduino allows for real-time monitoring and control based on water level readings.

Learning Competencies

1. Set Up the Water Level Sensor with Arduino
2. Recognize the wiring and connections required to integrate a water level sensor with an Arduino board.
3. Write Arduino code to read water level data from the sensor.
4. Develop code to display real-time water level readings on the Arduino Serial Monitor.

Key Takeaways

By the end of this topic, you will have a comprehensive understanding of how to monitor water levels with a water level sensor and Arduino, from basic wiring and programming to advanced applications and troubleshooting. You will be able to create effective and automated water monitoring systems that enhance the functionality and efficiency of your Arduino-based projects.



What is a Water Level Sensor?

A water level sensor is a device used to detect the presence and measure the level of water within a container or environment. These sensors can be of various types, including float sensors, capacitive sensors, and resistive sensors, each suited to different applications and precision requirements.

Resources:

https://tinyurl.com/2y4tbdcc

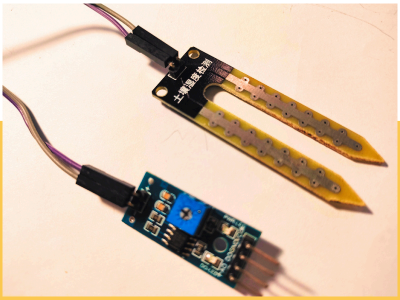
https://youtu.be/B7UIgOLMYoY

UNIT TOPIC CONTENT 7

Monitoring Soil Moisture with a Soil Moisture Sensor

Overview

In this topic, we will explore how to monitor soil moisture levels using a soil moisture sensor with Arduino. Soil moisture sensors are crucial in applications such as agricultural monitoring, automated watering systems, and environmental studies. Integrating these sensors with Arduino allows for real-time monitoring and control based on soil moisture readings.



What is a Soil Moisture Sensor?

A soil moisture sensor measures the volumetric water content in soil. It typically consists of two probes that are inserted into the soil. The sensor measures the electrical resistance or capacitance between the probes, which changes with soil moisture levels. The data is then used to determine the soil's moisture content.

Learning Competencies

1. Demonstrate the correct wiring of a soil moisture sensor to an Arduino board.
2. Write Arduino code to read soil moisture data from the sensor.
3. Develop code to display real-time soil moisture readings on the Arduino Serial Monitor.

Key Takeaways

By the end of this topic, you will have a comprehensive understanding of how to monitor soil moisture with a soil moisture sensor and Arduino, from basic wiring and programming to advanced applications and troubleshooting. You will be able to create effective and automated soil monitoring systems that enhance the functionality and efficiency of your Arduino-based projects.

Resources:

https://tinyurl.com/bdembh3z

https://youtu.be/C3NGTrF3lbI

Congratulations on completing this Arduino Intro course guide! Throughout this journey, you've delved into the exciting world of Arduino and learned how to harness its power for creating innovative projects.

Keep pushing the boundaries of what's possible with Arduino. Continue to collaborate, share your knowledge, and inspire others with your creations.

As you continue to explore and experiment with Arduino, remember that learning is a continuous process. Each project you undertake brings new challenges and opportunities for growth.

Thank you for joining me on this Arduino adventure. I wish you the best in your future projects and endeavors.

Happy making!

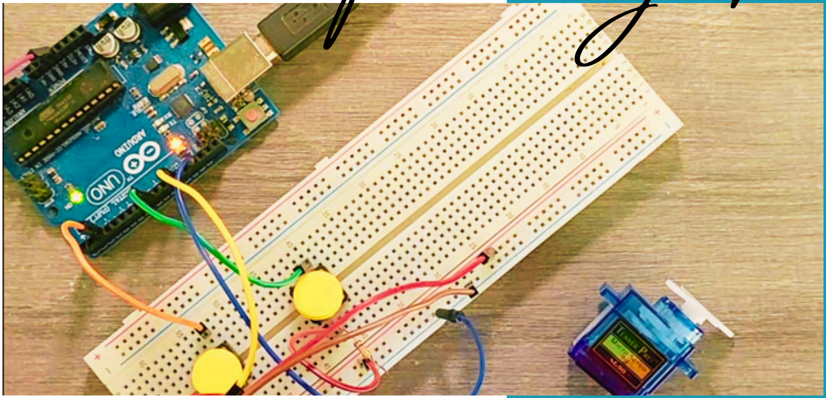
Sherwin Ramos
Computer Teacher

Xavier University Junior High School
Cagayan de Oro, Philippines

My fascination with Arduino began when I started teaching it way back in 2012 and discovered the limitless possibilities it offered for building interactive projects.

Inspired by the potential of this open-source platform, I embarked on a personal journey to learn more about it and use it to create unique and meaningful experiences.

Thank You!



ARDUINO INTRO