PT. SOLUSI INTEK INDONESIA

FUNDAMENTAL OF ELECTRONICS

BASIC ELECTRONICS TUTORIAL #01

COMPILED BY: M. LATIEF & TEAM

I2180622082

DEPARTMENT OF RESEARCH AND DEVELOPMENT

MECHATRONICS DIVISION

MAY 2024

TABLE OF CONTENTS

LIST OF EQUATIONS

LIST OF TABLES

LIST OF FIGURES

MODULE 1

PHYSICS LAWS FOR ELECTRICITY

I. PHYSICS LAWS FOR ELECTRICITY

In electronics, several fundamental laws govern the behavior of electric circuits. These laws are essential for analyzing and designing circuits. Here, we'll discuss Ohm's Law, Kirchhoff's Laws, and a few other important principles.

I.1. Ohm's Law

Ohm's Law is one of the most fundamental principles in electronics. It relates the voltage (V), current (I), and resistance (R) in a circuit.

$$V = I \times R \tag{1.1}$$

Where:

- V is the voltage across the resistor (in volts).
- I is the current flowing through the resistor (in amperes).
- R is the resistance (in ohms).

I.1.A. Applications Of Ohm's Law

- **Calculating Current**: If you know the voltage and resistance, you can find the current.

$$I = \frac{V}{R} \tag{1.2}$$

- **Calculating Voltage**: If you know the current and resistance, you can find the voltage.

$$V = I \times R \tag{1.3}$$

- **Calculating Resistance**: If you know the voltage and current, you can find the resistance.

$$R = \frac{V}{I} \tag{1.4}$$

I.2. Kirchhoff's Laws

Kirchhoff's Laws are two principles that deal with the conservation of charge and energy in electrical circuits.

I.2.A. Kirchhoff's Current Law (Kcl)

KCL states that the total current entering a junction (or node) in a circuit equals the total current leaving the junction. This law is based on the principle of conservation of electric charge.

$$\sum I_{in} = \sum I_{out} \tag{1.5}$$

I.2.B. Kirchhoff's Voltage Law (Kvl)

KVL states that the sum of all electrical voltages around any closed loop in a circuit is equal to zero. This law is based on the principle of conservation of energy.

$$\sum V = 0 \tag{1.6}$$

I.2.C. Applications Of Kirchhoff's Laws

- **Analyzing Complex Circuits**: KCL and KVL are used to solve circuits with multiple loops and nodes.
- **Finding Unknown Values**: These laws help in finding unknown currents and voltages in a circuit.

I.3. Thevenin's And Norton's Theorems

These theorems simplify complex circuits to make analysis easier.

I.3.A. Thevenin's Theorem

Thevenin's Theorem states that any linear electrical network with voltage and current sources and resistances can be replaced by an equivalent circuit consisting of a single voltage source (Thevenin voltage, V_{th}) in series with a resistance (Thevenin resistance, R_{th}).

I.3.B. Norton's Theorem

Norton's Theorem states that any linear electrical network can be replaced by an equivalent circuit consisting of a single current source (Norton current, I_N) in parallel with a resistance (Norton resistance, R_N).

I.3.C. Conversion Between Thevenin And Norton

- **Resistance Conversion**

$$R_{th} = R_N \tag{1.7}$$

- **Voltage Conversion**

$$V_{th} = I_N \times R_N \tag{1.8}$$

- **Current Conversion**

$$I_N = \frac{V_{th}}{R_{th}} \tag{1.9}$$

I.3.D. Applications

- **Simplifying Analysis**: These theorems make it easier to analyze complex circuits by reducing them to simpler equivalent circuits.
- **Design and Testing**: Useful in designing and testing circuit components by focusing on essential characteristics.

I.4. Superposition Theorem

The Superposition Theorem states that in a linear circuit with multiple independent sources, the response (voltage or current) at any point in the circuit can be found by summing the responses caused by each independent source acting alone, with all other independent sources turned off (replaced by their internal impedances).

I.4.A. Applications

- **Analyzing Circuits with Multiple Sources**: This theorem is particularly useful for circuits with multiple voltage and current sources.
- **Simplifying Complex Calculations**: Allows for easier calculation of circuit behavior by considering one source at a time.

I.5. Maximum Power Transfer Theorem

The Maximum Power Transfer Theorem states that maximum power is delivered to a load when the load resistance (R_L) is equal to the Thevenin resistance (R_{th}) of the circuit supplying the power.

I.5.A. Application

- **Optimizing Power Delivery**: Used in designing power systems to ensure maximum efficiency.

I.6. Summary of Physics Laws for Electricity

Understanding these fundamental laws and theorems is crucial for anyone studying electronics. They provide the basis for analyzing, designing, and optimizing electrical circuits. Here is a quick recap:

1. **Ohm's Law**: Relates voltage, current, and resistance.
2. **Kirchhoff's Laws**: KCL (current conservation at nodes) and KVL (voltage conservation in loops).
3. **Thevenin's and Norton's Theorems**: Simplify complex circuits to single-source equivalents.
4. **Superposition Theorem**: Analyzes circuits with multiple sources by considering one source at a time.
5. **Maximum Power Transfer Theorem**: Ensures maximum power delivery to a load when load resistance equals source resistance.

MODULE 2

ELECTRONICS COMPONENTS

II. ELECTRONICS COMPONENTS

In electronics, components are broadly classified into two categories: passive components and active components. Understanding the distinction between these types and their functions is fundamental for anyone learning about electronics.

II.1. Passive Components

Passive components are electronic components that do not require an external power source to operate. They cannot amplify or generate electrical signals but can influence the flow of electrical currents and voltages in a circuit.

II.1.A. Resistors

- **Symbol**:

Figure 2. 1 (Resistor)

- **Function**: Resist the flow of electric current, thereby controlling the voltage and current in a circuit.
- **Equation**:

$$R = \rho \frac{L}{A} \qquad (2.1)$$

  - Where: R = resistance, ρ = resistivity coefficient, L = length, A = cross section area.

- **Unit**: Ohms (Ω)
- **Common Types**: Fixed resistors, variable resistors (potentiometers), and thermistors.

II.1.B. Capacitors

- **Symbol**:

Figure 2. 2 (Capacitor)

- **Function**: Store and release electrical energy, filter signals, and manage power supply decoupling.

- **Equation**:

$$C = \varepsilon \frac{A}{d} \tag{2.2}$$

  - Where: *C*= capacitance, ε = resistivity coefficient, *A* = area of plate, *d* = distance between plates.

- **Unit**: Farads (F)

- **Common Types**: Ceramic, electrolytic, tantalum, and film capacitors.

II.1.C. Inductors

- **Symbol**:

Figure 2. 3 (Inductor)

- **Function**: Store energy in a magnetic field when electrical current flows through them, used in filters and transformers.

- **Equation**:

$$L = N^2 \frac{\mu A}{I} \tag{2.3}$$

$$\mu = \mu_r \mu_0 \tag{2.4}$$

  - Where: *L*= inductance of a coil, μ = <u>absolute</u> permeability of core material, μ_r = <u>relative</u> permeability (dimensionless), μ_0 = permeability of free space, *A* = area of plate, *d* = distance between plates.

- **Unit**: Henrys (H)

- **Common Types**: Air-core, iron-core, and ferrite-core inductors.

II.1.D. Transformers

- **Symbol**:

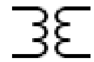

Figure 2. 4 (Transformer)

- **Function**: Transfer electrical energy between two or more circuits through electromagnetic induction.
- **Equation**:

$$\frac{N_P}{N_S} = \frac{V_P}{V_S} = \frac{I_S}{I_P} \tag{2. 5}$$

    - Where: *N* = number of turns, *V* = voltage, *I* = current, *P* = primary, *S* = secondary.

- **Unit**: N/A (but specified in terms of turns ratio, voltage, and current ratings)
- **Common Types**: Step-up, step-down, and isolation transformers.

II.2. Active Components

Active components are electronic components that require an external power source to operate and can amplify or switch electrical signals.

II.2.A. Diodes (Active Use)

- **Symbol**:

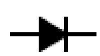

Figure 2. 5 (Diode)

- **Function**: In active circuits, Zener diodes are used for voltage regulation, LEDs for illumination and indicators, and photodiodes for light detection.
- **Common Types**: Zener diodes, light-emitting diodes (LEDs), and photodiodes.

II.2.B. Transistors

- **Symbol**:
    - NPN Transistor:

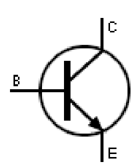

Figure 2. 6 (NPN Transistor)

- PNP Transistor:

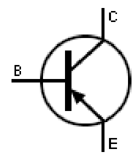

Figure 2. 7 (PNP Transistor)

- **Function**: Act as switches or amplifiers in circuits.
- **Unit**: N/A (but specified by current gain, maximum current, and voltage ratings)
- **Common Types**: Bipolar junction transistors (BJTs), field-effect transistors (FETs), and metal-oxide-semiconductor FETs (MOSFETs).

II.2.B.i. Bipolar Junction Transistors (BJT)

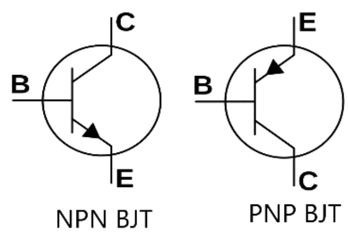

Figure 2. 8 (Bipolar Junction Transistors)

Structure: BJTs consist of three semiconductor regions: the emitter, the base, and the collector. There are two types of BJTs: NPN and PNP. In an NPN transistor, the emitter is made of N-type material, the base is P-type, and the collector is N-type. In a PNP transistor, the polarities are reversed.

Working Principle: The operation of a BJT relies on the movement of charge carriers (electrons or holes) across the transistor's semiconductor layers. When a small current flows from the base to the emitter in an NPN transistor (or from emitter to base in a PNP transistor), it controls a much larger current flowing from the collector to the emitter (or from emitter to collector in a PNP transistor). This amplification process forms the basis of BJT operation.

Modes of Operation: BJTs operate in three modes: cut-off, active, and saturation. In the cut-off region, there is no base current, and the transistor is effectively off. In the active region, the transistor operates as an amplifier, with a linear relationship between the base current and the collector current. In the saturation region, the transistor is fully on, and the collector current is at its maximum.

Applications: BJTs are widely used in analog circuits, such as amplifiers and switches. They are also employed in digital logic circuits, where they serve as the building blocks of integrated circuits (ICs).

II.2.B.ii. Field-Effect Transistors (FET)

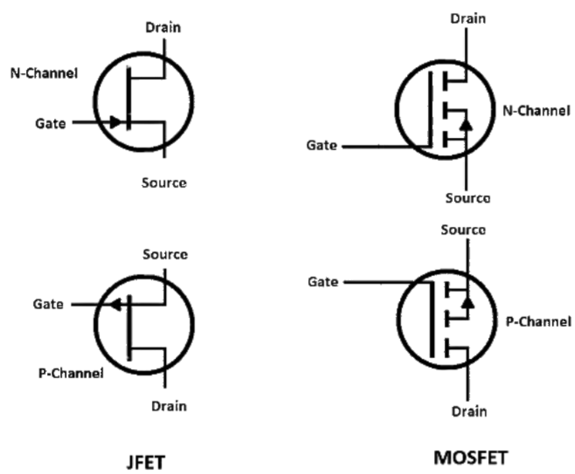

Figure 2. 9 (Field-Effect Transistors)

Structure: FETs consist of three terminals: the source, the gate, and the drain. Unlike BJTs, FETs do not require a current to flow through the gate terminal; instead, they control the flow of current between the source and drain terminals using an electric field.

Working Principle: The operation of an FET is based on the modulation of the conductivity of a semiconductor channel between the source and drain terminals by the voltage applied to the gate terminal. There are two main types of FETs: Junction FETs (JFETs) and Metal-Oxide-Semiconductor FETs (MOSFETs).

Junction FETs (JFETs): JFETs are controlled by the voltage applied across the PN junction formed between the gate and the channel. When a voltage is applied to the gate, it modulates the width of the depletion region, thereby controlling the flow of current between the source and drain terminals.

MOSFETs: MOSFETs are the most commonly used type of FETs in modern electronics. They consist of a metal gate separated from the semiconductor channel by a thin insulating layer (usually silicon dioxide). MOSFETs can be either enhancement-mode or depletion-mode devices, depending on whether they require a positive or negative voltage at the gate to turn on.

Applications: FETs are used in a wide range of applications, including amplifiers, switches, voltage regulators, and digital circuits. MOSFETs, in particular, are ubiquitous in integrated circuits (ICs) and are essential for the operation of modern microprocessors, memory chips, and other digital devices.

II.2.B.iii. BJT X FET

Table 2. 1

Feature	Bipolar Junction Transistors (BJTs)	Field-Effect Transistors (FETs)
Structure	Three semiconductor regions: emitter, base, collector	Three terminals: source, gate, drain
Two types: NPN and PNP	Two main types: Junction FETs (JFETs) and Metal-Oxide-Semiconductor FETs (MOSFETs)	
Working Principle	Controlled by current flow	Controlled by voltage at the gate terminal
Modes of Operation	Cut-off, active, saturation	Various modes based on enhancement or depletion
Control Mechanism	Current flow through base-emitter junction	Electric field modulating conductivity of semiconductor channel
Voltage vs. Current Control	Current control	Voltage control
Switching Speed	Slower compared to FETs	Generally faster than BJTs
Input Impedance	Moderate	High
Output Impedance	Moderate	High
Applications	Widely used in analog circuits, such as amplifiers and switches. Also used in digital logic circuits.	Widely used in amplifiers, switches, voltage regulators, digital circuits, and integrated circuits (ICs). MOSFETs are particularly prevalent in ICs.

II.2.C. Integrated Circuits (ICs)

- **Symbol**:

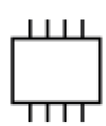

Figure 2. 10 (Integrated Circuit)

- **Function**: Combine multiple electronic components into a single package to perform complex functions.

- **Unit**: N/A (but specified by function, pin conFigureation, and electrical characteristics)
- **Common Types**: Operational amplifiers (op-amps), microcontrollers, microprocessors, and digital logic ICs.

II.2.D. Operational Amplifiers (Op-Amps)

- **Function**: Amplify voltage signals, used in signal conditioning, filtering, and mathematical operations.
- **Symbol**:

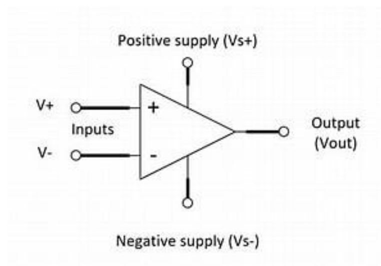

Figure 2. 11 (Operational Amplifier)

- **Unit**: N/A (but specified by gain, input impedance, and bandwidth)
- **Common Types**: Inverting, non-inverting, differential, and integrator amplifiers.

II.3. Key Differences Between Passive And Active Components

Table 2. 2

Feature	Passive Components	Active Components
Power Requirement	Do not require external power	Require an external power source to function
Functionality	Cannot amplify signals; only attenuate, store, or dissipate energy	Can amplify signals and control the flow of current
Signal Interaction	Generally linear and time-invariant	Can be nonlinear and time-variant
Examples	Resistors, capacitors, inductors, transformers	Active diodes (e.g., LEDs, Zener diodes), transistors, integrated circuits (ICs), op-amps,

II.4. Summary of Electronics Components

II.4.A. Passive Components

- **Resistors**: Control current and voltage.
- **Capacitors**: Store and release electrical energy.
- **Inductors**: Store energy in a magnetic field.
- **Transformers**: Transfer energy between circuits.

II.4.B. Active Components

- **Diodes (Active Use)**: Used in regulation, illumination, and detection.
- **Transistors**: Switch or amplify signals.
- **Integrated Circuits (ICs)**: Perform complex functions.
- **Operational Amplifiers (Op-Amps)**: Amplify voltage signals.

MODULE 3

MOTORS

III. MOTORS

Motors are essential components in many electronic devices, converting electrical energy into mechanical motion. We'll discuss the basics of DC motors, AC motors, and how to control them in circuits.

III.1. DC Motors

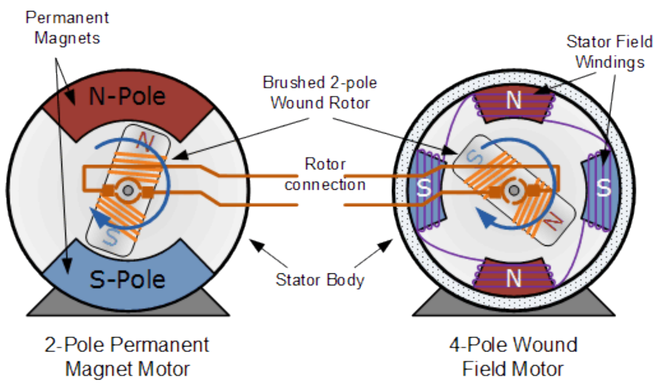

Figure 3. 1 (Conventional DC Motors)

DC motors run on direct current (DC) electricity. They are commonly used in applications where speed and direction control are essential, such as robotics, toys, and small appliances.

III.1.A. Basic Elements

- **Stator**: The stationary part that creates a magnetic field.
- **Rotor (Armature)**: The rotating part that turns when current passes through it.
- **Commutator and Brushes**: Switch the direction of current flow through the rotor, allowing continuous rotation.

III.1.B. Basic DC Motor Control

III.1.B.i. Materials Needed

- DC Motor
- Power Supply (e.g., 9V battery)
- Switch
- Diode (for back EMF protection)
- Resistor (if needed)
- Transistor (for controlling motor speed)
- Potentiometer (for adjustable speed control)

III.1.B.ii. Basic DC Motor Circuit

1. **Power Supply**: Connect the positive terminal of the battery to one terminal of the switch.
2. **Switch**: Connect the other terminal of the switch to the positive terminal of the motor.
3. **Motor**: Connect the negative terminal of the motor to the negative terminal of the battery.
4. **Diode**: Place a diode across the motor terminals (cathode to positive, anode to negative) to protect against back EMF (electromotive force).

III.1.B.iii. Speed Control Using Transistor And Potentiometer

1. **Transistor**: Use an NPN transistor (e.g., 2N2222). Connect the emitter to the negative terminal of the battery.
2. **Motor Connection**: Connect the motor between the positive terminal of the battery and the collector of the transistor.
3. **Base Resistor**: Connect a resistor (e.g., 1kΩ) between the base of the transistor and one terminal of the potentiometer.
4. **Potentiometer**: Connect the other terminal of the potentiometer to the positive terminal of the battery. The wiper (middle terminal) of the potentiometer goes to the base resistor.

III.1.B.iv. Schematic

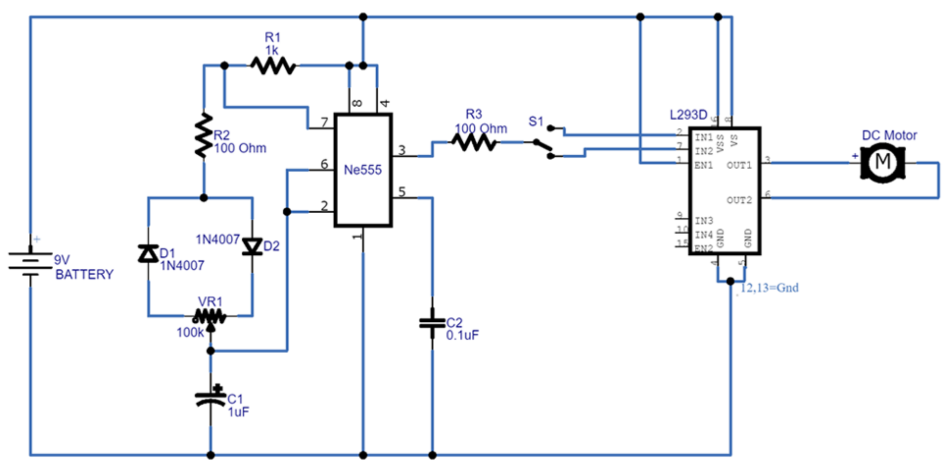

Figure 3. 2 (Schematic DC Motor)

This schematic use for DC motor speed control with PWM circuit

III.1.C. Advanced Control Methods

III.1.C.i. Pulse Width Modulation (PWM)

PWM is a common method for controlling the speed of DC motors. It involves switching the motor's power supply on and off rapidly, varying the duty cycle (the proportion of time the supply is on) to control the average voltage and, consequently, the motor speed.

III.1.C.ii. Basic PWM Circuit With Arduino

1. **Arduino**: Use an Arduino board to generate PWM signals.
2. **Transistor**: Connect the motor to the collector of an NPN transistor, with the emitter connected to ground.
3. **PWM Output**: Connect one of the Arduino PWM-capable pins (e.g., pin 9) to the base of the transistor through a base resistor.
4. **Motor Power Supply**: Connect the positive terminal of the motor to the positive power supply.

III.1.C.iii. Code Example

```
int motorPin = 9; // PWM pin connected to the transistor base

void setup() {
 pinMode(motorPin, OUTPUT);
}

void loop() {
 for (int speed = 0; speed <= 255; speed++) {
  analogWrite(motorPin, speed);
  delay(10);
 }
 for (int speed = 255; speed >= 0; speed--) {
  analogWrite(motorPin, speed);
  delay(10);
 }
}
```

III.2. AC Motors

AC motors run on alternating current (AC) electricity. They are used in high-power applications like household appliances, industrial machinery, and HVAC systems.

III.2.A. Types of AC Motors

- **Induction Motors**: Most common, no electrical connection to the rotor.
- **Synchronous Motors**: Rotor speed is synchronized with the AC frequency.

III.2.B. BASIC AC MOTOR CONTROL

Controlling AC motors typically involves more complex circuits due to the higher voltages and currents involved.

III.2.B.i. ON/OFF CONTROL WITH A RELAY

1. **Relay**: Use a relay rated for the AC voltage and current. The relay coil is controlled by a low voltage DC circuit.
2. **Switch**: The low voltage side includes a switch to control the relay.
3. **AC Load**: The relay contacts are connected in series with the AC motor and the power supply.

III.2.C. ADVANCED AC MOTOR CONTROL

III.2.C.i. SPEED CONTROL USING TRIAC

For applications requiring speed control, a TRIAC (Triode for Alternating Current) can be used in combination with a phase control circuit.

1. **TRIAC**: Acts as a switch that can be controlled by a gate signal.
2. **Diac**: Ensures triggering of the TRIAC.
3. **Phase Control Circuit**: Adjusts the firing angle of the TRIAC to control the amount of power delivered to the motor.

III.2.D. EXAMPLE AC MOTOR CONTROL SCHEMATIC

Here is basic AC Motor Control Circuit Diagram:

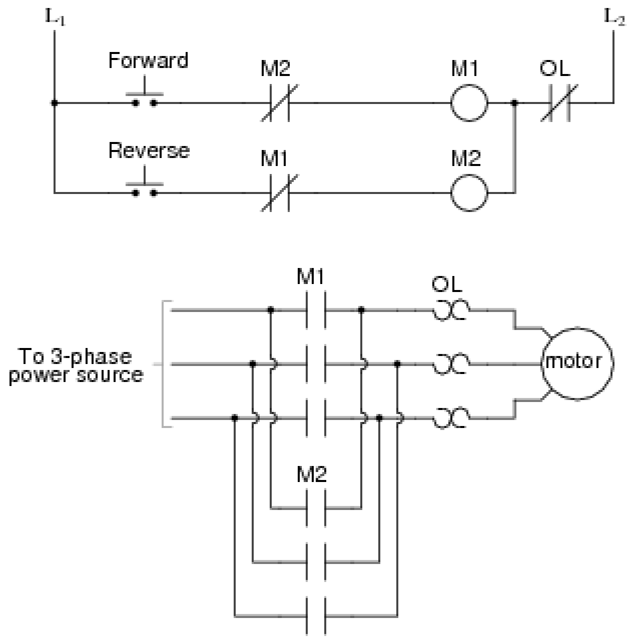

Figure 3. 3 (AC Motor Control)

III.3. Stepper Motors

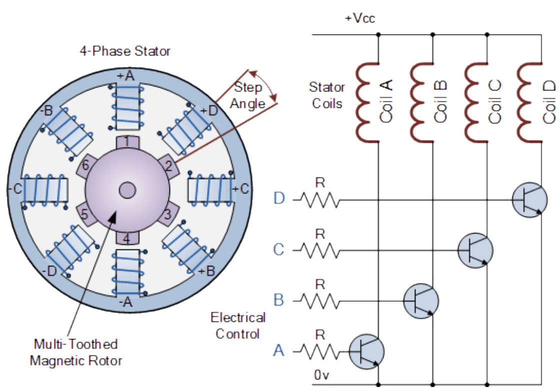

Figure 3. 4 (Stepper Motor Construction and Control)

Stepper motors are DC motors that move in discrete steps. They are used in applications requiring precise position control, such as 3D printers and CNC machines.

III.3.A. How Stepper Motors Work

A stepper motor consists of a rotor (a magnetic core) and a stator (a coil assembly). The rotor moves in discrete steps as current pulses are applied to the stator coils in a specific sequence.

III.3.B. Types Of Stepper Motors

1. **Unipolar Stepper Motors**: Have a center tap on each winding, allowing for simpler control circuits.
2. **Bipolar Stepper Motors**: Do not have a center tap, requiring more complex driving circuits but providing more torque.

III.3.C. Controlling A Stepper Motor

Stepper motors are typically controlled using a driver circuit, which sequences the current to the motor windings.

III.3.C.i. Materials Needed

- Stepper Motor
- Stepper Motor Driver (e.g., ULN2003 for unipolar, A4988 for bipolar)
- Power Supply
- Microcontroller (e.g., Arduino)
- Jumper Wires

III.3.C.ii. Basic Unipolar Stepper Motor Circuit

1. **Connect the Stepper Motor to the Driver**: Connect the motor wires to the corresponding terminals on the stepper motor driver.
2. **Power Supply**: Connect the motor power supply to the driver (e.g., 5V for the motor).
3. **Control Pins**: Connect the control pins of the driver to the microcontroller (e.g., IN1-IN4 to Arduino digital pins).

III.3.C.iii. Example Code For Unipolar Stepper Motor With Arduino

```
#include <Stepper.h>
const int stepsPerRevolution = 200; // Change this depending on the motor

// Initialize the stepper library on pins 8 through 11
Stepper myStepper(stepsPerRevolution, 8, 9, 10, 11);

void setup() {
 // Set the speed at 60 rpm
 myStepper.setSpeed(60);
 // Initialize the serial port
 Serial.begin(9600);
}

void loop() {
```

```
  // Step forward 1 revolution
  Serial.println("Forward");
  myStepper.step(stepsPerRevolution);
  delay(500);

  // Step backward 1 revolution
  Serial.println("Backward");
  myStepper.step(-stepsPerRevolution);
  delay(500);
}
```

III.3.C.iv. Stepper Motor Control Using A4988 Driver

For a bipolar stepper motor, you can use an A4988 driver for more precise control.

1. **Connect the Motor**: Connect the motor wires to the A4988 driver.
2. **Power Supply**: Connect the motor power supply and logic power supply to the driver.
3. **Control Pins**: Connect the step and direction pins of the driver to the Arduino.

III.3.C.v. Example Code For Bipolar Stepper Motor With A4988

```
const int dirPin = 2;
const int stepPin = 3;

void setup() {
  pinMode(stepPin, OUTPUT);
  pinMode(dirPin, OUTPUT);
  digitalWrite(dirPin, HIGH);
}

void loop() {
  // Rotate motor one revolution in one direction
  for(int x = 0; x < 200; x++) {
    digitalWrite(stepPin, HIGH);
    delayMicroseconds(500);
    digitalWrite(stepPin, LOW);
    delayMicroseconds(500);
  }
  delay(1000);

  // Change direction
  digitalWrite(dirPin, LOW);

  // Rotate motor one revolution in the other direction
  for(int x = 0; x < 200; x++) {
```

```
    digitalWrite(stepPin, HIGH);
    delayMicroseconds(500);
    digitalWrite(stepPin, LOW);
    delayMicroseconds(500);
  }
  delay(1000);
}
```

III.4. Servo Motors

Servo motors are DC motors with built-in feedback mechanisms to control position, speed, direction, and torque. There are two types of servo motors: 1. Angle Servo, and 2. Continuous Servo.

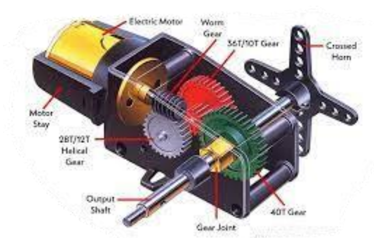

Figure 3. 5 Servo Motor

III.4.A. Angle Servo Motors

III.4.A.i. How Angle Servo Motor Work

Angle servo motors work based on the principle of feedback control. Inside the servo motor, there's a small DC motor, a set of gears, a control circuit, and a potentiometer (variable resistor). The control circuit receives commands from an external source (like a microcontroller) and adjusts the motor's position based on those commands.

When a pulse signal is sent to the servo motor, the control circuit compares the desired position (angle) with the current position obtained from the potentiometer feedback. It then drives the motor in the appropriate direction until the desired position is reached. The motor stops when the two positions match, maintaining the desired angle.

III.4.A.ii. Controlling Angle Servo Motor

To control an angle servo motor, you need to send it PWM (Pulse Width Modulation) signals. The width of the pulse determines the desired angle. Typically, a pulse width between 1 ms (for 0 degrees) to 2 ms (for 180 degrees) is used, with a frequency of around 50 Hz.

III.4.A.ii.1. Materials Needed

- Angle servo motor
- Arduino board
- Jumper wires
- Breadboard (optional)

III.4.A.ii.2. Basic Servo Motor Circuit

1. **Power Supply**: Connect the servo power wire (usually red) to the positive supply (5V), and the ground wire (usually black or brown) to the ground.
2. **Control Wire**: Connect the control wire (usually white, yellow, or orange) to a PWM-capable pin on the Arduino.

III.4.A.ii.3. Example Code For Servo Motor With Arduino

```
#include <Servo.h>

Servo myServo; // Create a servo object

void setup() {
 myServo.attach(9); // Attach the servo to pin 9
}

void loop() {
 for (int pos = 0; pos <= 180; pos += 1) { // Move from 0 to 180 degrees
  myServo.write(pos);
  delay(15); // Wait for the servo to reach the position
 }
 for (int pos = 180; pos >= 0; pos -= 1) { // Move from 180 to 0 degrees
  myServo.write(pos);
  delay(15); // Wait for the servo to reach the position
 }
}
```

III.4.B. Continous Servo Motors

III.4.B.i. How Continous Servo Motor Work

Continuous rotation servo motors are similar to angle servo motors but with modified internal mechanisms to allow continuous rotation. The control circuit inside the servo motor still receives PWM signals, but instead of positioning the motor shaft to a specific angle, it controls the speed and direction of rotation.

When the PWM signal is sent, the control circuit adjusts the speed and direction of the motor based on the pulse width. A pulse width of around 1.5 ms typically stops the motor, while shorter and longer pulses make it rotate in one direction or the other.

III.4.B.ii. Controlling Continous Servo Motor

Similar to angle servo motors, continuous rotation servo motors are controlled using PWM signals. However, instead of controlling the angle, the pulse width controls the speed and direction of rotation.

III.4.B.ii.1. Materials Needed

- Angle servo motor
- Arduino board
- Jumper wires
- Breadboard (optional)

III.4.B.ii.2. Basic Servo Motor Circuit

1. **Power Supply**: Connect the servo power wire (usually red) to the positive supply (5V), and the ground wire (usually black or brown) to the ground.
2. **Control Wire**: Connect the control wire (usually white, yellow, or orange) to a PWM-capable pin on the Arduino.

III.4.B.ii.3. Example Code For Servo Motor With Arduino

```
#include <Servo.h>

Servo myServo;  // Create a servo object

void setup() {
  myServo.attach(9);  // Attach the servo to pin 9
}

void loop() {
  myServo.writeMicroseconds(1500);  // Stop the servo
  delay(1000);                      // Wait for 1 second
  myServo.writeMicroseconds(1300);  // Rotate clockwise at slower speed
  delay(1000);                      // Wait for 1 second
  myServo.writeMicroseconds(1700);  // Rotate counterclockwise at faster speed
  delay(1000);                      // Wait for 1 second
}
```

III.5. Dc Motors With Gearboxes

A DC motor with a gearbox, also known as a geared DC motor, combines a standard DC motor with a gear reduction system. The gearbox reduces the motor's speed while increasing its torque.

Figure 3. 6 DC Motors with Gearboxes

III.5.A. How Gearboxes Work

Gearboxes consist of multiple gears that transfer motion from the motor shaft to the output shaft. The gear reduction ratio determines how much the motor's speed is reduced and how much its torque is increased.

III.5.A.i. Benefits of Using Geared Dc Motors

1. **Increased Torque**: The gearbox allows for greater torque, enabling the motor to drive heavier loads.
2. **Controlled Speed**: The reduction in speed allows for more precise control of motion, which is crucial in applications like robotics.
3. **Efficiency**: Geared motors can run more efficiently at lower speeds compared to a high-speed motor without a gearbox.

III.5.A.ii. Types Of Gearboxes

1. **Spur Gears**: Simple and cost-effective, used for moderate speed reduction.
2. **Planetary Gears**: Provide higher torque and compact size, used in high-torque applications.
3. **Worm Gears**: Provide significant speed reduction and high torque, often used in right-angle applications.

III.5.A.iii. Controlling A Geared DC Motor

III.5.A.iii.1. Materials Needed

- Geared DC Motor
- Power Supply (appropriate voltage for the motor)
- Motor Driver (e.g., L298N or H-Bridge)
- Microcontroller (e.g., Arduino)
- Potentiometer (for speed control)
- Diode (for back EMF protection)
- Jumper Wires

III.5.A.iii.2. Basic Geared DC Motor Circuit

1. **Power Supply**: Connect the positive terminal of the power supply to the motor driver's Vcc terminal and the ground to the GND terminal.
2. **Motor Connections**: Connect the motor terminals to the motor outputs on the motor driver.
3. **Control Pins**: Connect the motor driver's control pins to the Arduino (e.g., IN1, IN2 for direction control).
4. **Potentiometer**: Connect the potentiometer to the Arduino's analog input for variable speed control.

III.5.A.iii.3. Example Code For Controlling A Geared DC Motor with Arduino

```
const int motorPin1 = 3; // Motor driver input pin 1
const int motorPin2 = 4; // Motor driver input pin 2
const int enablePin = 9; // PWM pin for speed control
const int potPin = A0;   // Potentiometer input pin

void setup() {
 pinMode(motorPin1, OUTPUT);
 pinMode(motorPin2, OUTPUT);
 pinMode(enablePin, OUTPUT);
 Serial.begin(9600);
}

void loop() {
 int potValue = analogRead(potPin); // Read potentiometer value
 int speed = map(potValue, 0, 1023, 0, 255); // Map to PWM range

 // Print the speed value to the serial monitor for debugging
 Serial.print("Speed: ");
 Serial.println(speed);
```

```
  // Set motor direction
  digitalWrite(motorPin1, HIGH);
  digitalWrite(motorPin2, LOW);

  // Set motor speed
  analogWrite(enablePin, speed);

  delay(100);
}
```

III.5.A.iii.4. Practical Application: Controlling The Speed and Direction of A Geared Dc Motor

1. **Connecting the Motor Driver**: Ensure the motor driver is properly connected to the Arduino and the power supply.
2. **Setting Up the Potentiometer**: Use the potentiometer to control the speed of the motor by adjusting the voltage read by the analog pin.
3. **Running the Motor**: Upload the example code to the Arduino. Turning the potentiometer will vary the speed of the motor. Changing the digital write values can reverse the motor's direction.

III.5.A.iv. Example Application

In a robotic car, using a geared DC motor provides the necessary torque to move the car, while controlling the speed and direction using an H-Bridge driver and PWM signals from a microcontroller like an Arduino. This setup allows precise control over the car's movements, making it suitable for navigating obstacles and performing tasks that require controlled motion.

III.6. Summary of Motors

III.6.A. Uniqueness

- **DC Motors**
  - Move in continues rotation.
  - Used in applications requiring for high speed movement.
  - Controlled using a motor driver and PWM signals from a microcontroller for speed and direction control. Or, simple relay circuit for ON/OFF movement.
- **Stepper Motors**
  - Move in discrete steps, allowing precise control of position.
  - Used in applications like 3D printers, CNC machines, and robotics.
  - Controlled using stepper motor drivers and sequenced current pulses.
- **Servo Motors**
  - Provide precise control of angular position.
  - Used in applications requiring accurate positioning, like robotics and RC vehicles.

    - Controlled using PWM signals.
- **Geared DC Motors**
    - Increase torque and control speed through gear reduction.
    - Types of Gearboxes: Spur, planetary, and worm gears, each suited for different applications.
    - Controlled using a motor driver and PWM signals from a microcontroller for speed and direction control.

III.6.B. Key Takeaways

- **DC Motors**: Best for applications requiring high speed.
- **Stepper Motors**: Best for precise position control and applications needing controlled steps.
- **Servo Motors**: Best for applications needing precise angular position control with feedback.
- **Geared DC Motors:** Best for applications requiring high torque and controlled speed

MODULE 4

LOGICAL GATES AND FLIP-FLOP

IV. LOGICAL GATES AND FLIP-FLOP

IV.1. Logic Gates

Logic gates are electronic components that perform a specific logical function on one or more binary inputs to produce a single binary output. They are the building blocks of digital circuits and are used in various electronic devices, including computers, calculators, and digital watches.

IV.1.A. Basic Types of Logic Gates

IV.1.A.i. AND

- Symbol:

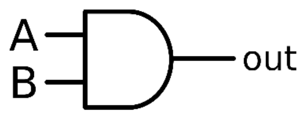

Figure 4. 1 (AND Logic Gate)

- Truth Table:
  Table 4. 1

A	B	Output
0	0	0
0	1	0
1	0	0
1	1	1

- Function: Outputs 1 only if both inputs A and B are 1.

IV.1.A.ii. OR

- Symbol:

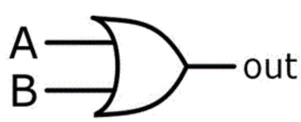

Figure 4. 2 (OR Logic Gate)

- Truth Table:
  Table 4. 2

A	B	Output
0	0	0
0	1	1
1	0	1
1	1	1

- Function: Outputs 1 if at least one of the inputs A or B is 1.

IV.1.A.iii. NOT / INVERTER

- Symbol:

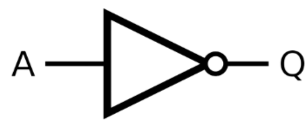

Figure 4. 3 (NOT Logic Gate)

- Truth Table:
  Table 4. 3

A	Output
0	1
1	0

- Function: Outputs the inverse of input A.

IV.1.A.iv. NAND

- Symbol:

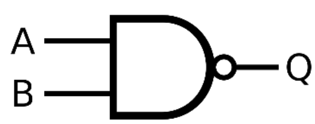

Figure 4. 4 (NAND Logic Gate)

- Truth Table:
  Table 4. 4

A	B	Output
0	0	1
0	1	1

A	B	Output
1	0	1
1	1	0

- Function: Outputs 1 if at least one of the inputs A or B is 0 (inverse of AND gate).

IV.1.A.v. NOR

- Symbol:

Figure 4. 5 (NOR Logic Gate)

- Truth Table:
  Table 4. 5

A	B	Output
0	0	1
0	1	0
1	0	0
1	1	0

- Function: Outputs 1 if both inputs A and B are 0 (inverse of OR gate).

IV.1.A.vi. XOR (Exclusive OR)

- Symbol:

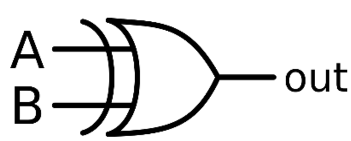

Figure 4. 6 (XOR Logic Gate)

- Truth Table:
  Table 4. 6

A	B	Output
0	0	0
0	1	1
1	0	1
1	1	0

- Function: Outputs 1 if the inputs A and B are different.

IV.1.A.vii. XNOR (EXCLUSIVE NOR)

- Symbol:

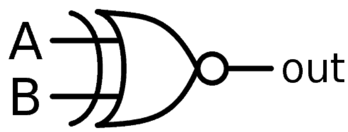

Figure 4. 7 (XNOR Logic Gate)

- Truth Table:
  Table 4. 7

A	B	Output
0	0	1
0	1	0
1	0	0
1	1	1

- Function: Outputs 1 if the inputs A and B are the same (inverse of XOR gate).S

IV.1.B. Practical Uses of Logic Gates

1. Digital Circuits: Logic gates are used to create combinational and sequential circuits that form the basis of digital systems such as computers, memory devices, and digital signal processors.
2. Arithmetic Operations: Gates are used in arithmetic logic units (ALUs) to perform operations like addition, subtraction, multiplication, and division.
3. Decision Making: Logic gates are used in control systems and decision-making circuits where certain conditions need to be met to trigger an action.
4. Data Processing: Used in encoding and decoding information, error detection, and correction in communication systems.

IV.1.C. Combinational and Sequential Logic

1. Combinational Logic: Outputs depend only on the current inputs. Examples include adders, multiplexers, and encoders.
2. Sequential Logic: Outputs depend on both current inputs and past inputs (history). Examples include flip-flops, counters, and memory units.

IV.1.D. Summary of Logic Gates

- AND Gate: Outputs 1 if both inputs are 1.
- OR Gate: Outputs 1 if at least one input is 1.

- NOT Gate: Outputs the inverse of the input.
- NAND Gate: Outputs 1 if at least one input is 0 (inverse of AND).
- NOR Gate: Outputs 1 if both inputs are 0 (inverse of OR).
- XOR Gate: Outputs 1 if inputs are different.
- XNOR Gate: Outputs 1 if inputs are the same (inverse of XOR).

IV.2. Flip-Flops

Flip-flops are bistable multivibrator circuits that have two stable states and can store one bit of data. They are used in memory elements, registers, and various types of counters in digital electronics.

IV.2.A. Types of Flip-Flops

IV.2.A.i. SR (Set-Reset) Flip-Flop

- Function: Stores a bit of data based on the set (S) and reset (R) inputs.
- Symbol:

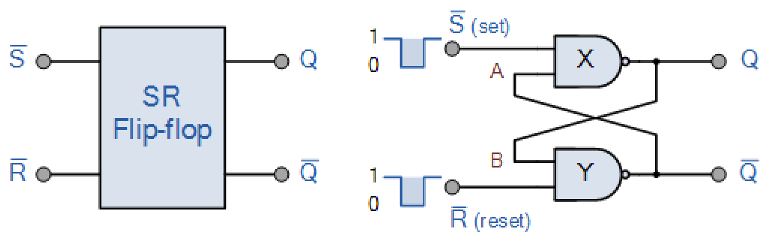

Figure 4. 8 (SR Flip-flop)

- Truth Table:

Table 4. 8

S	R	Q (next)
0	0	Q
0	1	0
1	0	1
1	1	Undefined

- Operation:
  - When S=1 and R=0, Q is set to 1.
  - When S=0 and R=1, Q is reset to 0.
  - When S=0 and R=0, Q retains its previous state.
  - When S=1 and R=1, the output is undefined (invalid state).

IV.2.A.ii. D (Data or Delay) Flip-Flop

- Function: Stores the value of the data input (D) at the rising or falling edge of the clock signal.
- Symbol:

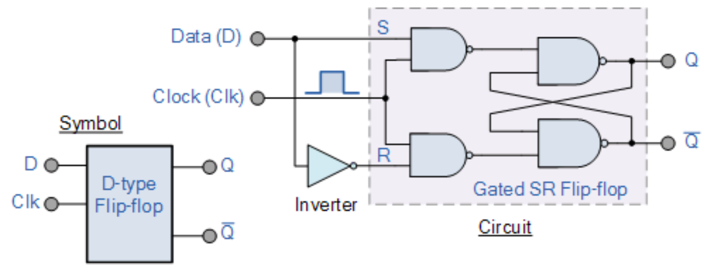

Figure 4. 9 (D Flip-flop)

- Truth Table:

Table 4. 9

D	CLK	Q (next)
0	↑	0
1	↑	1

- Operation:
  - On the rising edge (↑) of the clock signal, Q takes the value of D.
  - Otherwise, Q remains unchanged.

IV.2.A.iii. JK Flip-Flop

- Function: An enhancement of the SR flip-flop, with no invalid state.
- Symbol:

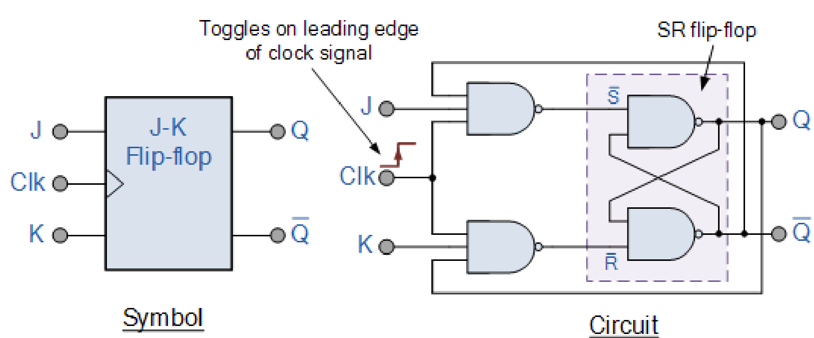

Figure 4. 10 (JK Flip-flop)

- Truth Table:

Table 4. 10

J	K	CLK	Q (next)
0	0	↑	Q
0	1	↑	0
1	0	↑	1
1	1	↑	!Q

- Operation:
  - When J=0 and K=0, Q retains its previous state.
  - When J=0 and K=1, Q is reset to 0.
  - When J=1 and K=0, Q is set to 1.
  - When J=1 and K=1, Q toggles its state.

IV.2.A.iv. T (Toggle) Flip-Flop

- Function: Toggles its state on each clock edge when the toggle input (T) is high.
- Symbol:

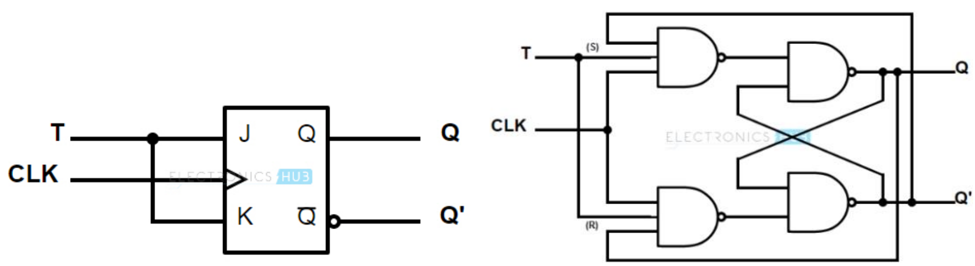

Figure 4. 11 (T Flip-flop)

- Truth Table:

Table 4. 11

T	CLK	Q (next)
0	↑	Q
1	↑	!Q

- Operation:
  - When T=1, Q toggles on the rising edge of the clock signal.
  - When T=0, Q retains its previous state.

IV.2.B. Applications of Flip-Flops

1. Data Storage: Flip-flops are used as basic memory elements to store individual bits of data.
2. Registers: Arrays of flip-flops are used to store multiple bits of data, forming registers.
3. Counters: Flip-flops are connected in series to create binary counters, which count pulses.
4. Frequency Division: Flip-flops can divide the frequency of a clock signal in digital circuits.
5. State Machines: Flip-flops are used to maintain the state of sequential logic circuits in finite state machines.

IV.2.B.i. Practical Example: Using D Flip-Flop

Materials Needed:

- D Flip-Flop IC (e.g., 74HC74)
- Breadboard
- Jumper Wires
- Power Supply (5V)
- Clock Pulse Generator (can be an Arduino or a 555 timer circuit)
- Push Buttons (for data input)

Circuit Setup:

1. Power Supply: Connect Vcc and GND of the flip-flop IC to the power supply.
2. Clock Input: Connect the clock input pin (CLK) to the clock pulse generator.
3. Data Input: Connect a push button to the data input pin (D) to manually input data.
4. Output: Connect the Q output pin to an LED for visual indication.

Example Code for Clock Pulse with Arduino

```
const int clockPin = 8;

void setup() {
  pinMode(clockPin, OUTPUT);
}

void loop() {
  digitalWrite(clockPin, HIGH);
  delay(500); // 500 ms HIGH
  digitalWrite(clockPin, LOW);
  delay(500); // 500 ms LOW
}
```

IV.2.C. Summary of Flip-Flops

- SR Flip-Flop: Set and reset states, with undefined state when both inputs are high.
- D Flip-Flop: Data is stored on the clock edge.
- JK Flip-Flop: Versatile with no invalid state; can set, reset, or toggle.
- T Flip-Flop: Toggles state on each clock edge when T is high.

Flip-flops are essential for creating memory elements, counters, and control circuits in digital systems. Understanding their operation and applications is crucial for designing complex sequential logic circuits.

MODULE 5

OPERATIONAL AMPLIFIER

V. OPERATIONAL AMPLIFIER

V.1. Overview

An operational amplifier, or op-amp, is a high-gain voltage amplifier with a differential input and usually a single-ended output. Op-amps are used in various applications such as amplification, filtering, and mathematical operations.

V.2. Basic Characteristics of Op-Amps

- **High Input Impedance**: This ensures that the op-amp draws very little current from the input signal source.
- **Low Output Impedance**: Allows the op-amp to drive heavy loads without significant loss of signal.
- **High Gain**: Op-amps have very high open-loop gain, typically in the range of 10510^{5105} to 10610^{6106}.
- **Differential Inputs**: Two inputs – inverting (−) and non-inverting (+). The output is proportional to the difference between the voltages at these inputs.
- **Single-Ended Output**: Most op-amps provide a single output voltage.

V.3. Basic Op-Amp Configurations

V.3.A.i. Inverting Amplifier

- **Circuit**:

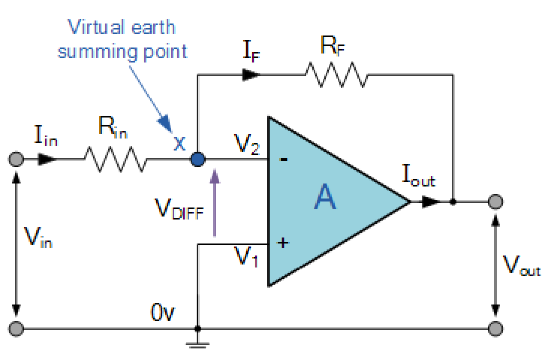

Figure 5. 1 (Inverting Amplifier)

- **Gain**:

$$Av = \frac{v_{out}}{v_{in}} = -\frac{R_f}{R_{in}} \quad (5.1)$$

- **Operation**: The input signal is applied to the inverting input through resistor R_{in}, and the output is fed back to the inverting input through feedback resistor R_f. The non-inverting input is grounded. The gain is controlled by the ratio of R_f to R_{in}.

V.3.A.ii. Non-Inverting Amplifier

- **Circuit**:

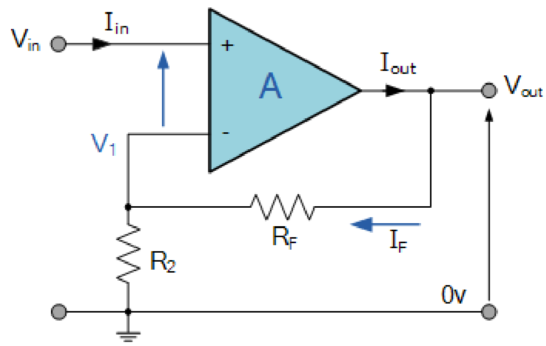

Figure 5. 2 (Non-Inverting Amplifier)

- **Gain**:

$$Av = 1 + \frac{R_f}{R_{in}} \tag{5. 2}$$

- **Operation**: The input signal is applied to the non-inverting input. The inverting input is connected to a voltage divider formed by R_{in} and R_f. The gain is determined by the values of R_f and R_{in}.

V.3.A.iii. Voltage Follower (Buffer)

- **Circuit**:

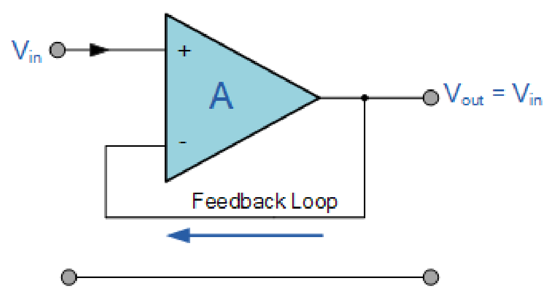

Figure 5. 3 (Buffer)

- **Gain**:

$$Av = 1 \tag{5. 3}$$

- **Operation**: The output voltage follows the input voltage (unity gain). This conFigureation is used for impedance matching and signal buffering.

V.3.A.iv. Summing Amplifier

- **Circuit**:

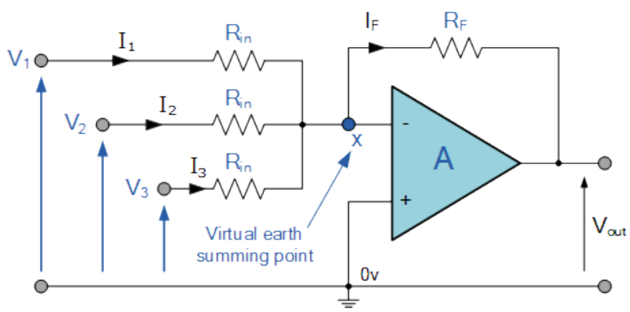

Figure 5. 4 (Summing Amplifier)

- **Output Voltage**:

$$Vout = -(\frac{R_f}{R_{in}}V_1 + \frac{R_f}{R_{in}}V_2 + \frac{R_f}{R_{in}}V_3) \tag{5. 4}$$

(5)

- **Operation**: This circuit adds multiple input voltages. Each input voltage is applied through a separate resistor, and the total gain for each input can be adjusted by the values of these resistors.

V.3.A.v. Differential Amplifier

- **Circuit**:

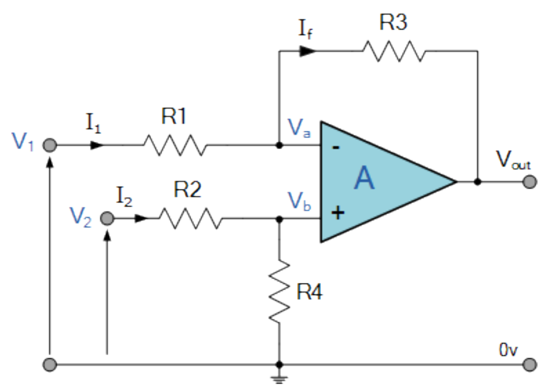

Figure 5. 5 (Differential Amplifier)

- **Output Voltage**:

$$Vout = \frac{R_3}{R_1}(V_2 - V_1) \tag{5.5}$$

- **Operation**: Amplifies the difference between two input voltages. The gain is set by the resistors R_f and R_1.

V.4. Applications of Op-Amps

1. **Amplifiers**: Op-amps are used in a variety of amplifier circuits, including inverting, non-inverting, differential, and summing amplifiers.
2. **Filters**: Op-amps are used in active filter circuits such as low-pass, high-pass, band-pass, and band-stop filters.
3. **Oscillators**: Op-amps are used in oscillator circuits to generate sine, square, and triangular waveforms.
4. **Comparators**: Op-amps can be conFigured as voltage comparators to compare two voltages and output a digital signal indicating which is larger.
5. **Integrators and Differentiators**: Op-amps can perform mathematical integration and differentiation of input signals.
6. **Signal Conditioning**: Op-amps are used to modify and condition signals, including buffering, scaling, and level shifting.

V.4.A. Example: Inverting Amplifier Circuit

- **Materials Needed**
  - Op-Amp IC (e.g., 741 or LM358)
  - Resistors (R_{in} and R_f)
  - Breadboard
  - Power Supply (±12V or ±15V)
  - Signal Generator (for input signal)
  - Oscilloscope (to view output signal)
- **Circuit Setup**
  - **Power Supply**: Connect the positive supply voltage to the op-amp's Vcc pin and the negative supply voltage to the op-amp's Vee pin.
  - **Input Signal**: Connect the input signal from the signal generator to the inverting input of the op-amp through resistor R_{in}.

- **Feedback Resistor**: Connect resistor R_f between the output and the inverting input.
- **Non-Inverting Input**: Connect the non-inverting input to the ground.
- **Output Signal**: Connect the output to the oscilloscope to observe the amplified signal.

- **Example Calculation**
  - **Given**: $R_{in} = 1k\Omega\ , Rf = 10k\Omega$
  - **Gain**: $Av\ = -\frac{R_f}{R_{in}} = -\frac{10k\Omega}{1k\Omega} = -10\ v$

If the input signal is a 1V peak sine wave, the output signal will be a 10V peak inverted sine wave.

V.5. Summary of Operational Amplifier

Operational amplifiers are versatile and essential components in analog electronics, used in various conFigureations and applications:

- **Inverting Amplifier**: Negative gain, input connected to inverting input.
- **Non-Inverting Amplifier**: Positive gain, input connected to non-inverting input.
- **Voltage Follower**: Unity gain, input follows output.
- **Summing Amplifier**: Adds multiple input signals.
- **Differential Amplifier**: Amplifies the difference between two inputs.

Understanding how to use and conFigure op-amps is crucial for designing effective analog circuits and systems.