IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

AUTOMATIC COLI WINDING MACHINE

[1]Rathod Sandip, [2]Karagir Shubham, [3]Ekhande Pratik, [4]Jawalagekar Samarth, [5]Digge Pranit

[1]Guide, [2]Student-1, [3]Student-2, [4]Student-3, [5]Student-4

[1]Electrical Engineering,

[1]Vishweshwarayya Institute Of Engineering And Technology

Dilip Nagar, Almala Tq. Ausa Dist. Latur - 413556,

Abstract

Coil winding is a critical manufacturing process in the electrical and electronics industry, widely used in the production of transformers, inductors, relays, solenoids, and motor windings. Conventional manual coil winding methods are time-consuming, labor-intensive, and prone to human errors such as inconsistent turn count and non-uniform winding tension, which directly affect product quality and performance. To overcome these limitations, this project presents the design and development of a **low-cost, microcontroller-based Automatic Coil Winding Machine** that ensures precise turn counting, improved accuracy, and enhanced operational efficiency.

Introduction

In the modern electrical and electronics industry, coil winding plays a vital role in the manufacturing of essential components such as transformers, inductors, solenoids, relays, electromagnets, and electric motors. The performance, efficiency, and reliability of these devices largely depend on the accuracy, uniformity, and consistency of the coil winding process. Traditionally, coil winding has been carried out manually or using semi-automatic machines, especially in small-scale industries and workshops. However, manual winding methods are often inefficient, time-consuming, and highly dependent on operator skill.

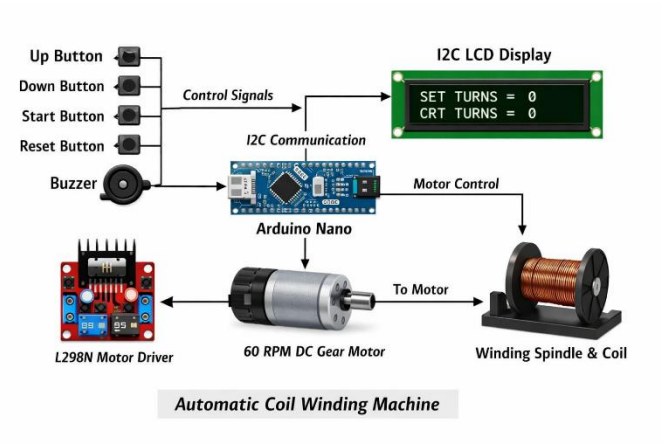
Need for Automation in Coil Winding

Manual coil winding presents several challenges, particularly when a large number of turns or consistent winding quality is required. Human operators may experience fatigue, leading to miscounting of turns, uneven winding layers, and variation in coil tension. These issues directly affect the electrical characteristics of the coil, such as inductance, resistance, and heat dissipation.

Automation in coil winding addresses these limitations by:

- Eliminating manual turn counting
- Ensuring consistent winding speed
- Improving accuracy and repeatability
- Reducing production time and labor effort
- Enhancing safety during operation

Working Principle of the



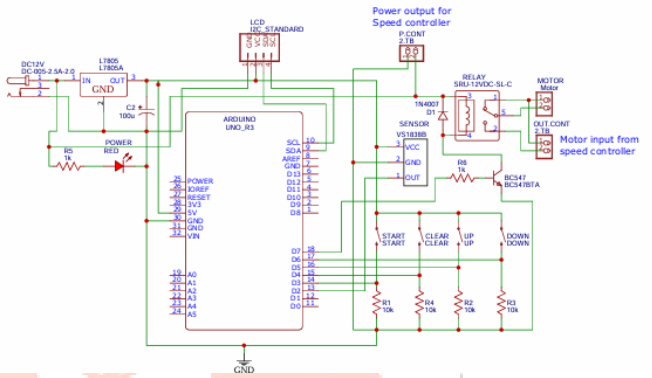
System

The block diagram represents the functional relationship between different modules of the system. The main blocks of the automatic coil winding machine are:

- Power Supply Unit
- Arduino Nano (Control Unit)
- User Input Unit (Push Buttons)
- Sensor Unit (IR Sensor)
- Motor Driver Unit (L298)
- DC Gear Motor
- Display Unit (16×2 LCD with I2C)
- Mechanical Winding Assembly

- The **12V, 2A DC adapter** supplies power to the system. The DC motor receives 12V through the motor driver, while the Arduino Nano and other control components operate at 5V.
- The **Arduino Nano** acts as the central controller. It receives input signals from the **push buttons**, which are used to set the required number of turns, start the winding process, and reset the system.
- The **IR sensor** continuously monitors the rotation of the winding shaft. Each rotation generates a pulse that is sent to the Arduino Nano. The controller counts these pulses to determine the number of completed turns.
- Based on the sensor feedback and user-defined settings, the Arduino sends control signals to the **L298 motor driver**, which drives the **12V DC gear motor**. The motor rotates the coil winding shaft through the bearing mechanism.
- The **LCD display** shows the preset number of turns and the current number of turns in real time. Once the required number of turns is completed, the Arduino automatically stops the motor.

Circuit Diagram



- Digital pins are used to interface push buttons, IR sensor, and motor driver inputs
- I2C pins (SDA and SCL) are used to connect the LCD display
- 5V and GND pins power the sensors and logic circuits
- IN1 and IN2 pins are connected to Arduino digital pins
- ENA pin is connected to 5V for continuous enable
- Motor terminals are connected to the DC gear motor
- 12V supply is provided from the adapter

Hardware Requirement

1. **Arduino Nano**

Arduino Nano is a compact microcontroller board based on the ATmega328P. It is responsible for executing the control program, reading inputs from sensors and buttons, processing logic, and controlling the motor driver.

Advantages:

- Small size
- Low power consumption
- Easy programming using Arduino IDE

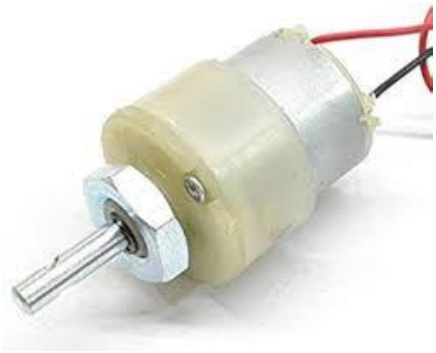
2. L298N Motor Driver

The L298N is a dual H-bridge motor driver IC capable of driving DC motors up to 2A.

Functions:

- Protects Arduino from high motor current
- Allows motor ON/OFF control
- Enables direction control

3. DC Gear Motor (12V, 60 RPM)



The DC gear motor provides controlled rotational motion for winding the coil.

Features:

- High torque at low speed
- Suitable for continuous operation
- Ideal for precise winding applications

4. IR Sensor

The IR sensor is used to detect shaft rotation and count coil turns.

Advantages:

- Non-contact sensing
- High reliability
- Low cost
- Easy interfacing

5. **LCD Display (16×2 with I2C)**

The LCD provides real-time system feedback to the user.

Displayed Information:

- Set turns
- Completed turns

6. **Push Buttons**

Push buttons allow user interaction.

Functions:

- UP: Increase number of turns
- DOWN: Decrease number of turns
- START: Begin winding process
- CLEAR: Reset count

7. **Power Supply (12V, 2A Adapter)**

The power supply provides sufficient current for both control electronics and motor operation.

8. **Bicycle Hub Bearing**

The bearing reduces friction and ensures smooth shaft rotation.

9. **Copper Winding Wire Spool**

Copper wire is wound uniformly onto the spool to form the required coil.

Safety and Reliability Considerations

- Motor driver prevents current overload
- Automatic motor stop avoids over-winding
- Stable power supply ensures reliable operation

Software Requirement

1. Software Objectives

The main objectives of the software are:

- To allow the user to set the required number of coil turns
- To read input from push buttons reliably
- To control the DC motor through the motor driver
- To count the number of rotations accurately using an IR sensor
- To display real-time information on the LCD
- To automatically stop the motor when the desired number of turns is reached
- To provide reset and safety control through software logic

2. Programming Language Used

The software is written in **Embedded C/C++**, which is the standard programming language used in Arduino-based systems. Embedded C is efficient, fast, and suitable for real-time hardware control.

Reasons for choosing Embedded C:

- Low execution time
- Direct hardware access
- Easy integration with Arduino libraries
- High reliability in embedded applications

3. Software Architecture

The software is divided into the following main sections:

1. Variable declaration
2. Pin configuration
3. System initialization
4. User input processing
5. Motor control logic
6. Turn counting logic
7. Display update
8. Reset and safety logic

This modular structure improves readability, debugging, and future modification.

4. Pin Configuration

Each hardware component is assigned a specific pin on the Arduino Nano:

- Push buttons → Digital input pins
- IR sensor → Digital input pin
- Motor driver inputs → Digital output pins
- LCD display → I2C pins (SDA, SCL)

This pin mapping allows organized and reliable communication between hardware and software

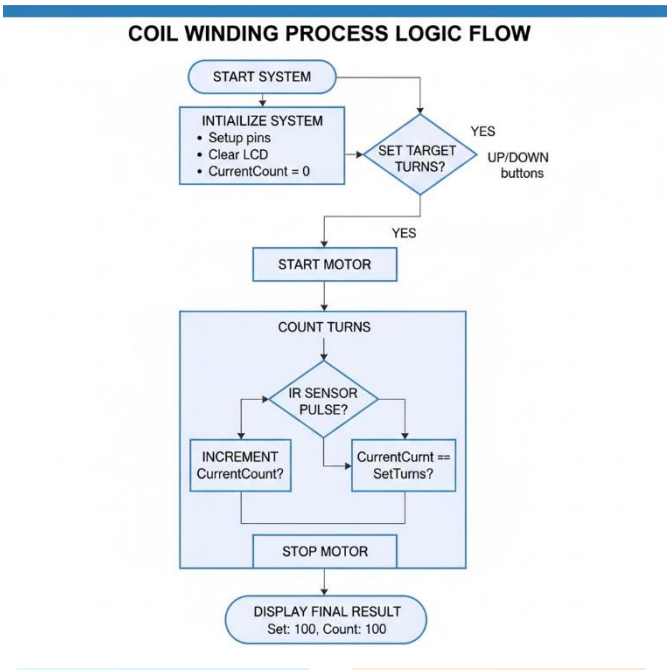
5. Step-by-Step Algorithm

1. Start the system
2. Initialize LCD and display default values
3. Set motor output pins as OUTPUT
4. Read push button inputs
5. Increase or decrease the required turn count using UP/DOWN buttons
6. Display the set turn value on LCD
7. When START button is pressed:
    - Start the motor
    - Monitor IR sensor output
8. For every IR sensor pulse:
    - Increment the current turn counter
    - Update LCD display
9. Compare current turns with set turns
10. If current turns equal set turns:

- Stop the motor automatically

11. If CLEAR button is pressed:
    - Reset current count
    - If held longer, reset both set and current count
12. Repeat the loop continuously

Flowchart



- **Turn Counting Logic**

The IR sensor generates a digital signal for every complete rotation of the winding shaft. The software uses a **flag variable** to ensure that only one count is registered per rotation. This avoids multiple counts due to sensor noise or bouncing.

Key logic features:

- Rising edge detection
- Debounce handling
- Reliable counting at low motor speeds

This logic ensures high accuracy in turn counting.

- **Motor Control Logic**

The DC motor is controlled using the L298 motor driver.

- When motor start condition is true, one input pin is set HIGH and the other LOW
- When motor stop condition is met, both pins are set LOW
- The software alternates motor ON and OFF states to control winding smoothly

This method prevents overheating and allows controlled rotation.

- **LCD Display Handling**

The LCD displays two main parameters:

- **SET TURNS** – Number of turns selected by the user
- **CRT TURNS** – Number of turns completed

The software updates the LCD in real time, allowing the operator to monitor progress easily.

- **Push Button Handling**

Four push buttons are used in the system:

- UP button: Increases turn count
- DOWN button: Decreases turn count
- START button: Begins winding process
- CLEAR button: Resets system

The software uses delays and loops to handle button press duration and avoid false triggering.

- **Error Handling and Safety Features**

The software includes basic safety features:

- Motor stops automatically when target turns are reached
- Reset option available at any time
- Prevents negative or excessive turn values
- Ensures motor is OFF during idle state

These features improve reliability and protect the mechanical system.

- **Advantages of the Software Design**

1. Simple and user-friendly logic
2. Accurate and reliable turn counting
3. Easy to modify and upgrade
4. Low memory usage
5. Compatible with future enhancements

- **Limitations of the Software**

1. No speed control using PWM
2. No automatic tension control
3. Single-direction winding only

- **Code**

```
// Winding Machine Arduino code by: Engineer USMAN AHMAD
// Modified: replace single relay with L298 IN1/IN2 control (no ENA in code)
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

// Set the LCD address to 0x27 for a 16 chars and 2 line display
LiquidCrystal_I2C lcd(0x27, 16, 2);

int a = 0; // number of turns that are set initially
int c = 0; // current turns counter
int clr = 6; // TACTILE BUTTON 4
int up = 3; // TACTILE BUTTON 1
int down = 4; // TACTILE BUTTON 2
int startt = 5; // TACTILE BUTTON 3
int sn = 2; // IR SENSOR

// L298 control pins (IN1 & IN2). ENA must be tied to +5V externally.
int in1 = 7;
int in2 = 8;

int f = 0;

void setup() {
  pinMode(in1, OUTPUT);
  pinMode(in2, OUTPUT);

  pinMode(clr, INPUT_PULLUP);
  pinMode(startt, INPUT_PULLUP);
  pinMode(up, INPUT_PULLUP);
  pinMode(down, INPUT_PULLUP);
  pinMode(sn, INPUT);

  // Ensure motor is stopped initially
  stopMotor();

  lcd.init(); // begins connection to the LCD module
  lcd.backlight(); // turns on the backlight
  lcd.setCursor(0, 0);
  lcd.print("SET TURNS = ");
  lcd.setCursor(12, 0);
  lcd.print(a);
  lcd.setCursor(0, 1);
  lcd.print("CRT TURNS = ");
  lcd.setCursor(12, 1);
  lcd.print(c);
}

void loop()
{
  // Increase turns
  if (digitalRead(up) == HIGH)
  {
    delay(200);
    a = a + 1;
    if (a > 3000) a = 3000;

    while (digitalRead(up) == HIGH)
    {
      lcd.setCursor(12, 0);
      lcd.print(a);
      lcd.print("   ");
      delay(50);
      a = a + 10;
      if (a > 3000) a = 3000;
    }
  }
  // Decrease turns
  if (digitalRead(down) == HIGH)
  {
    delay(200);
    a = a - 1;
    if (a <= 0) a = 0;

    while (digitalRead(down) == HIGH)
    {
      lcd.setCursor(12, 0);
      lcd.print(a);
      lcd.print("   ");
      delay(50);
      a = a - 10;
      if (a <= 0) a = 0;
    }
  }

  lcd.setCursor(12, 0);
  lcd.print(a);
  lcd.print("   ");
```

```
88.
89.  // Start process: rotate motor until c == a, counting pulses from IR sensor
90.  if (digitalRead(startt) == HIGH && c < a)
91.  {
92.   startMotor();
93.   // begin toggling motor to rotate step-by-step (mimicking previous relay logic)
94.   // here ON = IN1 HIGH, IN2 LOW. OFF = both LOW.
95.   delay(250);
96.   while (c != a)
97.   {
98.    if (digitalRead(sn) == HIGH && f == 0)
99.    {
100.      c = c + 1;
101.      f = 1;
102.     }
103.     if (digitalRead(sn) == LOW && f == 1)
104.     {
105.      f = 0;
106.     }
107.
108.     lcd.setCursor(12, 1);
109.     lcd.print(c);
110.     lcd.print("  ");
111.
112.     // Toggle motor similar to your previous timing:
113.     if (digitalRead(in1) == LOW && digitalRead(in2) == LOW && digitalRead(startt) == HIGH)
114.     {
115.      // Turn motor ON (rotate)
116.      startMotor();
117.      delay(200);
118.     }
119.     else if ((digitalRead(in1) == HIGH && digitalRead(in2) == LOW) && digitalRead(startt) == HIGH && c < a)
120.     {
121.      // Turn motor OFF
122.      stopMotor();
123.      delay(200);
124.     }
125.     // If clear button pressed at any time, stop and break
126.     if (digitalRead(clr) == HIGH)
127.     {
128.      startMotor();
129.      delay(200);
130.      break;
131.     }
132.    }
133.
134.    // ensure motor stopped when loop ends
135.    stopMotor();
136.    delay(10);
137.   }
138.
139.   // Clear / reset behavior: single press resets c, long press resets both a and c
140.   if (digitalRead(clr) == HIGH)
141.   {
142.    c = 0;
143.    lcd.setCursor(12, 1);
144.    lcd.print(c);
145.    lcd.print("  ");
146.    delay(200);
147.    while (digitalRead(clr) == HIGH)
148.    {
149.     delay(500);
150.     if (digitalRead(clr) == HIGH)
151.     {
152.      a = 0;
153.      lcd.setCursor(12, 0);
154.      lcd.print(a);
155.      lcd.print("  ")


156.  // Helper: start motor rotating one direction
157.  void startMotor() {
158.   digitalWrite(in1, HIGH);
159.   digitalWrite(in2, LOW);
160.  // Helper: stop motor (both low)
161.  void stopMotor() {
162.   digitalWrite(in1, LOW);
163.   digitalWrite(in2, LOW);
```

- **Advantages of Automatic Coil Winding Machine**

1. **High Accuracy in Turn Counting**

✓ Uses an IR sensor for precise rotation detection
✓ Eliminates manual counting errors
✓ Ensures exact number of coil turns

2. **Reduced Human Effort**

✓ Fully automatic winding operation
✓ Minimal operator involvement
✓ Reduces physical strain and fatigue

3. **Improved Winding Consistency**

- ✓ Uniform coil turns
- ✓ Better electrical performance of coils
- ✓ Reduced rejection rate

4. **Time Efficiency**

- ✓ Faster winding compared to manual methods
- ✓ Suitable for repetitive tasks
- ✓ Increases productivity

5. **Low Cost**

- ✓ Uses easily available components
- ✓ Affordable for students and small workshops
- ✓ Cost-effective alternative to industrial machines

6. **Reliability and Safety**

- ✓ Motor driver protects control circuitry
- ✓ Automatic motor stop after completion
- ✓ Stable and controlled operation

• Disadvantages of Automatic Coil Winding Machine

1. **Limited Speed Control**

- ✓ Motor runs at fixed speed
- ✓ No PWM-based speed variation
- ✓ Not suitable for all wire types

2. **No Automatic Wire Tension Control**

- ✓ Wire tension must be adjusted manually
- ✓ Risk of loose or tight winding
- ✓ Not ideal for very thin wire

3. **Single-Axis Operation**

- ✓ Only one winding direction
- ✓ Does not support complex winding patterns
- ✓ Not suitable for multi-layer coils

4. **Manual Setup Required**

- ✓ Initial coil setup is manual
- ✓ Spool alignment and wire feeding must be done carefully

• Applications of Automatic Coil Winding Machine

1. **Transformer Coil Winding**

- ✓ Used for winding small transformers
- ✓ Ensures uniform turns and better efficiency

2. **Inductor Manufacturing**

- ✓ Suitable for choke coils and inductors
- ✓ Accurate turn count improves inductance value

3. **Motor and Generator Coil Winding**

- ✓ Useful for small DC motor coils
- ✓ Suitable for repair and rewinding applications

4. **Educational Laboratories**

- ✓ Demonstrates automation and control systems
- ✓ Useful for teaching embedded systems and mechatronics

5. **Small-Scale Industries**

- ✓ Ideal for workshops and startups
- ✓ Reduces dependency on skilled labor

6. **Repair and Maintenance Shops**

- ✓ Used for rewinding damaged coils
- ✓ Saves time and improves accuracy

• Future Scope

1. **Speed Control Using PWM**

- ✓ Addition of PWM control for variable motor speed
- ✓ Allows adjustment based on wire thickness and coil type

2. Automatic Wire Tension Control

- ✓ Integration of tension sensors or spring-loaded mechanisms
- ✓ Ensures uniform winding tension and improved coil quality

3. Multi-Layer and Bidirectional Winding

- ✓ Support for layered winding patterns
- ✓ Forward and reverse winding capability

4. Digital Encoder for Higher Accuracy

- ✓ Replacement of IR sensor with rotary encoder
- ✓ Improves precision at higher speeds

5. Touch Screen Interface

- ✓ Use of TFT or touch display for better user interaction
- ✓ Easier parameter selection and monitoring

6. Data Logging and Connectivity

- ✓ Storage of winding data using SD card
- ✓ Integration with IoT platforms for remote monitoring

References

1. Massimo Banzi, *Getting Started with Arduino*, 2nd Edition, Maker Media Inc., USA, 2015.
2. Simon Monk, *Programming Arduino: Getting Started with Sketches*, McGraw-Hill Education, 2016.
3. R. S. Khurmi and J. K. Gupta, *Theory of Machines*, S. Chand & Company Ltd., New Delhi, India.
4. Muhammad Ali Mazidi, Janice Gillispie Mazidi, and Rolin D. McKinlay, *The AVR Microcontroller and Embedded Systems: Using Assembly and C*, Pearson Education, 2014.
5. Arduino Official Documentation, "Arduino Nano Technical Specifications and Pin Configuration," Arduino.cc.
6. Texas Instruments, *L298 Dual Full-Bridge Driver Datasheet*, STMicroelectronics.
7. Barrett Williams, *DC Motors, Speed Controls, Servo Motors and Steppers*, McGraw-Hill, 2012.
8. B. L. Theraja and A. K. Theraja, *A Textbook of Electrical Technology – Volume II*, S. Chand & Company Ltd.
9. S. Franco, *Design with Operational Amplifiers and Analog Integrated Circuits*, McGraw-Hill Education.
10. Datasheet of Infrared (IR) Sensor Module, Vishay Semiconductors.
11. Hitachi Ltd., *HD44780 LCD Controller Datasheet*.
12. John Bird, *Electrical and Electronic Principles and Technology*, 5th Edition, Routledge, UK.
13. R. Bates, *Embedded Systems Design with the Atmel AVR Microcontroller*, Academic Press.
14. Online Tutorials and Technical Articles from:
    - www.arduino.cc
    - www.electronics-tutorials.ws
    - www.allaboutcircuits.com
15. Research papers and student project reports on "Automatic Coil Winding Machine" published in IEEE and other academic journals.

Thank you !