# **CHAPTER 1**

# **INTRODUCTION**

The RFID Attendance System using the SPI protocol is an innovative and efficient solution for managing attendance in various environments like schools, colleges, and workplaces. This system utilizes Radio Frequency Identification (RFID) technology to uniquely identify individuals based on RFID tags, typically in the form of cards or keychains. The system operates using the Serial Peripheral Interface (SPI) protocol, a synchronous data transfer protocol commonly used for short-distance communication between microcontrollers and peripheral devices. In this setup, the RFID reader, which is equipped with an SPI interface, communicates with a microcontroller, such as an Arduino or Raspberry Pi, to read the RFID tag data. The SPI protocol ensures fast and reliable data transfer between the RFID reader and the microcontroller. Once a tag is scanned, the system compares the RFID data with a pre-programmed database, and if a match is found, it registers the attendance. This method offers high-speed data transmission, minimizing the delay between scanning and registration. The RFID Attendance System can also be integrated with a real-time clock (RTC) module to accurately record the time of entry. Additionally, the system can be expanded to send attendance data to a server or cloud for further processing, reporting, or analysis. The use of SPI ensures that the system is not only reliable but also scalable and capable of handling a large number of users. Overall, this approach offers a secure, efficient, and easy-to-use solution for managing attendance in diverse settings

**CHAPTER 2**

# **Literature Survey**

An RFID-based attendance system using the SPI protocol offers an efficient and reliable solution for tracking attendance in various settings, such as schools, universities, and workplaces. The core of this system involves Radio Frequency Identification (RFID) technology, which uses electromagnetic fields to automatically identify and track RFID tags attached to objects or individuals. The SPI (Serial Peripheral Interface) protocol, a synchronous data transfer protocol, plays a crucial role in enabling communication between the microcontroller and the RFID reader, ensuring swift and accurate data exchange. RFID readers capture the unique ID from RFID tags, and the microcontroller processes the data to register attendance. Using SPI allows for high-speed data transfer and minimizes interference from other devices, making it ideal for real-time applications. Additionally, SPI’s simplicity and ease of integration with various microcontrollers make it a preferred choice in embedded systems. Literature suggests that RFID-based attendance systems are not only secure but also offer scalability, allowing for large-scale implementations. Moreover, these systems often incorporate features such as automatic time-stamping, reporting, and remote monitoring. Several studies highlight the system's effectiveness in enhancing operational efficiency and reducing human error in attendance tracking. The integration of SPI with RFID technology allows for fast data transfer, low power consumption, and robust performance in various environmental conditions, making it a highly suitable solution for modern attendance management. Overall, this literature survey emphasizes the potential of SPI-based RFID systems in improving automation and streamlining attendance processes across different industries.

**CHAPTER 3**

# **Methodology**

Figure 3.1: Block Diagram

**3.1 NodeMCU (ESP8266 )**

The NodeMCU ESP8266 is a powerful and versatile platform designed for Internet of Things (IoT) development. The ESP8266 is a cost-effective Wi-Fi microchip known for its capability to enable wireless communication in IoT applications. NodeMCU, on the other hand, is an open-source firmware and development kit that simplifies the process of prototyping and programming the ESP8266. With built-in Wi-Fi connectivity, the NodeMCU ESP8266 allows devices to connect to the internet wirelessly, making it suitable for a wide range of IoT projects. One notable feature is its support for the Lua scripting language, providing a high-level programming environment for developers. Additionally, it is compatible with the Arduino IDE, allowing those familiar with Arduino to use the NodeMCU platform. Equipped with General Purpose Input/Output (GPIO) pins, the ESP8266 facilitates interfacing with various electronic components, making it ideal for applications such as home automation and sensor networks. The NodeMCU ESP8266 has garnered significant community support, resulting in an extensive collection of libraries and documentation, making it a popular choice for rapid IoT prototyping and development.

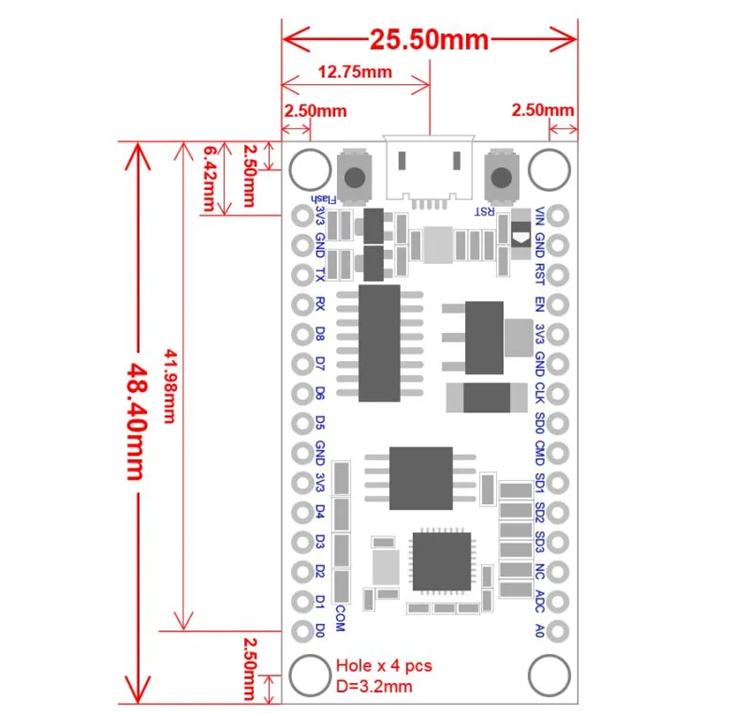


Figure 3.2 NodeMCU 2D View

**NodeMCU Specification:**

The NodeMCU development board is based on the ESP8266 microcontroller, and different versions of NodeMCU boards may have slight variations in specifications. As of my knowledge cutoff in January 2022, here are the general specifications for the NodeMCU ESP8266 development board:

**1. Microcontroller:** ESP8266 Wi-Fi microcontroller with 32-bit architecture.

**2. Processor:** Tensilica L106 32-bit microcontroller.

**3. Clock Frequency:** Typically operates at 80 MHz.

**4. Flash Memory:**

● Built-in Flash memory for program storage.

● Common configurations include 4MB or 16MB of Flash memory.

**5. RAM:** Typically equipped with 80 KB of RAM.

**6. Wireless Connectivity:**

● Integrated Wi-Fi (802.11 b/g/n) for wireless communication.

● Supports Station, SoftAP, and SoftAP + Station modes.

**7. GPIO Pins:** Multiple General Purpose Input/Output (GPIO) pins for interfacing with sensors, actuators, and other electronic components.

**8. Analog Pins:** Analog-to-digital converter (ADC) pins for reading analog sensor values.

**9. USB-to-Serial Converter:** Built-in USB-to-Serial converter for programming and debugging.

**10. Operating Voltage:** Typically operates at 3.3V (Note: It is crucial to connect external components accordingly to avoid damage).

**11. Programming Interface:** Programmable using the Arduino IDE, Lua scripting language, or other compatible frameworks.

**12. Voltage Regulator:** Onboard voltage regulator for stable operation.

**13. Reset Button:** Reset button for restarting the board.

**14. Dimensions:** Standard NodeMCU boards often have dimensions around 49mm x 24mm.

**15. Power Consumption:** Low power consumption, making it suitable for battery-operated applications.

**16. Community Support:** Active community support with extensive documentation and libraries.

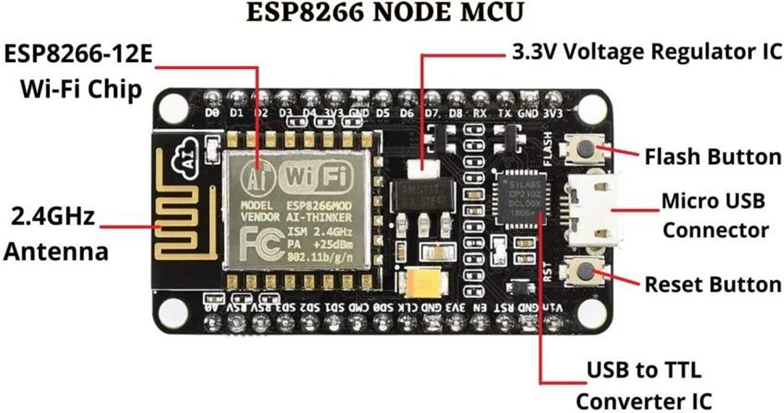
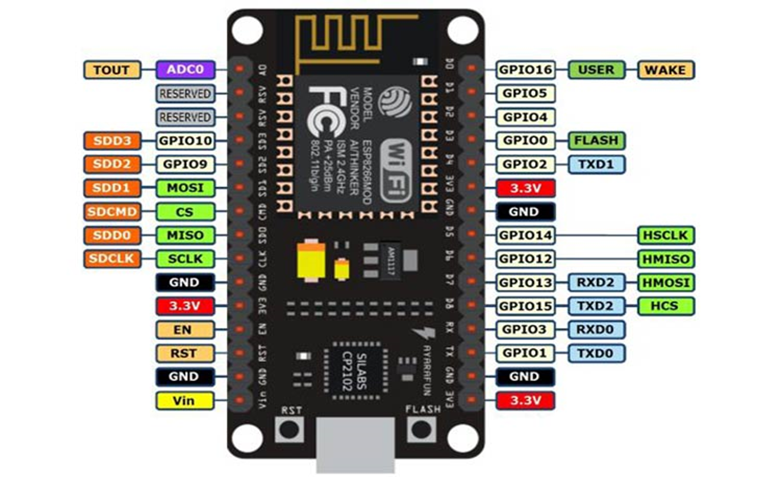


Figure 3.3: NodeMCU Parts

The NodeMCU ESP8266 development board typically has GPIO (General Purpose Input/Output) pins that can be used for various purposes, including interfacing with sensors, actuators, and other electronic components. Below is a common pinout configuration for the NodeMCU development board

Figure 3.4: NodeMCU ESP8266 Pinout

ADC | A0 | GPIO16

EN | Enable | GPIO14

D0 | GPIO16 | GPIO12

D1 | GPIO5 | GPIO13

D2 | GPIO4 | GPIO15

D3 | GPIO0 | GPIO2

D4 | GPIO2 | GPIO9

D5 | GPIO14 | GPIO10

D6 | GPIO12 | GPIO3

D7 | GPIO13 | GPIO1

D8 | GPIO15 | TX (GPIO1)

D9 | GPIO3 (RX) | RX (GPIO3)

D10 | GPIO1 (TX) | D11 (MOSI)

D11 | MOSI | D12 (MISO)

D12 | MISO | D13 (SCK)

**ADC**: Analog-to-Digital Converter pin for reading analog sensor values.

**EN** (Enable): Enable pin.

**D0-D8**: Digital GPIO pins.

**D9 (RX) and D10 (TX)**: Serial communication pins for programming and debugging.

**D11 (MOSI), D12 (MISO), D13 (SCK**): Pins used for SPI communication.

**D14 (SDA) and D15 (SCL)**: Pins used for I2C communication.

It's important to note that GPIO pins labeled as "D" (Digital) are typically used for general-purpose digital input/output. Additionally, GPIO pins labeled as "A" (Analog) can be used as analog inputs with the ADC. GPIO pins 6, 7, 8, 9, 10, and 11 have additional functions, so it's advised to refer to the specific NodeMCU documentation for detailed information on pin functionality and capabilities.

**ARDUINO UNO**

Arduino is an open-source electronics platform based on simple, user-friendly hardware and software, which has become a popular choice for hobbyists, students, and professionals engaged in prototyping and electronics projects. The core components of an Arduino system are microcontroller boards, which can be programmed to sense and control physical objects. The platform is characterized by its use of a microcontroller, typically from the Atmel AVR family (like the ATmega328 on the Arduino Uno), to manage tasks ranging from blinking LEDs to complex operations like sensor integration and wireless communication.

The hardware of Arduino boards comes in various models, such as Arduino Uno, Arduino Mega, and Arduino Nano, each designed for specific types of projects. These boards have digital and analog input/output pins that can be used to interface with a wide variety of sensors, motors, LEDs, and other components. The boards also feature built-in USB ports for programming and power, along with a voltage regulator and serial communication capabilities. The Arduino platform is complemented by a vast ecosystem of compatible shields and modules that simplify adding features like Wi-Fi, Bluetooth, or motor control.

****

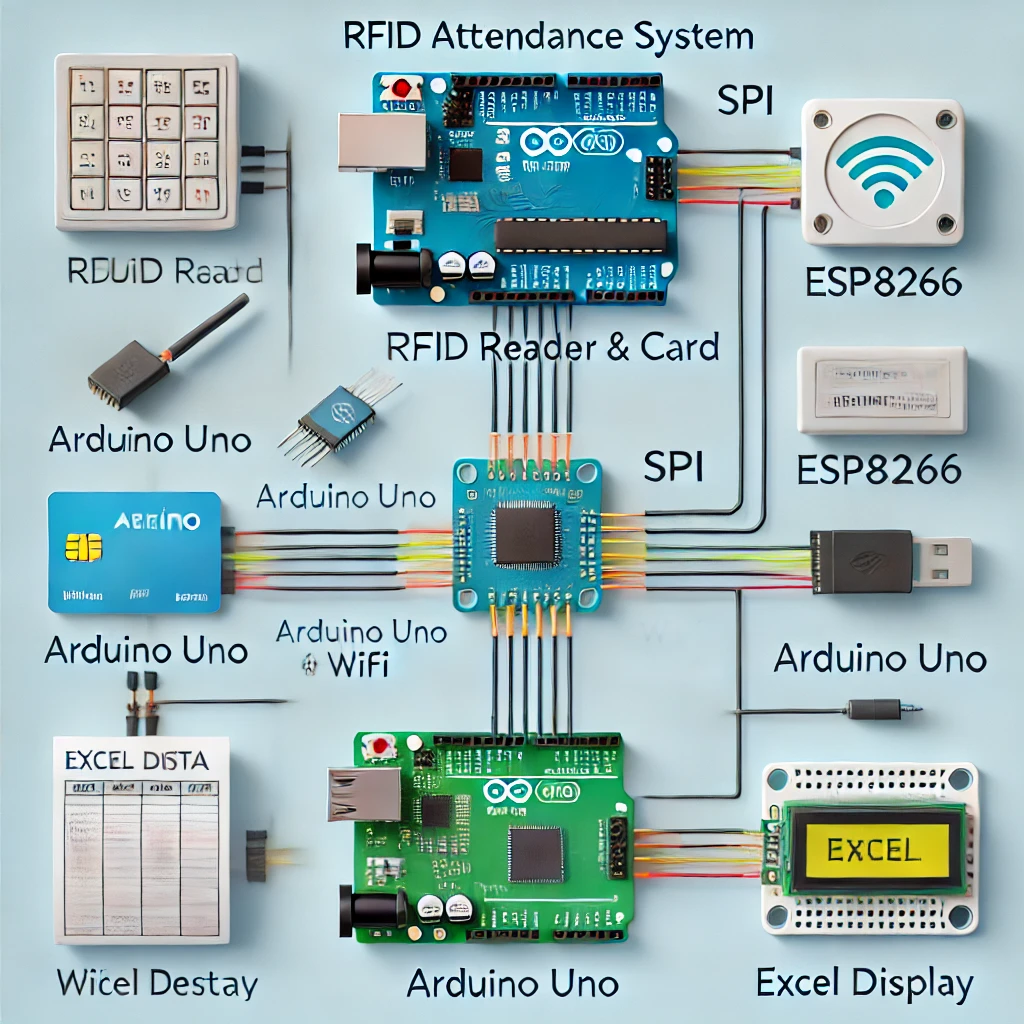
Radio Frequency Identification (RFID) is a technology used for automatic identification and data capture (AIDC). It utilizes electromagnetic fields to transfer data between an RFID reader and a tag attached to an object or individual. An RFID system typically consists of three components: the RFID tag, the RFID reader, and the backend system.



OLED (Organic Light Emitting Diode) is a display technology that utilizes organic compounds to produce light when an electric current is applied. Unlike traditional LED displays, which require a backlight, OLED panels are self-illuminating, meaning each individual pixel emits its own light. This characteristic leads to several advantages over other display technologies, such as LED and LCD.



**Block D**

****

### **Components and Connections:**

1. **RFID Reader and RFID Card (RC522)**
   * **Connections**:
     + **SDA** → Pin 10 on Arduino Uno
     + **SCK** → Pin 13 on Arduino Uno (SPI Clock)
     + **MOSI** → Pin 11 on Arduino Uno (SPI Master Out)
     + **MISO** → Pin 12 on Arduino Uno (SPI Master In)
     + **IRQ** → Not connected (optional)
     + **GND** → Ground on Arduino
     + **RST** → Pin 9 on Arduino Uno (for resetting the module)
     + **3.3V** → 3.3V on Arduino Uno
2. **OLED Display (I2C)**
   * **Connections**:
     + **VCC** → 5V on Arduino Uno
     + **GND** → Ground on Arduino Uno
     + **SCL** → A5 on Arduino Uno (SCL for I2C communication)
     + **SDA** → A4 on Arduino Uno (SDA for I2C communication)
3. **ESP8266 (WiFi Module)**
   * **Connections**:
     + **VCC** → 3.3V (important to use 3.3V, as ESP8266 doesn’t tolerate 5V)
     + **GND** → Ground on Arduino Uno
     + **TX** → Pin 2 on Arduino Uno (through a level shifter if possible or with a resistor divider for safe voltage)
     + **RX** → Pin 3 on Arduino Uno
     + **CH\_PD** → 3.3V (to enable the chip)
   * **Note**: Use the SoftwareSerial library in Arduino to communicate with ESP8266 using pins 2 and 3.

### **Circuit Diagram Layout**

You would arrange the components as described, ensuring that:

* The **RFID Reader** connects to Arduino using the SPI pins.
* The **OLED** connects to the I2C pins (A4 and A5).
* The **ESP8266** connects to pins 2 and 3 for TX and RX, using SoftwareSerial.

### **Code Workflow**

* **RFID Reader**: Detects the card, reads the unique ID, and sends it to Arduino Uno.
* **Arduino Uno**:
  + Receives the ID, verifies it, and sends attendance data to ESP8266.
  + Displays the attendance status on the OLED.
* **ESP8266**: Connects to WiFi and sends the attendance data to an online database (like Google Sheets or Excel) via HTTP requests.

**CHAPTER 4**

# **Design and Coding**

# 

**CHAPTER 5**

# **Conclusion**