

1.0 INTRODUCTION

AIM(S)

Apply web technology to condition monitoring and control of IoT sensors and actuators

LEARNING OUTCOMES

Upon the successful completion of this module, the student should be able to demonstrate the ability to:

1. Implement an Internet of Things sensor/actuator and server link
2. Apply appropriate methods and tools to create suitable management and monitoring software

1.1 NODEMCU

The Node MCU is an open source firmware and development kit that helps you to prototype your IoT product with ArduinoIDE or in few Lau script lines.

It includes firmware which runs on the ESP8266 Wi-Fi SoC. And hardware which is based on the ESP-12 module. In this tutorial we explain how to use NodeMCU with Arduino IDE.

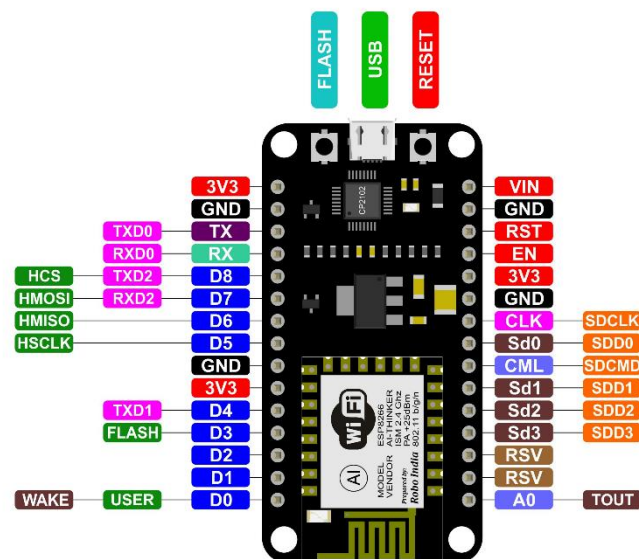


Figure 1: NodeMCU-pinout

1.2 CONNECT NODEMCU WITH ARDUINO IDE

To connect the NodeMCU with Arduino IDE you need to follow the following steps.

Please note that these might vary from one version to another.

- Open the Arduino IDE.
- Go to File -> Preferences -> Additional Boards Manager URLs:
http://arduino.esp8266.com/stable/package_esp8266com_index.json -> click OK
- Close the IDE and open it up again.
- Go to Tools -> Board (where you'd select your version of Arduino) -> Boards Manager, find the ESP8266 and click Install. You now should be able to use the ESP8266 as an Arduino. Simply select the NODEMCU 1.0 as your board with Port and you should be ready to code.
- Now, with ESP8266 board installed to Arduino IDE, we can program NodeMCU using Arduino IDE directly.

1.3 THE ARDUINO SKETCH

The Arduino platform was built with the beginner in mind. Compared to a normal C program the Arduino IDE hides a few things from you to simplify the setup. First of all you do not have to create a makefile to build your code into an executable binary. The Arduino IDE also includes a default header file for you: `#include "Arduino.h"`. This contains all definitions needed for a regular Arduino program.

Another important change compared to a regular C/C++ program are the two default functions `setup()` and `loop()`. The first will be only called once during startup, while the `loop()` method will be called repeatedly. On a normal Arduino hardware (ATmega chip) you can theoretically write code and never leave the `loop()` method again. The ESP8266 is a bit different here. If your operations run for too much time a so-called watchdog will reset the ESP8266. You can prevent this by allowing the controller to do important operations while you are still in the main loop. Calling `yield()` or `delay(ms)` will do this.

LIST OF THE COMPONENT IN YOUR LAB BOX

Package List for each kit
1 x Arduino Nano v3
1 x NodeMCU Mini
1 x HC-05 Serial Bluetooth Module
1 x IR receiver
1 x IR Transmitter
1 x Infrared Remote Controller
1 x 28BYJ-48 Stepper motor-5V
1 x ULN2003 Stepper Motor Driver
1 x Tower Pro 9g SG-90 Mini Servo
1 x MPU6050 3-Axis Acceleration Gyroscope 6DOF Module
1 x HC-SR04 Ultrasonic Range Measurement Module
1 x DHT11 Temperature& Humidity Module
1 x Light Sensor Module
1 x EC11 Encoder
1 x 1-Channel Relay Module-10A
1 x Analog 2-axis Thumb Joystick with Select Button
2 x Hat Button Module
1 x 10mm RGB LED Module
1 x Sealed Membrane 4X4 Button Pad With Sticker
1 x 0.96" I2C OLED 128x64- Blue
1 x Breadboard- 8.3x5.5cm
1 x 40 pin dual male splittable jumper wire - 20cm
1 x 40 pin dual female splittable jumper wire- 20cm
1 x 40 pin male female splittable jumper wire - 20cm
1 x Micro USB cable